Linux-Mandrake

Manual de Referencia

MandrakeSoft

Abril 2001

http://www.linux-mandrake.com/

Linux-Mandrake : Manual de Referencia
por MandrakeSoft

Tabla de contenidos

Tabla de figuras

Prefacio

1. Nota legal

2. Acerca de Linux-Mandrake

Linux-Mandrake es una distribución *GNU/Linux* soportada por **MandrakeSoft** S.A. **MandrakeSoft** nació en la *Internet* en 1998 con el objetivo principal de proporcionar un sistema *GNU/Linux* amigable y fácil de usar. Los dos pilares de **MandrakeSoft** son el código abierto y el trabajo colaborativo.

2.1. Contacte a la comunidad Mandrake

A continuación tiene varios vínculos a la Internet que lo llevan a varias fuentes relacionadas con **Linux-Mandrake**. Si desea saber más acerca de la compañía **MandrakeSoft**, conéctese al sitio *web* (http://www.mandrakesoft.com/) de la misma. Allí se encuentra el sitio para la distribución **Linux-Mandrake** (http://www.linux-mandrake.com/) y todos sus derivados.

Antes que nada, **MandrakeSoft** se enorgullece en presentar su nueva plataforma abierta de ayuda. MandrakeExpert (http://www.mandrakeexpert.com) no es sólo otro sitio *web* donde la gente ayuda a otras con sus problemas con las computadoras a cambio de honorarios prepagados, que se deben pagar sin importar la calidad del servicio recibido. Este sitio ofrece una experiencia nueva basada en la confianza y el placer de premiar a otros por sus contribuciones.

Además de esa plataforma, MandrakeCampus (http://www.mandrakecampus.com/) proporciona a la comunidad *GNU/Linux* cursos de entrenamiento y educación abiertos sobre todas las tecnologías y temas relacionados con la filosofía de código abierto, y a los maestros, tutores, y alumnos con un lugar donde pueden intercambiar conocimientos.

Hay un sitio para el “mandrakeólico” denominado Mandrake Forum (http://www.mandrakeforum.com/): un sitio primario para los consejos, trucos, rumores, pre-anuncios, noticias semi-oficiales, y más relacionados con

Linux-Mandrake. Además, este es el único sitio *web* interactivo que mantiene **MandrakeSoft**, por lo tanto, si tiene algo que decirnos, o algo que quiera compartir con otros usuarios, no busque más: ¡este es un lugar para hacerlo!

En la filosofía del código abierto, **MandrakeSoft** está ofreciendo varias formas de soporte (http://www. Linux-Mandrake.com/en/ffreesup.php3) para las distribuciones **Linux-Mandrake**. En particular, está invitado a participar en las distintas Listas de correo (http://www.Linux- Mandrake.com/en/flists.php3), donde toda la comunidad de **Linux-Mandrake** demuestra su vivacidad y bondad.

2.2. Soporte a Mandrake

Para los muy talentosos entre ustedes, sus habilidades serán sumamente útiles para alguna de las muchas tareas necesarias para realizar un sistema **Linux-Mandrake**:

- Empaquetado: un sistema *GNU/Linux* está compuesto principalmente por programas recogidos de la *Internet*. Estos programas tienen que empaquetarse de forma tal que puedan funcionar juntos.
- Programación: hay muchísimos proyectos que **MandrakeSoft** soporta directamente: encuentre el que más le atraiga, y proponga su ayuda al desarrollador principal.
- Internationalización: la traducción de las páginas *web*, los programas, y su documentación respectiva.
- Documentación: por último pero no por menos, el libro que Ud. está leyendo en este momento necesita de mucho esfuerzo para mantenerse actualizado con la evolución rápida del sistema.

Consulte la página de contribuyentes (http://www.mandrakesoft.com/labs/) para saber más acerca de la forma en la que Ud. puede ayudar a la evolución de **Linux-Mandrake**.

3. Autores y traductores

Las personas siguientes contribuyeron a la elaboración de los manuales de **Linux-Mandrake**:

- Yves Bailly
- Camille Bégnis
- Francis Galiègue
- Hinrich Göhlmann,
- Fabian Mandelbaum
- Roberto Rosselli Del Turco
- Christian Roy
- Stefan Siegel
- Marco De Vitis
- Todos los autores de los documentos que reprodujimos aquí (ver la lista en *Nota legal*, página 11)

También han participado, en mayor o menor grado: Jay Beale, Hoyt Duff, Carsten Heiming, Damien Krotkine, John Rye.

4. Las herramientas usadas en la elaboración de este manual

Este manual se escribió en *DocBook*. *Perl* y *GNU Make* se usaron para administrar el conjunto de archivos involucrados. Los archivos fuente SGML se procesaron con `openjade` y `jadetex` usando las hojas de estilo de Norman Walsh. Las instantáneas de pantallas se tomaron con `xwd` y *GIMP* y se convirtieron con `convert` (del paquete *ImageMagick*). Todas estas piezas de software están disponibles en su distribución **Linux-Mandrake**, y todas las partes son software libres.

5. Nota del Editor

Como podrá notar a medida que pasa de un capítulo a otro, este libro es un compendio escrito por varios autores. Incluso cuando se tomó mucho cuidado en asegurar la consistencia técnica y del idioma, obviamente se conserva el estilo de cada autor.

Algunos autores escriben en Inglés (idioma "oficial" de la documentación), aunque puede no ser su lengua materna. Por lo tanto, puede notar algunas construcciones idiomáticas extrañas; no dude en hacernos saber sobre esto si algo no le parece claro.

Soy de Argentina y los términos de informática que utilizamos aquí pueden no ser los mismos a los empleados en otros países de habla hispana (mouse en vez de ratón, archivo en vez de fichero, etc.), sin embargo he tratado de utilizar términos que puedan ser comprendidos por todos. Espero que la elección haya sido adecuada **:-)**

Siguiendo la filosofía del código abierto (*Open Source*), ¡las contribuciones son muy bienvenidas! Ud. puede proporcionar mucha ayuda a este proyecto de documentación de maneras diferentes. Si tiene un montón de tiempo, puede escribir un capítulo completo. Si habla una lengua extranjera, puede ayudar con la internacionalización de este libro. Si tiene ideas acerca de como mejorar el contenido, háganoslo saber; ¡incluso son bienvenidas las advertencias sobre errores de tecleo u ortografía!

Para mayor información sobre el proyecto de documentación de **Linux-Mandrake**, por favor contacte al administrador de la documentación (mailto:documentation@mandrakesoft.com).

6. Convenciones usadas en este libro

6.1. Convenciones tipográficas

Para poder diferenciar con claridad algunas palabras especiales del flujo del texto, se utilizan representaciones diferentes. La tabla siguiente le muestra un ejemplo de cada palabra o grupo de palabras especiales con su representación real y lo que esto significa.

Ejemplo formateado	Significado
i-nodo	Este formateo se usa para representar un término técnico explicado en el *Glosario*.
`ls -lta`	Indica comandos, o argumentos a dichos comandos. Este formateo se aplica a los comandos, las opciones y los nombres de archivos. Vea también la sección sobre "*Sinopsis de comandos, página 14*"
ls(1)	Referencia a una página Man. Para obtener la página en un *shell (o línea de comandos), simplemente ingrese man 1 ls*.
`$ ls *.pid imwheel.pid $`	Se usa para los textos que aparecen en su pantalla. Incluye a las interacciones con la computadora, los listados de programa, etc.

Ejemplo formateado	Significado
`localhost`	Esto es algún dato literal que por lo general no encaja en alguna de las categorías definidas previamente. Por ejemplo, una palabra clave tomada de un archivo de configuración.
Apache	Esto se usa para los nombres de las aplicaciones. Por ejemplo, no se usa en el nombre de un comando sino en contextos particulares donde el nombre del comando y de la aplicación pueden ser el mismo pero se formatean de maneras diferentes.
Configurar	Esto se usa para las entradas de menú o las etiquetas de las interfaces gráficas en general. La letra subrayada indica la tecla del atajo si es aplicable.
Bus-SCSI	denota una parte de una computadora o una computadora en sí misma.
Le petit chaperon rouge	Indica que estas palabras están en un idioma diferente al idioma en el cual está escrito el libro.
¡Atención!	Por supuesto, esto está reservado para las advertencias especiales para significar la importancia de las palabras; léalo en voz alta **:-)**

6.2. Convenciones generales

6.2.1. Sinopsis de comandos

El ejemplo que sigue le muestra los signos que encontrará en este manual cuando describimos los argumentos de un comando:

```
comando <argumento no textual> [-opción={arg1,arg2,arg3}] [argumento
 opcional ...]
```

Estas convenciones son típicas y las encontrará en otros lugares tales como las páginas Man.

Los signos “<” (menor que) y “>” (mayor que) denotan un argumento que no debe ser copiado textualmente, sino que debe llenarse de acuerdo con sus necesidades. Por ejemplo, `<archivo>` se refiere al nombre real de un archivo. Si este nombre es `pepe.txt`, Ud. debería teclear `pepe.txt`, y no `<pepe.txt>` ni `<archivo>`.

Los corchetes “[ ]” denotan argumentos opcionales, los cuales Ud. puede o no incluir en el comando.

Los puntos suspensivos “...” significan que en ese lugar se puede incluir un número arbitrario de elementos.

Las llaves “{ }” contienen los argumentos permitidos en este lugar. Uno de ellos debe ser puesto aquí.

6.2.2. Notaciones especiales

De vez en cuando se le indicará que presione las teclas Ctrl+R. Eso significa que Ud. debe presionar y mantener presionada la tecla Ctrl mientras presiona la tecla R también. Lo mismo aparece y vale para las teclas Alt y Shift.

También acerca de los menús, ir a la opción del menú Archivo→Resumir (**Ctrl+R**) significa: hacer click sobre el texto Archivo sobre el menú (generalmente horizontal en la parte superior de la ventana) y luego sobre el menú vertical que aparece, hacer click sobre la opción Resumir. Adicionalmente, se le informa que puede usar la combinación de teclas Ctrl+R como se describió anteriormente, para lograr el mismo resultado.

6.2.3. Usuarios genéricos del sistema

Siempre que ha sido posible, hemos utilizado dos usuarios genéricos en nuestros ejemplos:

Reina Amidala		Este usuario se crea en el momento de la instalación
Darth Vader		El administrador del sistema crea más tarde a este usuario

Capítulo 1. Introducción

¡Bienvenido, y gracias por usar **Linux-Mandrake**! Este libro está dirigido a las personas que desean bucear en las profundidades de sus sistemas *GNU/Linux* explotando sus enormes capacidades. Este libro está compuesto de tres partes:

- Parte I en ***Linux-Mandrake***: Aquí le presentaremos el uso de la línea de comandos, sus utilidades varias, y las cosas básicas sobre la edición de texto, esencial bajo *GNU/Linux*.

  Comenzamos con un capítulo que le presenta el mundo de *Unix* y más en particular el de *GNU/Linux*. El mismo es necesario para comprender bien los conceptos que se presentan aquí antes de seguir con el capítulo siguiente, dedicado a la línea de comandos. Este capítulo le presenta los utilitarios estándar para manipular archivos y también algunas características útiles proporcionadas por el *shell*.

  Se dedica otro capítulo a la edición de texto. Como la mayoría de los archivos de configuración de *Unix* son texto, puede necesitar editarlos en un ***editor de texto***. Aprenderá como usar dos de los editores de texto más famosos en el mundo *Unix*: el poderoso *Emacs* y el moderno **:-)** *VI*.

  Ahora debería poder realizar algún mantenimiento básico de su sistema. Los dos capítulos siguientes presentan usos prácticos de la línea de comandos y el control de los procesos en general.

- Parte II en ***Linux-Mandrake***: Estos son algunos detalles acerca del núcleo *Linux* y la arquitectura del sistema de archivos.

  Verá en el primer capítulo como se organiza el árbol de archivos. Los sistemas *Unix* tienden a crecer mucho, pero cada archivo tiene su lugar en un directorio específico. Después de leer este capítulo Ud. sabrá donde buscar los archivos dependiendo del rol de los mismos en el sistema.

  Otro capítulo cubre el tema de los ***sistemas de archivos*** y los ***puntos de montaje***. Aquí aprenderá lo que ambos términos significan y verá un ejemplo práctico.

  Se dedicará un capítulo al sistema de archivos de *GNU/Linux*: ext2fs. En él aprenderá más acerca de los tipos de archivos y algunos conceptos adicionales que pueden ser nuevos para Ud. Otro capítulo le presentará el sistema de archivo de *GNU/Linux* especial: `/proc`.

  Luego aprenderá el procedimiento de arranque de **Linux-Mandrake** y como usarlo de manera eficiente.

- Parte III en ***Linux-Mandrake***: finalmente algunos capítulos para los usuarios avanzados, más un capítulo muy útil sobre resolución de problemas.

  Comenzamos con un capítulo dedicado a la administración de la impresión y luego la parte de resolución de problemas.

  Finalmente, tres capítulos más dedicados especialmente a las personas que desean convertirse en expertos con *GNU/Linux*. El primero, está dedicado especialmente al núcleo 2.4 y sus cosas específicas. El siguiente, describe la manera de compilar e instalar un núcleo nuevo; y luego viene el capítulo dedicado a la construcción e instalación de software libre.

I. Introducción a GNU/Linux

Capítulo 2. Conceptos básicos de Unix

El nombre "*Unix*" puede ser familiar para algunos de Uds. Incluso hasta puede ser que use un sistema *Unix* en el trabajo, en cuyo caso, este capítulo puede no ser muy interesante para Ud.

Para aquellos de Uds. que nunca lo usaron, la lectura de este capítulo es absolutamente necesaria. El conocimiento de los conceptos que se presentarán aquí contesta un número sorprendentemente alto de preguntas frecuentes de los principiantes en el mundo de GNU/Linux . Similarmente, es probable que algunos de estos conceptos puedan darle pistas para resolver los problemas que Ud. pueda encontrar en el futuro.

2.1. Usuarios y grupos

El concepto de usuarios y grupos es extremadamente importante, ya que tiene una influencia directa sobre todos los demás conceptos que iremos presentando a lo largo del capítulo.

GNU/Linux es un sistema ***multi-usuario*** verdadero, por lo que para poder conectarse a su sistema *GNU/Linux* debe poseer una ***cuenta*** en el mismo. Cuando Ud. creó un usuario durante la instalación, en realidad, creó una cuenta de usuario. Puede recordar que se le pidieron, entre otros, los elementos siguientes:

- el "nombre verdadero" del usuario (de hecho, cualquier nombre que desee),
- un nombre de conexión (o `login`),
- una ***contraseña*** (en verdad **puso** una, ¿cierto? **:-)**).

Los dos parámetros importantes aquí son, el nombre de conexión (comúnmente abreviado *login*) y la contraseña. Estos son los que usará para poder ingresar al sistema.

Otra acción que ocurrió cuando se creó una cuenta de usuario es la creación de un grupo. Predeterminadamente, el programa de instalación habrá creado un grupo por cada usuario. Como veremos más adelante, los grupos son útiles cuando varias personas tienen que compartir archivos. Por lo tanto, un grupo puede contener tantos usuarios como Ud. quiera, y es muy común ver tal separación en sistemas grandes. En una universidad, por ejemplo, Ud. puede tener un grupo por cada departamento, otro grupo para los profesores, y así sucesivamente. La inversa también vale: un usuario puede ser miembro de uno o más grupos, hasta un máximo de diez[1]. Por ejemplo, un profesor de matemáticas puede ser un miembro del grupo de profesores y también ser miembro del grupo de sus queridos estudiantes de matemáticas.

Sin embargo, todo esto no le dice como conectarse. Acá viene.

Si eligió tener la interfaz gráfica en el arranque, su pantalla de conexión se parecerá a la de la Figura 2-1.

---

1. Sin embargo, en versiones futuras de *GNU/Linux* esta limitación puede quitarse.

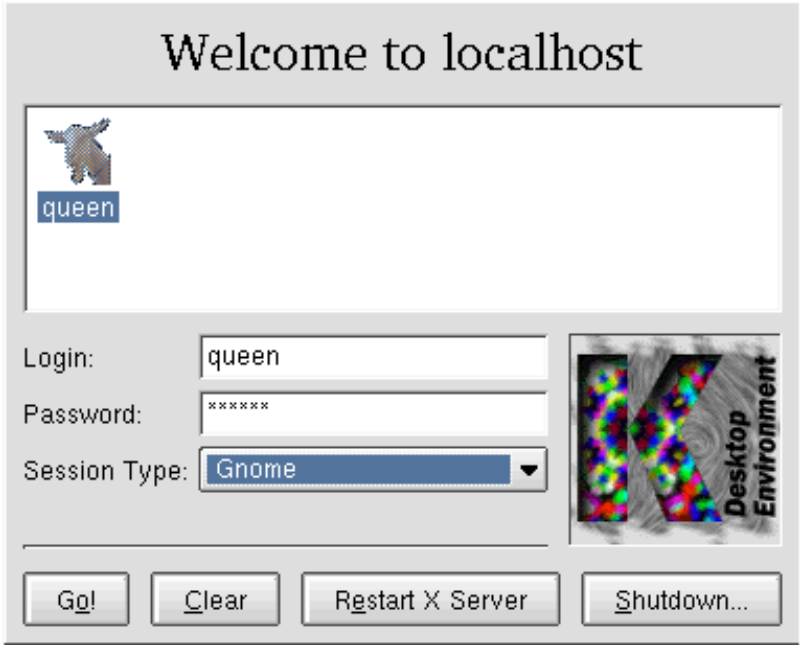

Figura 2-1. Conexión en modo gráfico

Para poder conectarse, debe ingresar su nombre de conexión en el campo de texto denominado Usuario:, luego ingrese su contraseña en el campo de contraseña . Note que tendrá que ingresar su contraseña a ciegas: no hay ***eco*** dentro del campo de texto.

Si está en modo consola , su pantalla será similar a la que se muestra en la Figura 2-2.

Figura 2-2. Conexión en modo consola

Luego tendrá que ingresar su nombre de conexión en el *prompt* Login: y presionar `Intro`, después aparecerá el programa de conexión (denominado *login*) que mostrará el *prompt* Password:, donde deberá ingresar la contraseña para esta cuenta. Debido a que la conexión en la consola no hace eco de los caracteres que representan a la contraseña, deberá tener cuidado cuando teclea su contraseña – a ciegas.

Note que Ud. se puede conectar varias veces usando la misma cuenta sobre ***consola***s adicionales y bajo *X*. Cada sesión que abra es independiente de las otras, e incluso es posible tener varias sesiones *X* abiertas concurrentemente. Predeterminadamente, **Linux-Mandrake** tiene seis ***consolas virtuales*** además de la reservada para la interfaz gráfica. Ud. puede cambiarse a cualquiera de ellas ingresando la secuencia de teclas `Ctrl-Alt-F<n>`[2], donde `<n>` es el número de consola a la cual quiere cambiarse. Predeterminadamente, la interfaz

2. En modo texto también puede presionar `Alt-F<n>`

gráfica está sobre la consola número 7.

Además de la creación de cuentas de usuario, durante la instalación también debe haber notado que *DrakX* (o el programa que Ud. utilizó) le habrá pedido la contraseña de un usuario muy especial: `root` . Este usuario es especial por una razón simple: normalmente esta es la cuenta que tiene el administrador del sistema, que es muy probable que sea Ud. Para la seguridad de su sistema, es muy importante que la cuenta de `root` ¡**siempre** esté protegida por una buena contraseña!

Si se conecta como `root` regularmente es muy fácil cometer un error que puede hacer que su sistema quede inútil; un único error grande puede hacer que esto ocurra. En particular, si no ha proporcionado una contraseña para la cuenta `root`, cualquier usuario puede conectarse usando dicha cuenta y alterar cualquier parte de su sistema (¡incluso de otros sistemas operativos presentes en su máquina!). Obviamente, ¡esto no es una idea muy buena!

Vale mencionar que, internamente, el sistema no lo identifica con su nombre de conexión sino con un número único asignado a este nombre de conexión: el UID (*User ID*, Identificador del usuario). Similarmente, cada grupo se identifica no por su nombre sino por su GID o *Group ID*, (Identificador del grupo).

2.2. Nociones básicas sobre los archivos

Los archivos son otro tópico donde *GNU/Linux* difiere bastante de *Windows* y muchos otros ***sistemas operativos***. Aquí cubriremos las diferencias más obvias, para más información lea el capítulo El sistema de archivos de Linux que ofrece mayor detalle.

La mayor diferencia es consecuencia directa del hecho de que *GNU/Linux* es un sistema multiusuario: cada archivo es de la exclusiva propiedad de un usuario y un grupo. Arriba mencionábamos a los usuarios, y una cosa que no mencionamos es que cada usuario tiene su propio directorio (denominado su ***directorio personal***, o *home* en inglés). El es el ***dueño*** de este directorio, y de los archivos que va a crear subsecuentemente. Él, y nadie más que él.

Sin embargo, esto no sería muy útil si sólo estuviera la noción de propiedad del archivo. Pero hay más: como dueño del archivo, un usuario puede configurar **permisos** sobre sus archivos. Estos permisos permisos distinguen tres categorías de usuarios: el dueño del archivo, todos los usuarios que son miembros del grupo asociado al archivo (denominado también ***grupo dueño***) pero no son el usuario dueño, y los otros, que son todos los usuarios que no son ni el dueño ni miembros del grupo dueño.

Hay tres permisos diferentes:

1. Permiso de lectura (`r` por *Read*, Leer): para un archivo, esto permite que se lea su contenido. Para un directorio, esto permite que se muestren los archivos que contiene (es decir, los archivos en este directorio).
2. Permiso de escritura (`w` por *Write*, Escribir): para un archivo, esto permite que se modifique su contenido. Para un directorio, esto permite que un usuario agregue y/o quite archivos de este directorio, incluso si no es el dueño de esos archivos.
3. Permiso de ejecución (`x` por *eXecute*, Ejecutar): para un archivo, esto permite su ejecución (en consecuencia, normalmente sólo los archivos ejecutables tienen activo este permiso). Para un directorio, esto permite que un usuario lo ***recorra*** (lo que significa poder ingresar a, o pasar por, ese directorio). Note que esto está separado del acceso de lectura: bien puede ser que Ud. pueda recorrer un directorio, ¡pero no leer su contenido!

Todas las combinaciones de estos permisos son posibles. Por ejemplo, Ud. puede autorizar la lectura de un archivo sólo a Ud. mismo y prohibirla a todos los demás usuarios. Incluso puede hacer lo contrario, aunque a primera vista no parezca muy lógico... Como dueño del archivo, también puede cambiar el grupo propietario (solamente si Ud. es miembro del grupo nuevo), e incluso privarse del archivo (es decir, cambiar su dueño). Por supuesto, si Ud. mismo se priva del archivo perderá todos sus derechos sobre el mismo...

Tomemos el ejemplo de un archivo y un directorio. Abajo se muestra el resultado de ingresar el comando `ls -l` desde una ***línea de comandos***:

```
$ ls -l
total 1
-rw-r-----   1 amidala    users           0 Jul  8 14:11 un_archivo
drwxr-xr--   2 darth    users        1024 Jul  8 14:11 un_directorio/
$
```

Los diferentes campos de salida del comando `ls -l` son los siguientes (de izquierda a derecha):

- los primeros diez caracteres representan sucesivamente el tipo de archivo y los derechos asociados al mismo. El primer carácter es el tipo del archivo: contiene un guión (`-`) si es un archivo regular, o una `d` si es un directorio. Hay otros tipos de archivos, de los que hablaremos en el **Manual de Referencia**. Los nueve caracteres que siguen representan los permisos asociados con ese archivo. Aquí puede ver la distinción que se hace entre los diferentes usuarios para el mismo archivo: los primeros tres caracteres representan los derechos asociados con el dueño del archivo, los siguientes tres se aplican a todos los usuarios que pertenecen al grupo pero que no son el dueño, y los últimos tres se aplican a los otros. Un guión (`-`) significa que el permiso no está activo;
- luego viene el número de vínculos del archivo. En el **Manual de Referencia** veremos que los archivos no sólo se identifican por su nombre sino también por un número (el número de ***i-nodo***), y por lo tanto es posible que un archivo en disco tenga varios nombres. Para un directorio, el número de vínculos tiene un significado especial, que también veremos en el **Manual de Referencia**;
- luego viene el nombre del dueño del archivo y el nombre del grupo dueño del mismo;
- y finalmente, se muestra el tamaño del archivo (en ***bytes***) y la fecha de su última modificación.

Ahora observemos en detalle los permisos asociados con cada uno de estos archivos: antes que nada, debemos quitar el carácter que representa el tipo, y para el archivo `un_archivo` obtenemos los derechos siguientes: `rw-r-----`. La interpretación de los mismos es la siguiente:

- los primeros tres (`rw-`) son los derechos del usuario dueño del archivo, en este caso `amidala`. El usuario `amidala`, entonces, tiene el derecho de leer el archivo (`r`), de modificarlo (`w`) pero no de ejecutarlo (`-`);
- los tres siguientes (`r--`) se aplican a todo usuario que no es `amidala` pero que es miembro del grupo `users`: dicho usuario podrá leer el archivo (`r`), pero no podrá modificarlo ni ejecutarlo (`--`);
- y los tres restantes (`---`) se aplican a todo usuario que no es `amidala` ni es miembro del grupo `users`: dicho usuario simplemente no tendrá derecho alguno sobre el archivo.

Para el directorio `un_directorio`, los derechos son `rwxr-xr--`, entonces:

- `darth`, como dueño del directorio, puede listar los archivos que contiene (`r`), agregar o quitar archivos del mismo (`w`), y recorrerlo (`x`);
- cada usuario que no es `darth` pero es miembro del grupo `users`, podrá listar los archivos de ese directorio (`r`), pero no podrá quitar ni agregar archivos (`-`), y lo podrá recorrer (`x`);
- cualquier otro usuario sólo podrá listar el contenido de este directorio (`r--`), y nada más. Incluso no podrá ingresar al directorio.

Hay **una** excepción a estas reglas: `root`. `root` puede cambiar los atributos (permisos, dueño, y grupo dueño) de todos los archivos, incluso si no es el propietario de los mismos. ¡Esto significa que también puede garantizarse la propiedad! Él puede leer archivos sobre los que no tiene permisos, recorrer directorios a los que normalmente no tendría acceso, y así sucesivamente. Y si no tiene un permiso, sólo tiene que adjudicárselo él mismo...

Para finalizar, vale la pena mencionar otra diferencia sobre los nombres de los archivos en el mundo de *Unix* y en el mundo de *Windows*. *Unix* permite mayor flexibilidad y tiene menos limitaciones:

- un nombre de archivo pueden contener cualquier caracter (excepto el ***caracter nulo*** - caracter ASCII 0 - y una `/` que es el separador de directorio), incluso los no imprimibles. Es más, *Unix* distingue entre mayúsculas y minúsculas (***capitalización***): los archivos `leame` y `Leame` son dos archivos diferentes, porque `l` y `L` son dos **caracteres** diferentes;
- como debe haber notado, un nombre de archivo no contiene extensión alguna a menos que Ud. lo prefiera así. Bajo *GNU/Linux* las extensiones no caracterizan el contenido de un archivo, y tampoco lo hacen bajo otros sistemas operativos si es por eso. No obstante, las así llamadas “extensiones del archivo” siempre son muy convenientes. El caracter del punto (`.`) bajo *Unix* es simplemente un caracter entre otros. Vale la pena mencionar que, bajo *Unix* los nombres de archivo que comienzan con un punto son “archivos ocultos”;

2.3. Los procesos

Un ***proceso*** define una instancia de un programa en ejecución y su ***entorno***. De la misma forma que con los archivos, aquí sólo mencionamos las diferencias más importantes. Para una discusión más profunda del tema refiérase al **Manual de Referencia**.

La diferencia más importante está, una vez más, directamente relacionada al concepto de usuarios: cada proceso se ejecuta con los derechos del usuario que lo inició. Internamente, el sistema identifica a los procesos de forma unívoca con un número. Este número se conoce como el PID (*Process ID*, ID del Proceso). A partir de este PID, el sistema sabe, entre otras cosas, quien (es decir, que usuario) ha lanzado el proceso. Entonces, simplemente tiene que verificar si lo que ese proceso pide es “legal”.Por lo tanto, si volvemos al ejemplo del archivo `un_archivo` mencionado anteriormente, un proceso lanzado por el usuario `darth` sólo podrá abrir este archivo en ***modo de sólo lectura***, pero no en el ***modo de lectura-escritura***, ya que los derechos asociados al archivo lo prohíben. Una vez más, `root` es la excepción a la regla...

Un beneficio de esto es que *GNU/Linux* es virtualmente inmune a los virus. Un virus necesita infectar archivos ejecutables para poder operar. Un usuario no tiene los derechos suficientes para los archivos vulnerables del sistema, razón por la cual el riesgo se reduce notablemente. Agregue a esto que los virus son muy raros en el mundo de *Unix* en general. Hasta ahora, sólo han habido tres virus conocidos para *Linux*, y eran completamente inofensivos cuando los iniciaba un usuario no privilegiado. Sólo un usuario puede dañar un sistema activando estos virus, y es, una vez más, `root`.

Sin embargo, existe software anti-virus para *GNU/Linux*, pero para los archivos de *DOS*/*Windows*... La razón de esto es que, cada vez se ven más y más servidores de archivos *GNU/Linux* sirviendo a las máquinas *Windows*, con la ayuda del paquete de software *Samba* (ver el capítulo *Samba* en el **Manual de Referencia**).

Linux hace que sea fácil controlar a los procesos. Una forma de controlarlos es por medio de señales. Con las señales Ud. puede, por ejemplo, suspender un proceso o terminarlo. Simplemente, debe enviar la señal correspondiente al proceso y ya está. Sin embargo, está limitado a enviar señales sólo a los que Ud. inició y no los procesos de otros usuarios. De nuevo, ¡la excepción a la regla es, `root`! En *Control de procesos*, página 61 aprenderá como obtener el PID de un proceso y enviarle señales.

2.4. Breve introducción a la línea de comandos

La línea de comandos es la manera más directa de enviar comandos a la máquina. Si usa la línea de comandos de *GNU/Linux*, rápidamente verá que es mucho más potente y tiene más capacidades que los *prompts* que puede haber usado con anterioridad. La razón de esto es que tiene un acceso directo, no sólo a todas las aplicaciones *X*, sino también a los miles de utilitarios en modo consola (en oposición al modo gráfico) que no tienen su equivalente gráfico, o nunca será posible mostrar en forma de menús y botones todas las opciones y combinaciones posibles.

Pero, admitámoslo, hace falta un poquito de ayuda para poder empezar. Para eso es este capítulo. La primera cosa a hacer, si está en el modo gráfico, es iniciar un emulador de terminal. Tiene un icono que identifica claramente a la terminal (Figura 2-3).

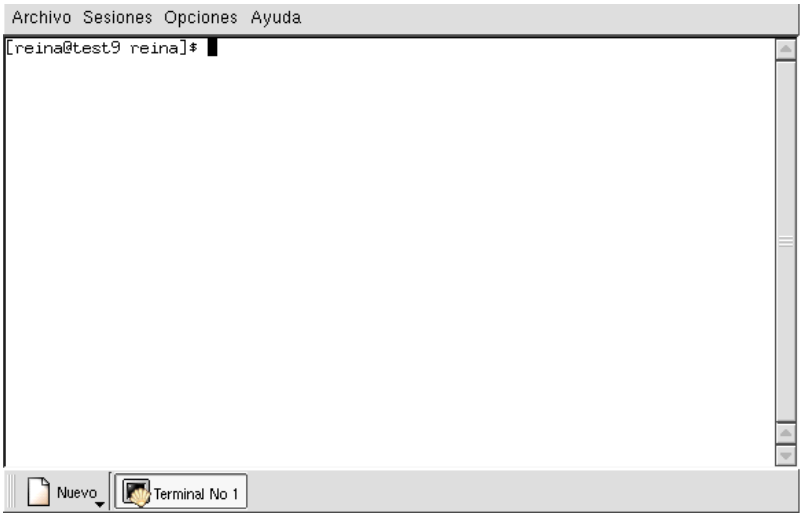

Figura 2-3. El icono de la terminal en el panel de KDE

Lo que obtiene en realidad al iniciar este emulador de terminal es un *shell*. Este es el nombre del programa con el cual Ud. interactúa. Ud. se encontrará frente al prompt:

```
[amidala@localhost] ~ $
```

Esto supone que su nombre de usuario es amidala y que el nombre de su máquina es localhost (que es el caso usual si su máquina no es parte de una red existente). Todo lo que aparece después del *prompt* es lo que tiene que teclear. Note que cuando Ud. es root el signo $ del *prompt* cambia por un signo #. (esto sólo es válido con la configuración predeterminada, ya que puede personalizar todos estos detalles en *GNU/Linux*). El comando para "volverse" root cuando inició un *shell* como usuario no privilegiado es su:

```
# Ingresar la contraseña de root; la misma no aparecerá en la pantalla
[amidala@localhost] ~ $ su
Password:
# exit lo llevará de vuelta a su cuenta de usuario no privilegiado
[root@localhost] exit
[amidala@localhost] ~ $
```

En el libro, en general el *prompt* será representado simbólicamente con un signo $, sea Ud. un usuario no privilegiado o root. Cuando tenga que ser root para ejecutar un comando se la avisará, así que, no se olvide del comando su que mostramos antes. Un signo # al comienzo de una línea de código representará un comentario.

Cuando Ud. ***lanza*** el *shell* por primera vez normalmente se encontrará en su directorio personal. Para mostrar el directorio en donde se encuentra en este momento, ingrese el comando pwd (que significa *Print Working Directory*, Imprimir el directorio de trabajo):

```
$ pwd
/home/amidala
```

Hay unos comandos básicos que veremos ahora, y pronto se dará cuenta de que le serán indispensables.

2.4.1. cd: `Change Directory` (Cambiar de directorio)

El comando `cd` es exactamente el mismo que en *DOS*, con alguna funcionalidad extra. Hace justo lo que su acrónimo indica, cambia el directorio de trabajo. Puede usar `.` para referirse al directorio corriente y `..` para referirse al directorio padre del directorio corriente. Si ingresa `cd` sólo, será llevado de vuelta a su directorio personal. Si ingresa `cd -` será llevado al último directorio en el cual estuvo. Y, finalmente, puede especificar el directorio personal del usuario `darth` ingresando `cd ~darth` (`~` sólo o seguido de `/` significa el directorio personal suyo). Note que, como usuario no privilegiado, normalmente no puede ir a los directorios personales de otros usuarios (a menos que esos usuarios lo hayan autorizado explícitamente o esa sea la configuración predeterminada del sistema), excepto si Ud. es `root`, así que, seamos `root` y practiquemos:

```
$ pwd
/root
$ cd /usr/share/doc/HOWTO
$ pwd
/usr/share/doc/HOWTO
$ cd ../FAQ
$ pwd
/usr/share/doc/FAQ
$ cd ../../lib
$ pwd
/usr/lib
$ cd ~darth
$ pwd
/home/darth
$ cd
$ pwd
/root
```

Ahora, vuelva a ser un usuario no privilegiado.

2.4.2. Algunas variables de entorno y el comando echo

Los procesos tienen sus ***variables de entorno*** y el *shell* le permite verlas directamente con el comando `echo`. Algunas variables interesantes son:

1. `HOME`: esta variable de entorno contiene una cadena que representa su directorio personal.
2. `PATH`: esta variable contiene la lista de todos los directorios en los cuales el *shell* busca los ejecutables cuando Ud. ingresa un comando. Note que predeterminadamente, a diferencia de *DOS*, el *shell* ¡**no** buscará los comandos en el directorio corriente!
3. `USERNAME`: esta variable contiene una cadena que representa su nombre de conexión.
4. `UID` Contiene su identificador de usuario
5. `PS1`: contiene la definición de su *prompt*. Generalmente, es una combinación de secuencias especiales. Puede leer la ***página Man*** de bash(1) para más información.

Para hacer que el *shell* muestre el valor de una variable, debe anteponer al nombre de la misma un `$`. Aquí, el comando `echo` lo ayudará:

```
$ echo Hola
Hola
```

```
$ echo $HOME
/home/amidala
$ echo $USERNAME
amidala
$ echo Hola $USERNAME
Hola amidala
$ cd /usr
$ pwd
/usr
$ cd $HOME
$ pwd
/home/amidala
```

Como puede ver, el *shell* substituye el valor de la variable antes de ejecutar el comando. De no ser así nuestro `cd $HOME` no hubiese funcionado. De hecho, el *shell* primero ha reemplazado `$HOME` por su valor, `/home/amidala`, por lo que la línea se convirtió en `cd /home/amidala`, que es lo que queríamos. Lo mismo vale para `echo $USERNAME` y así sucesivamente.

2.4.3. cat: mostrar el contenido de uno o más archivos en la pantalla

No hay mucho más que decir, este comando simplemente hace eso: mostrar el contenido de uno o más archivos en la salida estándar, normalmente la pantalla:

```
$ cat /etc/fstab
/dev/hda5 / ext2 defaults 1 1
/dev/hda6 /home ext2 defaults 1 2
/dev/hda7 swap swap defaults 0 0
/dev/hda8 /usr ext2 defaults 1 2
/dev/fd0 /mnt/floppy auto sync,user,noauto,nosuid,nodev 0 0
none /proc proc defaults 0 0
none /dev/pts devpts mode=0620 0 0
/dev/cdrom /mnt/cdrom auto user,noauto,nosuid,exec,nodev,ro 0 0
$ cd /etc
$ cat conf.modules shells
alias parport_lowlevel parport_pc
pre-install plip modprobe parport_pc ; echo 7 > /proc/parport/0/irq
#pre-install pcmcia_core /etc/rc.d/init.d/pcmcia start
#alias car-major-14 sound
alias sound esssolo1
keep
/bin/zsh
/bin/bash
/bin/sh
/bin/tcsh
/bin/csh
/bin/ash
/bin/bsh
/usr/bin/zsh
```

2.4.4. less: un paginador

Su nombre es un juego de palabras relacionado al primer *pager* existente bajo *Unix*, que se denominaba `more`[3]. Un ***paginador*** es un programa que permite al usuario ver archivos largos página por página (o, más precisamente, pantalla por pantalla). Hablamos más de `less` que de `more` porque su uso es mucho más intuitivo. Utilice el comando `less` para ver archivos grandes, que no entran en una pantalla. Por ejemplo:

```
less /etc/termcap
```

Para navegar por el archivo, use las teclas de las flechas para arriba y para abajo. Utilice **q** (por *quit*, salir) para salir del programa. En realidad, `less` puede hacer mucho más que eso. De hecho, simplemente presione **h** para la ayuda (en inglés) y eche un vistazo. Pero de todas formas, el objetivo de esta sección es que Ud. sea capaz de leer archivos largos, y ya está cumplido **:-)**

2.4.5. ls: listar archivos

El comando `ls` (*LiSt*, LiStar) es equivalente a `dir` de *DOS*, pero puede hacer mucho más. De hecho, esto se debe en gran parte al hecho de que los archivos también pueden hacer más **:-)** La sintaxis del comando `ls` es la siguiente:

```
ls [opciones] [archivo|directorio] [archivo|directorio...]
```

Si no se especifica archivo o directorio alguno en la línea de comandos, `ls` imprimirá la lista de los archivos del directorio corriente. Sus opciones son muy numerosas y sólo citaremos algunas de ellas:

1. `-a`: listar todos los archivos, incluyendo los ***archivos ocultos*** (en *Unix* los archivos ocultos son aquellos cuyo nombre comienza con un "`.`"); la opción `-A` lista "casi" todos los archivos, lo que significa que se mostrarán todos los archivos que mostraría la opción `-a` excepto "`.`" y "`..`";
2. `-R`: listar "recursivamente", es decir, todos los archivos y sub-directorios del directorio que se menciona en la línea de comandos;
3. `-s`: muestra el tamaño en kilobytes junto a cada archivo;
4. `-l`: muestra información adicional sobre los archivos;
5. `-i`: muestra el número de i-nodo (el número único del archivo en el sistema de archivos, ver el capítulo El sistema de archivos de Linux) junto a cada archivo;
6. `-d`: trata a los directorios de la línea de comandos como si fueran archivos normales en vez de listar su contenido.

Algunos ejemplos:

- `ls -R`: lista recursivamente el contenido del directorio corriente;
- `ls -is images/ ..`: lista los archivos en el directorio `images/` y en el directorio padre, e imprime, para cada archivo, su número de i-nodo y su tamaño en kilobytes.

3. "less" significa "menos", y "more" significa "más"

- `ls -al images/*.png` lista todos los archivos (incluso los archivos ocultos) del directorio `images/` cuyo nombre termina con `.png`. Note que esto también incluye al archivo `.png` si es que existe uno.

2.4.6. Atajos de teclado útiles

Esta sección presentará algunas de las muchas secuencias de tecleo que están disponibles y le harán ganar tiempo. Esta sección asume que Ud. está utilizando el *shell* predeterminado provisto con **Linux-Mandrake**, *Bash*, pero estas secuencias de tecleo también deberían funcionar con otros *shells*.

Primero: las teclas de las flechas. *Bash* mantiene un historial de los comandos que ingresó previamente, el cual puede verse con las teclas de las flechas para arriba y para abajo. Ud. puede remontarse hasta un número de líneas definido en la variable de entorno `HISTSIZE`. Es más, el histórico es persistente de una sesión a otra, por lo que no va a perder los comandos que ingresó en una sesión previa.

Las teclas de las flechas izquierda y derecha mueven el cursor hacia la izquierda y hacia la derecha en la línea corriente, por lo que puede editar sus comandos de esta forma. Pero hay más en materia de edición: `Ctrl+a` y `Ctrl+e`, por ejemplo, lo llevarán al comienzo y al final, respectivamente, de la línea corriente. Las teclas `Backspace` [4] y `Supr` funcionan como se espera. Un equivalente de `Backspace` es `Ctrl+h` y un equivalente de `Supr` es `Ctrl+d`. `Ctrl+k` borrará toda la línea desde la posición del cursor hasta el final de la misma, y `Ctrl+w` borrará la palabra delante del cursor.

Ingresar `Ctrl+d` en una línea en blanco le hará cerrar la sesión corriente, lo cual es mucho más corto que tener que ingresar `exit`. `Ctrl+c` interrumpirá el comando en curso de ejecución, excepto si se encuentra en el proceso de edición, en cuyo caso interrumpirá la edición y lo devolverá al *prompt*. `Ctrl+l` borra la pantalla.

Finalmente, están `Ctrl+s` y `Ctrl+q`: estas secuencias de teclas sirven, respectivamente, para suspender y reanudar el flujo de caracteres sobre una terminal. No son muy usadas, pero sin embargo, puede ocurrir que ingrese `Ctrl+s` por error (después de todo, **s** y **d** están muy cerca una de la otra en el teclado...). Entonces, si presiona las teclas pero no ve aparecer carácter alguno en la terminal, primero intente `Ctrl+q` y preste atención: aparecerán en la pantalla todos los caracteres juntos que ingresó entre el `Ctrl+s` no deseado y `Ctrl+q`.

4. `Backspace` es la última tecla de la fila que contiene las teclas de los números.

Capítulo 3. Introducción a la línea de comandos

En el capítulo Conceptos básicos de Unix del **Guía del Usuario**, le hemos mostrado como iniciar un *shell*. En este capítulo, le mostraremos como ponerlo a trabajar.

La ventaja principal del *shell* es el número de utilitarios existentes: hay miles de ellos, y cada uno está dedicado a una tarea en particular. Sólo veremos una cantidad (pequeña) de ellos aquí. Una de las ventajas principales de *Unix* es la capacidad de combinar estos utilitarios, como veremos más adelante.

3.1. Utilitarios de manipulación de archivos

En este contexto, la manipulación de archivos significa copiar, mover y borrar archivos. Más adelante, veremos formas de cambiar los atributos de los mismos (dueño, permisos asociados).

3.1.1. mkdir, touch (tocar): creación de directorios y archivos vacíos

mkdir (*MaKe DIRectory*, crear directorio) se usa para crear directorios. Su sintaxis es simple:

```
mkdir [opciones] <directorio> [directorio ...]
```

Sólo una opción es digna de interés: la opción -p. La misma hace dos cosas:

1. creará los directorios padre si es que aún no existían. Si no se especifica esta opción y los directorios padre no existen, mkdir simplemente fallará, quejándose que dichos directorios padre no existen;
2. retornará silenciosamente si el directorio que Ud. desea crear ya existe. Similarmente, si no especificó la opción -p, mkdir fallará, quejándose que el directorio ya existe.

Aquí tiene algunos ejemplos:

1. mkdir pepe crea un directorio denominado pepe en el directorio corriente;
2. mkdir -p images/misc docs crea un directorio misc en el directorio images creando primero el último si es que no existe (-p); también crea un directorio denominado docs en el directorio corriente.

Inicialmente, el comando touch no está orientado a la creación de archivos sino a la actualización de la hora de acceso y modificación de los archivos[1]. Sin embargo, touch creará los archivos mencionados como archivos de 0 bytes si es que no existían. La sintaxis es:

```
touch [opciones] archivo [archivo...]
```

Entonces, ejecutar el comando:

```
touch archivo1 images/archivo2
```

creará un archivo de 0 bytes denominado archivo1 en el directorio corriente y un archivo de 0 bytes denominado archivo2 en el directorio images.

1. Hay tres etiquetas de tiempo distintas para cada archivo en *Unix*: la última fecha de acceso al mismo (atime), es decir, la fecha cuando se abrió para lectura o escritura; la última fecha cuando se modificaron los atributos del i-nodo (mtime); y finalmente, la última fecha cuando se modificó el **contenido** del archivo (ctime).

3.1.2. rm : borrar archivos o directorios

El comando `rm` (*ReMove*, quitar) reemplaza a los comandos `del` y `deltree` de *DOS*, y agrega más opciones. Su sintaxis es la siguiente:

```
rm [opciones] <archivo|directorio> [archivo|directorio...]
```

Entre las opciones, se encuentran:

- `-r`, o `-R`: borrar recursivamente. Esta opción es **obligatoria** para borrar un directorio, vacío o no. Sin embargo, también puede usar el comando `rmdir` para borrar directorios vacíos.
- `-i`: pedir confirmación antes de cada supresión. Note que predeterminadamente en **Linux-Mandrake**, `rm` es un ***alias*** a `rm -i`, por razones de seguridad (existen alias similares para los comandos `cp` y `mv`). La utilidad de estos alias puede variar para Ud. según su experiencia. Si desea quitarlos, puede editar su `.bashrc` y agregar esta línea: `unalias rm cp mv`.
- `-f`: la opuesta de `-i`, fuerza la supresión de los archivos o directorios, incluso si el usuario no tiene derecho de escritura sobre los archivos[2].

Algunos ejemplos:

- `rm -i images/*.jpg archivo1`: borra todos los archivos cuyo nombre termina en `.jpg` en el directorio `images` y borra el archivo `archivo1` en el directorio corriente, pidiendo confirmación para cada uno de los archivos. Responda **y** para confirmar la supresión, **n** para cancelarla.
- `rm -Rf images/misc/ archivo*`: borra todo el directorio `misc/` del directorio `images/` junto con todos los archivos del directorio corriente cuyos nombres comiencen con `archivo` sin pedir confirmación alguna.

> **Aviso**
>
> un archivo borrado utilizando `rm` se borra **irrevocablemente**. ¡No hay forma alguna de recuperar los archivos! No dude en usar la opción `-i` para asegurarse de no borrar algo por error.

3.1.3. mv : mover o renombrar archivos

La sintaxis del comando `mv` (*MoVe*, mover) es la siguiente:

```
mv [opciones] <archivo|dir.> [archivo|dir. ...] <destino>
```

Algunas opciones:

- `-f` Fuerza el movimiento de archivos – no hay advertencia alguna en caso de que la operación sobre-escriba un archivo que ya existe.
- `-i` La opción contraria – pedir confirmación al usuario antes de sobre-escribir un archivo existente.
- `-v` ***Modo verboso***, reportar todos los cambios y la actividad.

2. Es suficiente que un usuario no privilegiado tenga derecho de escritura sobre un **directorio** para que pueda borrar los archivos que se encuentran en el mismo, incluso si dicho usuario no es el dueño de los archivos.

Algunos ejemplos:

- `mv -i /tmp/pics/*.gif .` Mover todos los archivos del directorio `/tmp/pics/` cuyos nombres terminan en `.gif` al directorio corriente (`.`), pidiendo confirmación antes de sobre-escribir cualquier archivo.
- `mv pepe pupu` Cambiar el nombre del archivo `pepe` por `pupu`.
- `mv -vf archivo* images/ tacho/` Mover, sin pedir confirmación, todos los archivos del directorio corriente cuyos nombres comiencen con `archivo` junto con todo el directorio `images/` al directorio `tacho/`, y mostrar cada operación llevada a cabo.

3.1.4. cp : copiar archivos y directorios

`cp` (*CoPy*, copiar) reemplaza a los comandos `copy`, `xcopy` de *DOS*, y agrega más opciones. Su sintaxis es la siguiente:

```
cp [opciones] <archivo|dir.> [archivo|dir. ...] <destino>
```

`cp` tiene un montón de opciones. Estas son las más comunes:

- `-R`: copiar recursivamente; **obligatoria** para copiar un directorio, incluso uno vacío.
- `-i`: pedir confirmación antes de sobre-escribir cualquier archivo que podría sobre-escribirse.
- `-f`: lo opuesto de `-i`, reemplazar cualquier archivo existente sin pedir confirmación alguna.
- `-v`: modo “verboso”, reporta todas las acciones ejecutadas por `cp`.

Algunos ejemplos:

- `cp -i /tmp/images/* images/`: copia todos los archivos del directorio `/tmp/images` al directorio `images/` del directorio corriente, pidiendo confirmación si se va a sobre-escribir un archivo.
- `cp -vR docs/ /shared/mp3s/* miscosas/`: copia todo el directorio `docs` al directorio actual más todos los archivos del directorio `/shared/mp3s` al directorio `miscosas` ubicado en el directorio corriente.
- `cp pepe pupu`: hace una copia del archivo `pepe` bajo el nombre `pupu` en el directorio corriente.

3.2. Manipulación de los atributos de los archivos

La serie de comandos que se presentan aquí se usan para cambiar el dueño o el grupo propietario de un archivo o sus permisos. En el capítulo `Conceptos básicos de Unix` del **Guía del Usuario** vimos los diferentes permisos.

3.2.1. chown, chgrp : cambiar el dueño y el grupo propietario de uno o más archivos

La sintaxis del comando `chown` (*CHange OWNer*, cambiar el dueño) es la siguiente:

```
chown [opciones] <usuario[.grupo]> <archivo|dir.> [archivo|dir. ...]
```

Las opciones incluyen:

- `-R`: “Recursivo”; para cambiar el dueño de todos los archivos y sub-directorios en un directorio dado.

- `-v` Modo “verboso”; muestra todas las acciones efectuadas por `chown`; reporta cuales archivos cambiaron de dueño como resultado del comando y cuales no han cambiado.
- `-c` Como `-v`, pero sólo reporta cuales archivos cambiaron.

Algunos ejemplos:

- `chown nobody /shared/book.tex` cambiar el dueño del archivo `/shared/book.tex` a `nobody`.
- `chown -Rc amidala.musica *.mid conciertos/`: atribuye todos los archivos en el directorio actual cuyos nombres terminan con `.mid` y todos los archivos y sub-directorios del directorio `conciertos/` al usuario `amidala` y al grupo `musica`, reportando sólo los archivos afectados por el comando.

El comando `chgrp` (*CHange GRouP*, cambiar el grupo) le permite cambiar el grupo propietario de un archivo o un grupo de archivos; su sintaxis es muy similar a la del comando `chown`:

```
chgrp [opciones] <grupo> <archivo|dir.> [archivo|dir. ...]
```

Las opciones de este comando son las mismas que las de prog-chown;, y se usa de manera muy similar. Por lo tanto, el comando:

```
chgrp disk /dev/hd*
```

le atribuye al grupo `disk` todos los archivos en el directorio `/dev/` cuyos nombres comiencen con `hd`.

3.2.2. chmod : cambiar los permisos sobre los archivos y directorios

El comando `chmod` (*CHange MODe*, cambiar el modo) tiene una sintaxis bien particular. La sintaxis general es:

```
chmod [opciones] <cambio de modo> <archivo|dir.> [archivo|dir. ...]
```

pero lo que lo distingue son las diferentes formas que puede tomar el cambio de modo. Este se puede especificar de dos maneras:

1. en octal; entonces los derechos del usuario dueño se corresponden con números de la forma `<x>00`, donde `<x>` corresponde al permiso asignado: 4 para lectura, 2 para escritura, y 1 para ejecución; similarmente, los derechos del grupo propietario toman la forma `<x>0` y los derechos de los “otros” la forma `<x>`. Por lo tanto, todo lo que Ud. necesita hacer es sumar los derechos asignados para obtener el número correcto. Por lo tanto, los derechos `rwxr-xr--` corresponden a 400+200+100 (derechos del dueño, `rwx`) +40+10 (derechos del grupo propietario, `r-x`) +4 (derechos de los otros, `r--`) = 754; de esta forma, los derechos se expresan en términos absolutos. Esto significa que los derechos previos se reemplazan incondicionalmente;
2. con expresiones: aquí los derechos se expresan con una secuencia de expresiones separadas por comas, donde una expresión toma la forma `[categoría]<+|-><permisos>`.

   La categoría puede ser una o más de:

   - `u` (*User*, Usuario, permisos para el dueño),
   - `g` (*Group*, Grupo, permisos para el grupo propietario) u
   - `o` (*Others*, Otros, permisos para los “otros”).

   Si no se especifica categoría alguna, los cambios se aplican para todas las categorías. Un `+` garantiza un permiso y un `-` lo niega y un `=` lo garantiza. Finalmente, el permiso es uno o más de:

   - `r` (*Read*, lectura),

- `w` (*Write*, escritura) o
- `x` (*eXecute*, ejecución).

Las opciones principales son bastante similares a las de `chown` o `chgrp`:

- `-R`: cambiar los permisos recursivamente.
- `-v`: modo "verboso", muestra las acciones efectuadas para cada archivo.
- `-c`: como `-v` pero solo muestra los archivos afectados por el comando.

Ejemplos:

- `chmod -R o-w /shared/docs`: quitar, recursivamente, el permiso de escritura para los "otros" sobre todos los archivos y sub-directorios de `/shared/docs/`.
- `chmod -R og-w,o-x privado/` Quitar, "recursivamente", el permiso de escritura para el grupo propietario y para los "otros" sobre todo el directorio `privado/`, y quitar el permiso de ejecución para los "otros".
- `chmod -c 644 varios/archivo*` cambia los permisos de todos los archivos del directorio `varios/` cuyos nombres comiencen con `archivo` a `rw-r--r--` (es decir, permiso de lectura para todos y permiso de escritura solo para el dueño), y reporta solo los archivos afectados por la operación.

3.3. Patrones de englobamiento del `shell`

Probablemente Ud. ya usa caracteres de ***englobamiento*** sin saberlo. Cuando Ud. especifica un archivo en *Windows* o cuando Ud. busca un archivo, Ud. usa `*` para hacer coincidir con una cadena aleatoria de caracteres. Por ejemplo, `*.txt` hace coincidir a todos los archivos cuyo nombre termina con `.txt`. Nosotros también los usamos mucho en la última sección. Pero el englobamiento va más allá que el simple `*`.

Cuando Ud. ingresa un comando como `ls *.txt` y presiona `Intro`, la tarea de encontrar cuales archivos se corresponden con el patrón `*.txt` no la realiza el comando `ls`, sino el *shell* en sí mismo. Esto requiere de una pequeña explicación sobre como interpreta el *shell* la línea de comandos. Cuando Ud. ingresa:

```
$ ls *.txt
leerme.txt  recetas.txt
```

primero la línea de comandos se separa en palabras (`ls` y `*.txt`, en este ejemplo) . Cuando el *shell* ve un `*` en una palabra, interpretará toda la palabra como un patrón de englobamiento y la reemplazará con los nombres de todos los archivos que se correspondan con el patrón. Por lo tanto, justo antes que el *shell* la ejecute, la línea se convirtió en la línea `ls leerme.txt recetas.txt`, lo que da el resultado esperado. El *shell* reacciona así frente a otros caracteres como:

- `?` se corresponde con un único caracter (uno y sólo uno), cualquiera que sea este;
- `[...]` se corresponde con cualquiera de los caracteres que se encuentran entre los corchetes; los caracteres pueden estar referidos por intervalos (por ejemplo, `1-9`) o por ***valores discretos***,o una mezcla de ambos. Ejemplo: `[a-zBE5-7]` se corresponderá con todos los caracteres desde la `a` hasta la `z`, una `B`, una `E`, un `5`, un `6` o un `7`;
- `[!...]` se corresponde a cualquier caracter que **no** se encuentre entre los corchetes. `[!a-z]`, por ejemplo, se corresponderá con cualquier caracter que no sea una letra minúscula[3];

- `{c1,c2}` se corresponde con `c1` o con `c2`, donde `c1` y `c2` también son patrones de englobamiento, lo cual significa que Ud. puede escribir `{[0-9]*,[acr]}` por ejemplo.

Aquí tiene algunos patrones y su significado:

- `/etc/*conf`: todos los archivos del directorio `/etc` cuyo nombre termine con `conf`. Se puede corresponder con `/etc/inetd.conf`, pero también con `/etc/conf.linuxconf` y también con `/etc/conf` si existe tal archivo: recuerde que `*` puede corresponderse con una cadena vacía.
- `image/autos,espacio[0-9]/*.jpg`: todos los archivos cuyo nombre termina en `.jpg` en los directorios `image/autos`, `image/espacio0`, ... , `image/espacio9`, si es que dichos directorios existen.
- `/usr/doc/*/LEAME`: todos los archivos denominados `LEAME` en todos los sub-directorios inmediatos del directorio `/usr/doc`. Esto hará que `/usr/share/doc/mandrake/LEAME` corresponda, pero no `/usr/share/doc/miprog/doc/LEAME`.
- `*[!a-z]` Todos los archivos en el directorio corriente cuyo nombre **no** termine con una letra minúscula.

3.4. Redirecciones y tuberías

3.4.1. Un poco más sobre los procesos

Para entender el principio de las redirecciones y las tuberías, necesitamos explicar una noción acerca de los procesos que todavía no ha sido introducida. Cada proceso *Unix* (esto también incluye a las aplicaciones gráficas) abre un mínimo de tres descriptores de archivo: la ***entrada estándar***, la ***salida estándar***, y el ***error estándar***. Sus números respectivos son 0, 1 y 2. En general, estos tres descriptores están asociados con la terminal desde la cual se inició el proceso, siendo el teclado la entrada. El objetivo de las redirecciones y las tuberías es redirigir estos descriptores. Los ejemplos en esta sección lo ayudarán a comprender mejor este concepto.

3.4.2. Redirecciones

Suponga, por ejemplo, que Ud. quiere una lista de los archivos que terminan en `.png`[4] en el directorio `images`. Esta lista es muy larga, por lo que Ud. quiere almacenarla en un archivo para consultarla a gusto después. Ud. puede ingresar el comando siguiente:

```
$ ls images/*.png 1>lista_de_archivos
```

Esto significa que la salida estándar de este comando (`1`) se redirecciona (`>`) al archivo denominado `lista_de_archivos`. El operador `>` es el operador de redirección de la salida. Si el archivo de redirección no existe, se crea, pero si existe se sobre-escribe su contenido. Sin embargo, el descriptor predeterminado que redirecciona

3. ¡Cuidado! Aunque esto es cierto bajo *GNU/Linux*, puede que no lo sea bajo otros sistemas operativos tipo *Unix*. Esta característica depende del **orden de comparación**. En algunos sistemas, `[a-z]` se corresponderá con a, A, b, B, ... , z. Y ni siquiera mencionamos el hecho que algunos idiomas tienen caracteres acentuados...

4. Ud. podría creer que decir “los archivos que terminan en `.png`” en vez de “las imágenes PNG” es una locura. Sin embargo, una vez más, los archivos bajo *Unix* sólo tienen una extensión por convención: de ninguna manera las extensiones definen al tipo de archivo. Un archivo que termina en `.png` podría ser perfectamente una imagen JPEG, una aplicación, un archivo de texto o cualquier otro tipo de archivo. ¡Lo mismo es cierto también bajo *Windows*!

este operador es la salida estándar y no es necesario especificarla en la línea de comandos. Ud. podría haber escrito simplemente:

```
$ ls images/*.png >lista_de_archivos
```

y el resultado será exactamente el mismo. Luego, Ud. puede mirar el archivo usando un visualizador de archivos de texto tal como `less`.

Imagine ahora que Ud. quiere saber cuantos de estos archivos hay. En vez de contarlos a mano, Ud. puede usar el utilitario denominado `wc` (*Word Count*, contador de palabras) con la opción `-l`, que escribe en la salida estándar el número de líneas en el archivo. Una solución es la siguiente:

```
wc -l 0<lista_de_archivos
```

y esto da el resultado deseado. El operador `<` es el operador de redirección de la entrada, y similarmente el descriptor redirigido predeterminadamente es el de la entrada estándar, es decir, `0`, y Ud. simplemente tiene que escribir la línea:

```
wc -l <lista_de_archivos
```

Suponga ahora que Ud. quiere consultar esta lista, quitar todas las “extensiones” de los archivos y poner el resultado en otro archivo. Una herramienta para hacer esto es `sed`, por *Stream EDitor* (editor de flujo). Ud. simplemente redirecciona la entrada estándar del comando `sed` al archivo `lista_de_archivos` y redirecciona su salida al archivo resultado, por ejemplo `la_lista`:

```
sed -e 's/\.png$//g' <lista_de_archivos >la_lista
```

y aquí tiene creada su lista, disponible para ser consultada a gusto con cualquier visualizador.

También puede ser útil redirigir el error estándar. Por ejemplo, Ud. quiere saber a cuales directorios de `/shared` no tiene acceso: una solución es listar este directorio recursivamente y redirigir los errores a un archivo, mientras no se muestran por el canal de salida estándar:

```
ls -R /shared >/dev/null 2>errores
```

lo que significa que se redireccionará la salida estándar (`>`) a `/dev/null`, un archivo especial donde todo lo que escribe se pierde (es decir que, como efecto secundario, no se muestra la salida estándar) y el canal de error estándar (`2`) se redirecciona (`>`) al archivo `errores`.

3.4.3. Tuberías

Las tuberías (*pipes*, en inglés) son de alguna forma, una combinación de redirecciones de la entrada y la salida. Su principio es el de un tubo físico, de aquí el nombre: un proceso envía datos por un extremo del tubo y otro proceso lee los datos en el otro extremo. El operador de la tubería es `|`. Volvamos al ejemplo anterior de la lista de archivos. Suponga que Ud. quiere encontrar directamente cuantos archivos hay sin tener que almacenar la lista en un archivo temporal, entonces Ud. usaría el comando siguiente:

```
ls images/*.gif | wc -l
```

lo cual significa que la salida estándar del comando `ls` (es decir, la lista de archivos) se redirecciona a la entrada estándar del comando `wc`. Así, Ud. obtiene el resultado deseado.

Ud. también puede construir directamente una lista de archivos "sin las extensiones" usando el comando siguiente:

```
ls images/*.png | sed -e 's/\.png$//g' >la_lista
```

o, si quiere consultar la lista directamente sin almacenarla en un archivo:

```
ls images/*.png | sed -e 's/\.png$//g' | less
```

Las tuberías y las redirecciones no están limitadas solamente a textos que pueden ser leídos por seres humanos. Por ejemplo, el comando siguiente enviado desde una terminal:

```
xwd -root | convert - ~/mi_escritorio.png
```

hará una captura de pantalla de su escritorio y la almacenará en el archivo `mi_escritorio.png`[5] en su directorio personal.

3.5. El completado de la línea de comandos

El "***completado***" es una funcionalidad muy útil, y todos los *shell*s modernos (*Bash*, inclusive) la tienen. Su rol es darle al usuario el menor trabajo posible. La mejor manera de ilustrarlo es con un ejemplo.

3.5.1. Ejemplo

Suponga que su directorio personal contiene un archivo cuyo nombre es `archivo_con_un_nombre_muy_largo`, y Ud. quiere mirarlo. Suponga que Ud. también tiene en el mismo directorio otro archivo denominado `archivo_texto`. Ud. está en su directorio personal. Así que Ud. ingresa la secuencia siguiente:

```
$ less ar<TAB>
```

(es decir, ingresa `less ar` y luego presiona la tecla `TAB`). El *shell* ahora extenderá la línea de comandos por Ud.:

```
$ less archivo_
```

5. Sí, de hecho, será una imagen PNG **:-)** (Siempre y cuando tenga instalado el paquete "`ImageMagick`" ...)

y también le dará la lista de elecciones posibles (en su configuración predeterminada, que se puede personalizar). Luego ingrese la siguiente secuencia de teclas:

```
less archivo_c<TAB>
```

y el *shell* extenderá la línea de comandos para darle el resultado que Ud. quiere:

```
less archivo_con_un_nombre_muy_largo
```

Entonces, todo lo que necesita hacer es presionar la tecla `Intro` para confirmar y leer el archivo.

3.5.2. Otros métodos de completado

La tecla `TAB` no es la única manera de activar el completado, aunque es la más común. Como regla general, la palabra a completar será el nombre de un comando para la primera palabra de la línea de comandos (`nsl<TAB>` dará `nslookup`), y el nombre de un archivo para todos los demás parámetros, a menos que la palabra esté precedida por un caracter "mágico" como `~`, `@` o `$`, en cuyo caso el *shell* intentará completar, respectivamente, un nombre de usuario, una máquina o una variable de entorno[6]. También hay un caracter mágico para completar el nombre de un comando (`!`) o el nombre de un archivo (`/`).

Las otras dos formas de activar el completado son las secuencias `Esc-<x>` y `Ctrl+x <x>`, donde `<x>` es uno de los caracteres mágicos ya mencionados. `Esc-<x>` intentará el completado de manera única, y si falla completará la palabra con la sub-cadena más larga posible de la lista de opciones. Un ***bip*** significa que la opción no es única o que no hay opción correspondiente. La secuencia `Ctrl+x <x>` muestra la lista de opciones posibles sin intentar completado alguno. Presionar la tecla `TAB` es lo mismo que presionar sucesivamente `Esc-<x>` y `Ctrl+x <x>`, donde el caracter mágico depende del contexto.

Por lo tanto, una forma de ver todas las variables de entorno definidas es ingresar la secuencia `Ctrl+x $` en una línea en blanco. Otro ejemplo: si Ud. quiere ver la página `man` del comando `nslookup`, simplemente ingresa `man nsl` luego `Esc-!`, y el *shell* completará automáticamente como `man nslookup`.

3.6. Inicio y manipulación de procesos en segundo plano: el `job control`

Ud. debe haber notado que cuando ingresa un comando desde una terminal, normalmente tiene que esperar a que el comando termine antes que el *shell* le devuelva el control. Esto significa que Ud. envió el comando en ***primer plano***. Sin embargo, hay ocasiones donde esto no es deseable.

Suponga, por ejemplo, que Ud. decidió copiar recursivamente un directorio grande a otro. Ud. también decidió ignorar los errores, por lo que redirecciona el canal de error a `/dev/null`:

```
cp -R images/ /shared/ 2>/dev/null
```

6. Recuerde: *Unix* diferencia entre mayúsculas y minúsculas. La variable de entorno `HOME` y la variable de entorno `home` no son la misma variable.

Un comando como este puede tardar varios minutos para terminar. Entonces, Ud. tiene dos soluciones: la primera es violenta y significa terminar el comando y volver a hacerlo más tarde cuando tenga el tiempo. Para hacer esto, ingrese `Ctrl+c`: esto le devolverá el *prompt*. Pero espere, ¡no lo haga! Siga leyendo.

Suponga que Ud. quiere ejecutar el comando mientras hace otra cosa al mismo tiempo. Entonces, la solución es poner al proceso en ***segundo plano***. Para hacer esto, ingrese `Ctrl+z` para suspender al proceso:

```
$ cp -R images/ /shared/ 2>/dev/null
  # Presione C-z
[1]+  Stopped                 cp -R images/ /shared/ 2>/dev/null
$
```

y aquí está, de nuevo en el *prompt*. El proceso está entonces suspendido, esperando que Ud. lo vuelva a iniciar (como muestra la palabra clave `Stopped`, detenido). Eso, por supuesto, es lo que Ud. quiere hacer, pero en segundo plano. Ingrese `bg` (por *BackGround*, segundo plano) para obtener el resultado deseado:

```
$ bg
[1]+ cp -R images/ /shared/ 2>/dev/null &
$
```

Entonces, el proceso comenzará a ejecutar nuevamente como una tarea en segundo plano, como lo indica el signo `&` (ampersand) al final de la línea. Ud. volverá al *prompt* y podrá continuar trabajando. Un proceso que corre como tarea en el fondo, o en segundo plano, se denomina `job`.

Por supuesto, Ud. puede iniciar procesos directamente como tareas en segundo plano, precisamente agregando un caracter **&** al final del comando. Por ejemplo, Ud. puede iniciar el comando para copiar el directorio en segundo plano escribiendo:

```
cp -R images/ /shared/ 2>/dev/null &
```

Si Ud. lo desea, también puede volver este proceso a un primer plano y esperar a que termine ingresando `fg` (*ForeGround*, primer plano). Para volverlo al segundo plano, ingrese la secuencia `Ctrl+z`, `bg`.

Ud. puede iniciar varios *jobs* de esta forma: entonces, se asignará un número de *job* a cada comando. El comando `jobs` del *shell* lista todos los *jobs* asociados al *shell* corriente. El *job* precedido por un signo `+` indica el último proceso iniciado como tarea de segundo plano. Para pasar a un *job* en particular al primer plano, Ud. puede ingresar `fg <n>` donde `<n>` es el número de *job*, por ejemplo, `fg 5`.

Note que Ud. también puede suspender o lanzar aplicaciones de ***pantalla completa*** (si es que están programadas correctamente) de esta forma, tales como `less` o un editor de texto como *VI*, y pasarlos al primer plano cuando Ud. quiera.

3.7. Una palabra final

Como puede ver, el *shell* es muy completo y usarlo efectivamente sólo es cuestión de práctica. En este capítulo relativamente largo, sólo hemos mencionado algunos de los comandos disponibles: **Linux-Mandrake** tiene miles de utilitarios, e incluso los usuarios más experimentados no los usan a todos.

Hay herramientas para todos los gustos y propósitos: Ud. puede manipular imágenes (como `convert`, mencionado arriba, pero también, el modo ***por lotes*** de *GIMP* y todas las herramientas de manipulación de `pixmaps`), sonidos (codificadores MP3, reproductores de CD de audio), para la grabación de CD, programas de *correo-*

e, clientes FTP e incluso navegadores de *web* (como `Lynx` o `links`), sin olvidarnos de todas las herramientas de administración.

Incluso si existen aplicaciones gráficas con funcionalidad equivalente, generalmente son interfaces gráficas construidas sobre estos mismos utilitarios; además, los utilitarios de la línea de comandos tienen la ventaja de poder operar en modo no interactivo: Ud. puede comenzar a escribir un CD y luego "desconectarse" del sistema con la confianza que se efectuará la grabación (ver la página Man nohup(1)).

Capítulo 4. La edición de texto: Emacs y VI

Como se dijo en la introducción, la edición de texto[1] es una característica fundamental en el uso de un sistema *Unix*. Los dos editores que vamos a estudiar brevemente, al principio son un poco difíciles, pero una vez que entendió las bases, ambos resultan ser herramientas potentes.

4.1. Emacs

Emacs es probablemente el editor de texto más potente que existe. Puede hacer absolutamente de todo y es extensible infinitamente gracias a su lenguaje de programación incorporado, basado en *Lisp*. Con *Emacs*, Ud. puede moverse por la *web*, leer su correo, tomar parte en foros de discusión, hacer el café, y así sucesivamente. Pero lo que Ud. podrá hacer al final de esta sección estará limitado a abrir *Emacs*, editar uno o más archivos, guardarlos y salir de *Emacs*. Lo cual no es poco para empezar.

4.1.1. Presentación breve

Invocar a *Emacs* es relativamente simple:

```
emacs [archivo] [archivo...]
```

Emacs abrirá cada archivo ingresado como argumento en un *buffer* hasta el número máximo de dos *buffers* visibles a la vez, y le presentará el *buffer* `*scratch*` si no especifica archivo alguno. Si está en *X*, también tendrá disponible un menú, pero en este capítulo trabajaremos con el teclado.

4.1.2. Comenzando

Es tiempo de poner manos a la obra. A modo de ejemplo, abramos dos archivos, `archivo1` y `archivo2`. Si estos dos archivos no existen, serán creados tan pronto como Ud. escriba algo en ellos:

```
$ emacs archivo1 archivo2
```

Obtendrá la ventana que se muestra en la Figura 4-1.

---

1. “Editar texto” significa modificar el contenido de un archivo que principalmente contiene letras, dígitos, y signos de puntuación; tales archivos pueden ser mensajes electrónicos, código fuente de programas, documentos, o archivos de configuración.

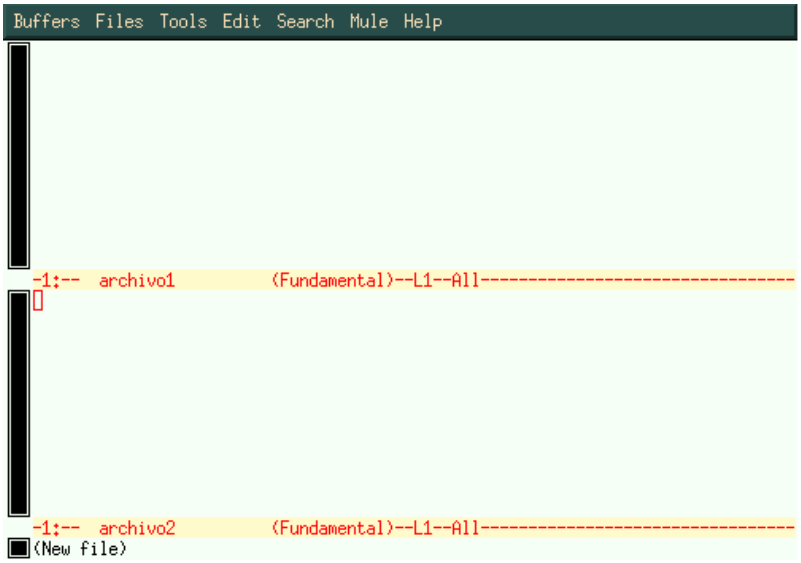

Figura 4-1. Emacs, editar dos archivos a la vez

Como puede ver, se crearon dos *buffers*: uno por archivo. También está presente un tercero en la parte inferior de la pantalla (donde Ud. ve (New file)); este es el mini-*buffer*. Ud. no puede ir, por las suyas, a este *buffer*; *Emacs* debe invitarlo durante las entradas interactivas. Para cambiar el *buffer* corriente ingrese Ctrl+x o. Puede ingresar texto como en un editor "normal", y borrar caracteres con la tecla Supr o la tecla Backspace.

Para desplazarse por ahí, puede usar las teclas de las flechas, así como también estas otras combinaciones de teclas: Ctrl+a para ir al principio de la línea, Alt+< para ir al principio del *buffer* y Alt+> para ir al final del mismo. Hay muchas otras combinaciones, incluso para cada una de las teclas de las flechas[2].

Tan pronto como quiera guardar los cambios hechos en un archivo, ingrese Ctrl+x Ctrl+s, o si Ud. quiere grabar el contenido del *buffer* en otro archivo, ingrese Ctrl+x Ctrl+w y *Emacs* le pedirá el nombre del archivo en el cual se escribirá el contenido del *buffer*. Puede usar el "***completado***" para hacer esto.

4.1.3. Manipulación de los buffers

Si quiere, Ud. puede mostrar un solo *buffer* en la pantalla. Hay dos formas de hacer esto:

- Ud. está en el *buffer* que quiere ocultar: ingrese Ctrl+x 0;
- Ud. está en el *buffer* que quiere conservar en la pantalla: ingrese Ctrl+x 1.

Por lo tanto, hay dos maneras de restaurar el *buffer* que desea en la pantalla:

- ingrese Ctrl+x b e introduzca el nombre del *buffer* que quiere,
- ingrese Ctrl+x Ctrl+b, entonces se abrirá un *buffer* nuevo, denominado *Buffer List*; Ud. se puede desplazar por este *buffer* usando la secuencia Ctrl+x o, luego seleccione el *buffer* que desea y presione la tecla Intro, o si no ingrese el nombre del *buffer* en el mini-*buffer*. El *buffer* *Buffer List* vuelve a segundo plano una vez que Ud. ha hecho su elección.

Si Ud. ha finalizado con un archivo y quiere deshacerse del *buffer* asociado, ingrese Ctrl+x k. Entonces *Emacs* le preguntará qué *buffer* debe cerrar. Predeterminadamente, es el nombre del *buffer* en el cual Ud.

2. *Emacs* ha sido diseñado para funcionar en una gran variedad de máquinas, algunas de las cuales incluso no tienen teclas de las flechas. Esto es incluso más cierto en *VI*.

se encuentra en ese momento; si Ud. quiere deshacerse de un *buffer* que no sea el propuesto, ingrese su nombre directamente o bien presione TAB: *Emacs* abrirá entonces otro *buffer* más denominado *Completions* dando la lista de elecciones posibles. La tecla Intro valida la elección.

Ud. también puede restaurar dos *buffers* visibles en la pantalla al mismo tiempo; para hacer esto ingrese Ctrl+x 2. Predeterminadamente, el nuevo *buffer* creado será una copia del *buffer* corriente (lo que le permite, por ejemplo, editar un archivo grande en varios lugares "a la vez"), y Ud. simplemente procederá como se describió anteriormente para moverse entre los *buffers*.

Ud. puede abrir otros archivos en cualquier momento, usando Ctrl+x Ctrl+f. Entonces *Emacs* le pedirá el nombre del archivo y Ud. dispondrá otra vez del completado.

4.1.4. Copiar, cortar, pegar, buscar

Suponga que estamos en la situación de la Figura 4-2.

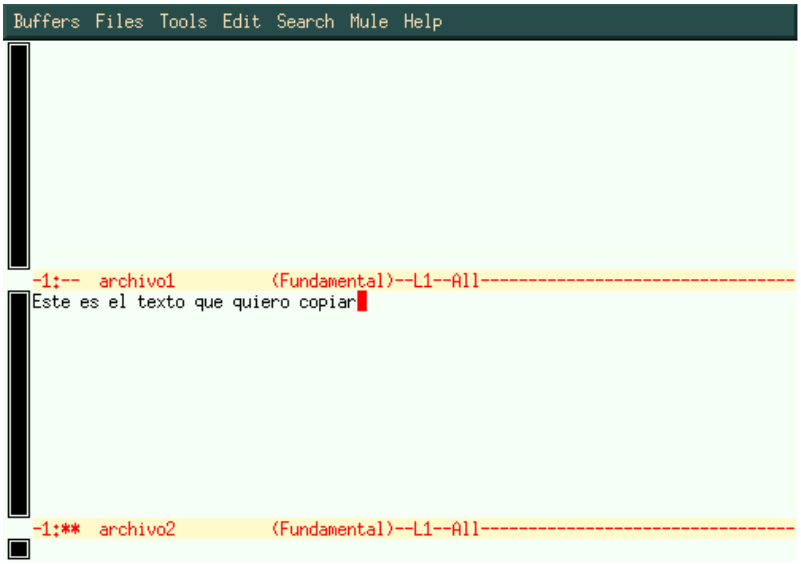

Figura 4-2. Emacs, antes de copiar el bloque de texto

Primero, necesitará seleccionar el texto que Ud. quiere copiar. En *X*, Ud. lo puede hacer usando el ratón, e incluso se resaltará el área seleccionada. Pero aquí estamos en modo texto **:-)** En este caso, queremos copiar toda la oración. Primero, necesitará poner una marca para indicar el comienzo del área. Asumiendo que el cursor está en la posición donde se encuentra en la figura anterior, primero ingrese Ctrl+ESPACIO (Control + barra espaciadora); *Emacs* entonces mostrará el mensaje Mark set en el mini-*buffer*. Luego muévase al principio de la línea con Ctrl+a: el área seleccionada para copiar o cortar es toda el área que se encuentra entre la marca y la posición corriente del cursor, entonces en el caso presente será toda la línea. Luego, ingrese Alt+w (para copiar) o Ctrl+w (para cortar). Si Ud. copia, *Emacs* volverá brevemente a la posición de la marca, para que Ud. pueda ver el área seleccionada.

Luego vaya al *buffer* sobre el cual quiere copiar el texto, e ingrese Ctrl+y, para obtener lo que se muestra en la Figura 4-3.

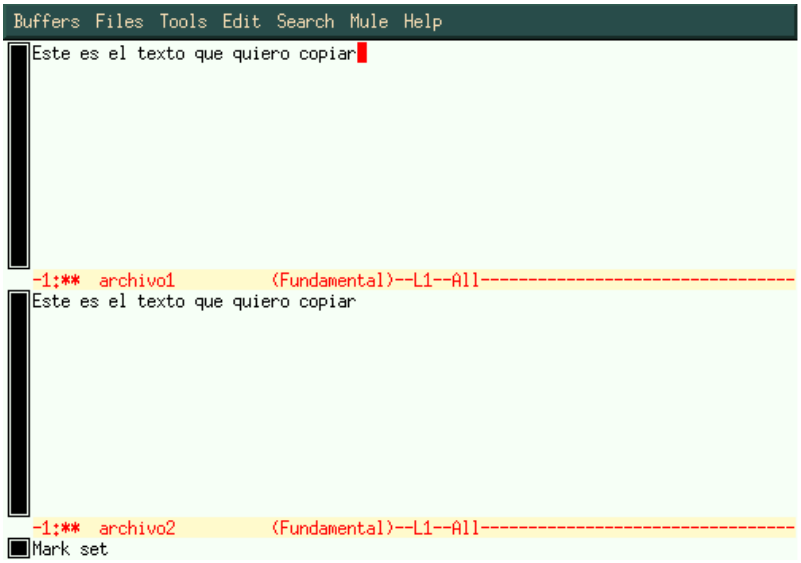

Figura 4-3. Emacs, después de la copia del bloque de texto

De hecho, lo que Ud. acaba de hacer es copiar texto al ***"kill ring"*** (*anillo de los muertos*) de *Emacs*: este *kill ring* contiene todas las regiones copiadas o cortadas desde que se inició *Emacs*. **Cualquier** región copiada o cortada se pone al comienzo del *kill ring*. La secuencia Ctrl+y simplemente "pega" la región que está en el tope. Si Ud. quiere tener acceso a otras regiones, presione Ctrl+y ,y luego Alt+y hasta que Ud. obtiene el texto deseado.

Para buscar texto, vaya al *buffer* deseado e ingrese Ctrl+s. Entonces, *Emacs* le pedirá la cadena a buscar. Para comenzar una búsqueda nueva con la misma cadena, todavía en el *buffer* corriente, ingrese Ctrl+s de nuevo. Cuando *Emacs* llega al final del *buffer* y no encuentra más ocurrencias, Ud. puede ingresar Ctrl+s de nuevo para volver a iniciar la búsqueda desde el principio del *buffer*. Al presionar la tecla Intro se finaliza la búsqueda.

Para buscar y reemplazar, ingrese Alt+%. *Emacs* le pedirá la cadena a buscar, con qué reemplazarla, y le pide confirmación para cada ocurrencia que encuentra.

Una última cosa muy útil: Ctrl+x u deshace la operación previa. Ud. puede deshacer tantas operaciones como quiera.

4.1.5. Salir de Emacs

El atajo para esto es Ctrl+x Ctrl+c. Entonces, *Emacs* le pedirá si Ud. quiere grabar las modificaciones hechas a los *buffers* si todavía no los ha grabado.

4.2. VI: el ancestro

VI fue el primer editor de pantalla completa que existió. Ese es uno de los argumentos de los detractores de *Unix*, pero también uno de los argumentos principales de sus defensores: si bien es complicado de aprender, también es una herramienta extremadamente potente una vez que uno se acostumbra a usarlo. Con unos pocos tecleos, un usuario de *VI* puede mover montañas y, aparte de *Emacs*, pocos editores de texto pueden decir lo mismo.

La versión provista con **Linux-Mandrake** es de hecho, *VIM*, por *VI iMproved* (VI Mejorado), pero lo llamaremos *VI* a lo largo de este capítulo.

4.2.1. Modo de inserción, modo comando, modo `ex`...

Primero, necesitamos iniciar *VI*, lo cual se hace exactamente como con *Emacs*. Así que, volvamos a nuestros dos archivos e ingresemos:

```
$ vi archivo1 archivo2
```

En este punto, Ud. se encontrará frente a una ventana como la que se muestra en la Figura 4-4.

Figura 4-4. Situación inicial en `VIM`

Ahora Ud. está en ***modo comando*** frente al primer archivo abierto. En modo comando, Ud. no puede insertar texto en un archivo... Para hacer esto, Ud. debe pasar a ***modo inserción***, y por lo tanto debe ingresar uno de los comandos que le permiten hacerlo:

- **a** e **i**: para insertar texto antes y después del cursor, respectivamente (**A** e **I** insertan texto al final y al principio de la línea corriente);
- **o** y **O**: para insertar texto debajo y por encima, respectivamente, de la línea corriente.

En modo de inserción, Ud. verá aparecer la cadena `--INSERT--` en la base de la pantalla (de esta forma, Ud. sabrá en que modo se encuentra). Es en este, y sólo en este modo, que Ud. puede ingresar texto. Para volver al modo comando, presione la tecla `Esc`.

En modo de inserción, Ud. puede usar las teclas `Backspace` y `Supr` para borrar texto a medida que avanza. Para desplazarse por el texto, tanto en modo comando como en modo inserción, Ud. usa las teclas de las flechas. También hay otras combinaciones de teclas en modo comando, que veremos más adelante.

Ud. accede al modo `ex` presionando la tecla **:** en modo comando. Aparecerá en la base de la pantalla los mismos **:**, y allí se posicionará el cursor. Todo lo que Ud. ingrese subsecuentemente, seguido por la presión de la tecla `Intro`, *VI* lo considerará como un comando `ex`. Si Ud. borra el comando y los **:** que ingresó, volverá al modo comando y el cursor retornará a su posición original.

Para grabar los cambios hechos a un archivo, Ud. ingresa la secuencia `:w` en modo comando. Si Ud. quiere grabe el contenido del *buffer* en otro archivo, ingrese `:w <nombre_de_archivo>`

4.2.2. Manipulación de los `buffers`

Ud. puede tener varios *buffers* visibles en la pantalla a la vez, como con *Emacs*. Para esto, use el comando `:split`.

Para pasar de un archivo a otro, en un *buffer*, Ud. ingresa `:next` para moverse al archivo siguiente y `:prev` para moverse al archivo previo. Ud. puede usar `:e <nombre_de_archivo>` también, lo cual le permite tanto

cambiar al archivo deseado, si es que ya está abierto, como también abrir otro archivo. Aquí también Ud. dispone del "completado".

Para cambiar de *buffer*, ingrese `Ctrl+w j` para ir al *buffer* de abajo o `Ctrl+w k` para ir al *buffer* de arriba. Ud. también puede usar las teclas de las flechas para arriba y para abajo, en lugar de **k** o **j**, respectivamente. El comando `:close` oculta un *buffer*, el comando `:q` lo cierra.

Atención, *VI* es fastidioso: si Ud. intenta ocultar o cerrar un *buffer* sin guardar los cambios, el comando no se llevará a cabo y Ud. obtendrá este mensaje:

```
No write since last change (use! to override)
```

(*no se escribió desde el ultimo cambio (use ! para forzar el comando)*) En este caso, haga lo que se le dice e ingrese `:q!` o `:close!`.

4.2.3. Edición de texto y comandos de desplazamiento

Además de las teclas `Backspace` y `Supr` en modo de edición, *VI* tiene muchos otros comandos para borrar, copiar, pegar, y reemplazar texto – en modo comando. Aquí, veremos algunos. Todos los comandos que se muestran aquí están, de hecho, separados en dos partes: la acción a realizar y su efecto. La acción puede ser:

- **c**: para cambiar (*Change*) o reemplazar; el editor borra el texto que se pide y vuelve al modo de inserción después de este comando;
- **d**: para borrar (*Delete*);
- **y**: para copiar (*Yank*), lo veremos en la sección siguiente.
- **.**: repite la última acción.

El efecto define al grupo de caracteres sobre los cuales actúa el comando. Estos mismos comandos de efecto ingresados como están en modo comando corresponden a los desplazamientos:

- **h**, **j**, **k**, **l**: un caracter a la izquierda, abajo, arriba, a la derecha[3] respectivamente;
- **e**, **b**, **w**: hasta el final (respecto al comienzo) de la palabra corriente; del comienzo de la palabra siguiente;
- **^**, **0**, **$**: hasta el primer caracter no blanco de la línea corriente, hasta el comienzo de la línea corriente, hasta el final de la línea corriente;
- `f<x>`: hasta la próxima ocurrencia del caracter `<x>`; por ejemplo, `fe` desplaza el cursor hasta la próxima ocurrencia del caracter **e**;
- `/<cadena>`, `?<cadena>`: hasta la próxima ocurrencia de la cadena o expresión regular `<cadena>`, y lo mismo yendo hacia atrás en el archivo; por ejemplo, `/pepe` mueve el cursor hasta la próxima ocurrencia de la palabra `pepe`;
- **{**, **}**: hasta el comienzo, hasta el final, del párrafo corriente;
- **G**, **H**: hasta el final del archivo, hasta el comienzo de la pantalla.

Cada uno de estos caracteres de efecto o comandos de movimiento puede estar precedido por un número de repetición. **G** referencia al número de línea en el archivo. A partir de esto, Ud. puede hacer toda clase de combinaciones. Algunos ejemplos:

- `6b`: se mueve 6 palabras hacia atrás;
- `c8fk`: borrar todo el texto hasta la octava ocurrencia del caracter **k** y luego pasar a modo de inserción;

3. Un atajo para `dl` (borrar un caracter hacia adelante) es **x**; un atajo para `dh` es **X**; `dd` borra la línea corriente.

- `91G`: ir a la línea 91 del archivo;
- `d3$`: borra hasta el final de la línea corriente más las dos líneas siguientes.

Es cierto que estos comandos no son muy intuitivos, pero como siempre, el mejor método es la práctica. Aunque Ud. puede ver que la expresión “mover montañas con unas pocas teclas” no es tan exagerada **:-)**

4.2.4. Cortar, copiar, pegar

VI tiene un comando para copiar texto que ya hemos visto: el comando **y**. Para cortar texto, simplemente use el comando **d**. Ud. tiene 27 memorias para almacenar texto: una memoria anónima y 26 memorias que llevan el nombre de las 26 letras minúsculas del alfabeto inglés.

Para usar la memoria anónima Ud. ingresa el comando tal cual. Así, el comando `y12w` copia a la memoria anónima las 12 palabras que están después del cursor[4]. Si Ud. quiere cortar este área, use `d12w`.

Para usar una de las 26 memorias nombradas, ingrese la secuencia `"<x>` después del comando, donde `<x>` da el nombre de la memoria. Entonces, para copiar las mismas 12 palabras en la memoria **k**, Ud. puede ingresar `"ky12w`, y `"kd12w` para cortarlas.

Para pegar el contenido de la memoria anónima, Ud. usa los comandos **p** o **P** (por *Paste*, *Pegar*), para insertar texto después o antes del cursor, respectivamente. Para pegar el contenido de una memoria nombrada, use `"<x>p` o `"<x>P` de la misma forma (por ejemplo, `"dp` pegará el contenido de la memoria `d` después del cursor).

Veamos el ejemplo de la Figura 4-5.

Figura 4-5. `VIM`**, antes de copiar el bloque de texto**

Para efectuar esta acción, nosotros:

- volveremos a copiar las primeras 6 palabras de la oración en la memoria **r** (por ejemplo): `"ry6w`[5];
- pasaremos al buffer `file2`, que está ubicado abajo: `C-w j`;
- copiaremos el contenido de la memoria **r** antes del cursor: `"rp`.

El resultado, mostrado en la Figura 4-6, es el que esperábamos.

4. ... si el cursor está posicionado ¡al comienzo de la primer palabra!

5. `y6w` significa literalmente: “*Yank 6 words*”.

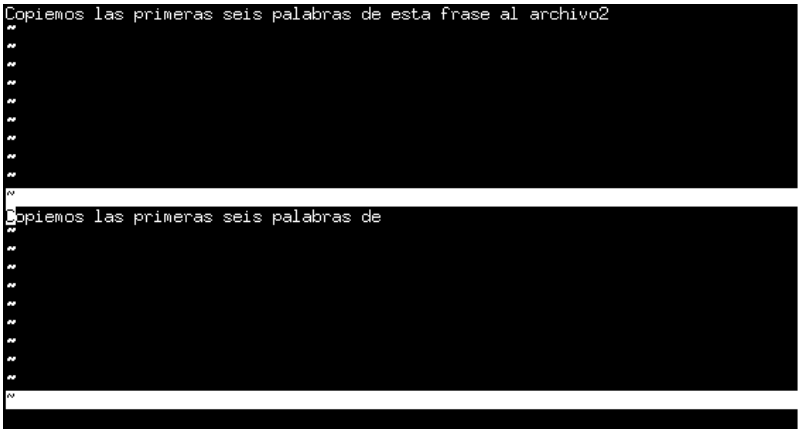

Figura 4-6. VIM, después de copiar un bloque de texto

La búsqueda de texto es muy simple: en modo comando, Ud. simplemente ingresa / seguida de la cadena a buscar, y luego presiona la tecla `Intro`. Por ejemplo, `/fiesta` buscará la cadena `fiesta` desde la posición corriente del cursor. Presionar **n** lo lleva a la próxima ocurrencia, y si Ud. llega al final del archivo, la búsqueda comenzará nuevamente por el principio. Para iniciar una búsqueda hacia atrás, use **?** en vez de /.

4.2.5. Salir de VI

El comando para salir, es `:q` (de hecho, este comando cierra el *buffer* activo, como hemos visto, pero si es el único *buffer* presente, Ud. sale de *VI*). Hay un atajo: la mayoría de las veces, Ud. edita solo un archivo. Entonces, para salir Ud. usará:

- `:wq` para guardar los cambios y salir (una solución más rápida es `ZZ`), o
- `:q!` para salir sin grabar.

Ud. habrá notado que si tiene varios *buffers*, `:wq` escribirá el *buffer* activo y luego lo cerrará.

4.3. Una última palabra...

Por supuesto, aquí hemos dicho mucho más de lo necesario (después de todo, el primer propósito era editar un archivo de texto), pero también es para mostrarle algunas de las posibilidades de cada uno de estos editores. Hay mucho más para decir de ellos, como lo prueba el número de libros dedicados a uno y al otro.

Tómese el tiempo para absorber toda esta información, elija por uno de ellos, o aprenda solo lo que Ud. crea necesario. Pero por lo menos Ud. sabe que cuando quiera ir más lejos, lo podrá hacer **:-)**

Capítulo 5. Los utilitarios de la línea de comandos

El propósito de este capítulo es introducir un número pequeño de herramientas de la línea de comandos que pueden resultar ser útiles para el uso diario. Por supuesto, puede saltar este capítulo si tiene la intención de usar sólo un entorno gráfico, pero un pequeño vistazo puede cambiar su opinión **:-)**

En realidad este capítulo no está organizado. Los utilitarios se indican sin un orden en particular, desde los más usados hasta los más complejos. Cada comando se ilustrará con un ejemplo, pero se le deja como ejercicio encontrar usos más útiles para ellos.

5.1. grep: General Regular Expression Parser

Analizador Sintáctico General de Expresiones Regulares

De acuerdo, el nombre no es muy intuitivo, tampoco su acrónimo, pero su uso es simple: buscar el patrón pasado como argumento en uno o más archivos. Su sintaxis es:

```
grep [opciones] <patrón> [uno o más archivos]
```

Si se mencionan varios archivos, su nombre aparecerá antes de las líneas que muestran los resultados que se corresponden con el criterio de búsqueda. Use la opción `-h` para ocultar estos nombres; use la opción `-l` para mostrar sólo los nombres de archivo en los cuales se cumple la condición de búsqueda. Puede ser útil, especialmente en listas de argumentos largas, recorrer los archivos con un bucle del *`shell`* y usar el truco siguiente: `grep <patrón> <nombre_de_archivo> /dev/null`. El patrón es una expresión regular, aunque generalmente consiste en una palabra simple. Las opciones usadas más frecuentemente son las siguientes:

- `-i`: Realizar una búsqueda que ignore la ***capitalización***. [1].
- `-v`: Búsqueda inversa: mostrar las líneas que **no** se corresponden con el patrón.
- `-n`: Mostrar, para cada línea encontrada, el número de línea.
- `-w`: Le dice a *`grep`* que el patrón debe corresponderse con una palabra completa, es decir debe aparecer tal cual y no como parte de otra palabra.

Aquí tiene un ejemplo de como usarlo:

```
$ cat mi_padre
Hola papá
Hola papi
Adiós papá

  # Buscar la cadena "hola", sin distinguir entre mayúsculas y minúsculas
$ grep -i hola mi_padre
Hola papá
Hola papi

  # Buscar "papá" como palabra completa, e imprimir el número de línea
  #   al principio de cada coincidencia
$ grep -nw papá mi_padre
1:Hola papá
3:Adiós papá
```

1. Distinción entre las mayúsculas y las minúsculas.

```
  # Queremos que coincidan todas las líneas que no comiencen con una "H"
$ grep -v "^H" mi_padre
Adiós papá
$
```

En caso que quiera usar `grep` en una tubería, no tiene que especificar el nombre del archivo ya que, predeterminadamente, toma su entrada desde la ***entrada estándar***. Similarmente, de manera predeterminada, imprime los resultados en la ***salida estándar***, por lo que puede meter la salida de un `grep` por una tubería a otro programa más sin miedo. Ejemplo:

```
$ cat /usr/share/doc/HOWTO/Parallel-Processing-HOWTO | \
  grep -n thread | less
```

5.2. find: busca archivos en función de ciertos criterios

`find` es un utilitario de *Unix* muy antiguo. Su rol es buscar recursivamente uno o más directorios y encontrar archivos que se correspondan con un cierto conjunto de criterios en esos directorios. Aunque es muy útil, su sintaxis es verdaderamente compleja, y usarlo requiere cierta práctica. La sintaxis general es:

```
find [opciones] [directorios] [criterios] [acción]
```

Si no especifica directorio alguno, `find` buscará en el directorio corriente. Si no especifica el criterio, esto es equivalente a "verdadero", por lo que se encontrarán todos los archivos. Las opciones, criterios y acciones son tan numerosas que solo mencionaremos algunas de cada una. Comencemos por las opciones:

- `-xdev`: No extender la búsqueda a los directorios ubicados en otros sistemas de archivos.
- `-mindepth <n>`: Descender al menos `<n>` niveles bajo el directorio especificado antes de comenzar a buscar los archivos.
- `-maxdepth <n>`: Buscar los archivos que se encuentran a lo sumo `n` niveles bajo el directorio especificado.
- `-follow`: Seguir los vínculos simbólicos si apuntan a directorios. Predeterminadamente, `find` no los sigue.
- `-daystart`: Cuando se usan las pruebas relativas a la fecha y la hora (ver debajo), toma el comienzo del día corriente como etiqueta temporal en vez del predeterminado (24 horas antes de la hora corriente).

Un criterio puede ser uno o más de varias pruebas ***atómicas***; algunas pruebas útiles son:

- `-type <tipo>`: Busca los archivos de un tipo dado; `<tipo>` puede ser uno de: `f` (archivo regular), `d` (directorio), `l` (vínculo simbólico), `s` (*socket*), `b` (archivo en modo de bloques), `c` (archivo en modo caracter) o `p` (tubería nombrada).
- `-name <patrón>`: Encontrar los archivos cuyo nombre se corresponde con el `<patrón>` dado. Con esta opción, se trata a `<patrón>` como un ***patrón de englobamiento*** del *shell* (vea el capítulo *Patrones de englobamiento del shell*, página 35).
- `-iname <patrón>`: Como `-name`, pero sin tener en cuenta la capitalización.
- `-atime <n>`, `-amin <n>`: Encontrar los archivos a los que se ha accedido en los últimos `<n>` días (`-atime`) o en los últimos `<n>` minutos (`-amin`). También puede especificar `+<n>` o `-<n>`, en cuyo caso la búsqueda se hará para los archivos accedidos respectivamente hace al menos o a lo sumo `<n>` días/minutos.
- `-anewer <archivo>`: Encontrar los archivos que han sido accedidos más recientemente que el archivo `<archivo>`
- `-ctime <n>`, `-cmin <n>`, `-cnewer <archivo>` Igual que para `-atime`, `-amin` y `-anewer`, pero se aplica a la última fecha en la cual se modificó el contenido del archivo.
- `-regex <patrón>`: Como para `-name`, pero `patrón` se trata como una ***expresión regular***.

- `-iregex <patrón>`: Como `-regex`, pero sin tener en cuenta la capitalización.

Existen muchas otras pruebas, debe referirse a la página Man para más detalles. Para combinar las pruebas, Ud. puede utilizar uno de:

- `<c1>` -a <c2>: Verdadero si tanto `<c1>` como `<c2>` son verdaderas; `-a` está implícito, por lo tanto puede ingresar `<c1>` <c2> <c3> ... si quiere que todas las pruebas `<c1>`, `<c2>`, ... sean verdaderas.
- `<c1>` -o <c2>: Verdadero si `<c1>` o `<c2>` o ambos son verdaderos. Note que `-o` tiene una ***precedencia*** menor que `-a`, por lo tanto si desea, por ejemplo, los archivos que verifican los criterios `<c1>` o `<c2>` y verifican el criterio `<c3>`, tendrá que usar paréntesis y escribir `( <c1> -o <c2> ) -a <c3>`. Debe ***escapar*** (desactivar) los paréntesis, ya que si no lo hace ¡el *shell* los interpretará!
- `-not <c1>`: Invertir la prueba `<c1>`, por lo tanto `-not <c1>` es verdadero si `<c1>` es falso.

Finalmente, puede especificar una acción para cada archivo encontrado. Las acciones más usadas frecuentemente son:

- `-print`: Simplemente imprime el nombre de cada archivo en la salida estándar. Esta es la acción predeterminada si Ud. no especifica acción alguna.
- `-ls`: Imprime en la salida estándar el equivalente de `ls -ilds` para cada archivo que encuentra.
- `-exec <comando>`: Ejecutar el comando `<comando>` sobre cada archivo encontrado. La línea de comandos `<comando>` debe terminar con un `;`, que deberá desactivar para que el *shell* no lo interprete; la posición del archivo se representa con `{}`. Vea los ejemplos de uso para entender mejor esto.
- `-ok <comando>`: Igual que `-exec` pero pedir confirmación para cada comando.

¿Todavía está aquí? Está bien, ahora practiquemos un poco, ya que todavía es la mejor forma de entender a este monstruo. Digamos que quiere encontrar todos los directorios en `/usr/share`. Entonces ingresará:

```
find /usr/share -type d
```

Suponga que tiene un servidor HTTP, todos sus archivos HTML están en `/var/www/html`, que coincide con su directorio corriente. Ud. desea encontrar todos los archivos que no se modificaron en el último mes. Debido a que tiene páginas de varios autores, algunos archivos tienen la extensión `html` y otros la extensión `htm`. Desea vincular estos archivos en el directorio `/var/www/obsolete`. Entonces ingresará[2]:

```
find \( -name "*.htm" -o -name "*.html" \) -a -ctime -30 \
-exec ln {} /var/www/obsolete \;
```

Está bien, este es uno un poco complejo y requiere una pequeña explicación. El criterio es este:

```
\( -name "*.htm" -o -name "*.html" \) -a -ctime -30
```

que hace lo que queremos: encuentra todos los archivos cuyos nombres terminan con `.htm` o con `.html` "`\( -name "*.htm"-o -name "*.html"\)`", **y** (`-a`) que no han sido modificados en los últimos 30 días, lo que es más o menos un mes (`-ctime -30`). Note los paréntesis: aquí son necesarios, porque `-a` tiene una precedencia mayor. Si no hubiera paréntesis alguno, se hubieran encontrado todos los archivos que terminen con `.htm`, y todos los archivos que terminen con `.html` y que no han sido modificados por un mes, que no es lo que nosotros queremos. Note también que los paréntesis están desactivados para el *shell*: si hubiésemos puesto

2. Note que este ejemplo necesita que `/var/www` y `/var/www/obsolete` ¡estén en el mismo sistema de archivos!

`( .. )` en vez de `\( .. \)`, el *shell* los hubiese interpretado y tratado de ejecutar `-name "*.htm"-o -name "*.html"` en un sub-shell... Otra solución podría haber sido poner los paréntesis entre comillas simples o dobles, pero aquí es preferible una contra-barra ya que solo tenemos que aislar un solo caracter.

Y finalmente, está el comando a ejecutar para cada uno de los archivos:

```
-exec ln {} /var/www/obsolete \;
```

Aquí también, tiene que desactivar el `;` para el *shell*, ya que de no ser así el *shell* lo interpretaría como un separador de comandos. Si no lo hace, `find` se quejará que le falta un argumento a `-exec`.

Un último ejemplo: tiene un directorio enorme denominado `/shared/images`, con todo tipo de imágenes en él. Regularmente, Ud. usa el comando *touch* para actualizar la fecha de un archivo denominado `stamp` en este directorio, para que tenga una referencia temporal. Ud. quiere encontrar todas las imágenes ***JPEG*** en el mismo que son más nuevas que el archivo `stamp`, y ya que Ud. obtuvo las imágenes de varias fuentes, estos archivos tienen las extensiones `jpg`, `jpeg`, `JPG` o `JPEG`. También quiere evitar buscar en el directorio `old`. Quiere que se le envíe la lista de estos archivos por correo electrónico, y su nombre de usuario es `darth`:

```
find /shared/images -cnewer     \
     /shared/images/stamp       \
     -a -iregex ".*\.jpe?g"     \
     -a -not -regex ".*/old/.*" \
       | mail darth -s "Imagenes nuevas"
```

¡Y eso es todo! Por supuesto, este comando no es muy útil si tiene que ingresarlo cada vez, y quisiera ejecutarlo regularmente... Puede hacer lo siguiente:

5.3. crontab: reportar o editar su archivo crontab

`crontab` es un comando que le permite ejecutar comandos a intervalos de tiempo regulares, con la ventaja adicional que no tiene que estar conectado al sistema y que el reporte de salida se le envía por correo electrónico. Los intervalos se pueden especificar en minutos, horas, días, e incluso meses. Dependiendo de las opciones, `crontab` actuará diferentemente:

- `-l`: Mostrar su archivo `crontab` corriente.
- `-e`: Editar su archivo `crontab`.
- `-r`: Eliminar su archivo `crontab` corriente.
- `-u <usuario>`: Aplicar una de las opciones de arriba para el usuario `<usuario>`. Sólo `root` puede hacer esto.

  Comencemos editando un `crontab`. Si ingresa `crontab -e`, estará frente a su editor de texto favorito si tiene definida la variable de entorno `EDITOR` o la variable de entorno `VISUAL`, caso contrario se usará *VI*. Una línea de un archivo `crontab` se compone de seis campos. Los primeros cinco campos son los intervalos de tiempo para los minutos, horas, días en el mes, meses y días en la semana. El sexto campo es el comando a ejecutar. Las líneas que comienzan con un `#` se consideran como comentarios y serán ignoradas por `crond` (el programa que es responsable de ejecutar los archivos `crontab`). Aquí tiene un ejemplo de `crontab`:

  Nota: para poder imprimir esto con una tipografía legible, tenemos que separar las líneas largas. Por lo tanto, algunos trozos deben ser ingresados en una única línea. Cuando el caracter \ termina en una línea,

esto significa que la línea tiene que continuarse. Esta convención funciona en los archivos `Makefile` y en el *shell*, así como también en otros contextos.

```
# Si no quiere recibir correo electrónico simplemente
#   "comente" la siguiente línea
#MAILTO=""
#
# Hacer un reporte de todas las imágenes nuevas a
#   las 14 hs. cada dos días, desde el ejemplo de
#   arriba - depués de eso, "retocar" el archivo de
#   "estampa". El "%" se considera como una línea
#   nueva, esto le permite poner varios comandos
#   en una misma línea.
0 14 */2 * *  find /shared/images              \
  -cnewer /shared/images/stamp                \
  -a -iregex ".*\.jpe?g"                      \
  -a -not -regex                              \
    ".*/old/.*"%touch /shared/images/stamp
#
# Cada Navidad, reproducir una melodía :)
0 0 25 12 * mpg123 $HOME/canciones/feliz_navidad.mp3
#
# Imprimir la lista de compras cada martes a las 17 hs...
0 17 * * 2 lpr $HOME/lista_de_compras.txt
```

Hay muchas otras maneras de especificar los intervalos aparte de las que se muestran. Por ejemplo, puede especificar un conjunto de ***valores discretos*** separados por comas (`1,14,23`) o un rango (`1-15`), o incluso una combinación de ambos (`1-10,12-20`), con un paso opcional (`1-12,20-27/2`). Ahora queda en sus manos encontrar comandos útiles para poner **:-)**

5.4. at: programar un comando, pero solo una vez

También podría querer ejecutar un comando un día dado, pero no regularmente. Por ejemplo, quiere que se le recuerde de una cita, hoy a las 18 horas. Ud. emplea *X*, y quiere que se le notifique, por ejemplo, a las 17:30 hs. que debe irse. `at` es lo que Ud. quiere aquí:

```
$ at 5:30pm
  # Ahora está frente al prompt "at"
at> xmessage "¡Hora de irse! Cita a las 6pm"
  # Presione C-d para
at> <EOT>
$
```

Se puede especificar la hora de diferentes maneras:

- `now +<intervalo>`: Significa eso, ahora, más un intervalo opcional. (Si no se especifica el intervalo significa ahora mismo). La sintaxis para el intervalo es `<n>` (minutes|hours|days|weeks|months) (minutos|horas|días|semanas|meses ; sólo en inglés). Por ejemplo, puede especificar `now + 1 hour` (dentro de una hora), `now + 3 days` (dentro de tres días) y así sucesivamente.
- `<hora> <día>`: Especificar la fecha por completo. El parámetro `<hora>` es obligatorio. `at` es muy liberal en lo que acepta: por ejemplo, puede ingresar `0100`, `04:20`, `2am`, `0530pm`, `1800`, o uno de los tres valores especiales: `noon` (mediodía), `teatime` (la hora del té, 16 hs.) o `midnight` (medianoche). El parámetro `<día>` es opcional. También puede especificarlo de diferentes maneras: `12/20/2001` por ejemplo, notación americana para el 20 de diciembre de 2001, o, a la europea, `20.12.2001`. Puede omitir el año, pero entonces

sólo se acepta la notación europea: `20.12`. También puede especificar el mes por su abreviatura en inglés: `Dec 20` o `20 Dec` son ambos válidos.

`at` también acepta opciones diferentes:

- `-l`: Imprime la lista de los trabajos que están programados; el primer campo es el número de trabajo. Esto es equivalente al comando `atq`.
- `-d <n>`: Quita el trabajo número `<n>` de la lista. Puede obtener los números de los trabajos con el comando `atq` o con la opción anterior. Esto es equivalente al comando `atrm <n>`.

Como siempre, vea la página Man ejecutando `man 1 at` para más opciones.

5.5. tar: `Tape ARchiver` (Archivador de cinta)

Aunque ya hemos visto un uso para `tar` en el capítulo *Compilando e instalando software libre*, página 139, no hemos explicado como funciona. Para eso está esta sección aquí. Al igual que `find`, `tar` es un utilitario *Unix* de larga data, y como tal, su sintaxis es un poco especial. La sintaxis es:

```
tar [opciones] [archivos...]
```

Ahora, aquí tiene la lista de opciones. Note que todas tienen una opción larga equivalente, pero para esto se tendrá que referir a la página Man ya que no se indicarán aquí. Y, por supuesto, tampoco se indicarán todas las opciones **:-)**

> **Nota:** el guión inicial (`-`) de las opciones cortas ahora no está desaprobado para el comando `tar`, excepto después de una opción larga.

- `c`: Esta opción se usa para crear archivadores nuevos.
- `x`: Esta opción se usa para extraer los archivos de un archivador existente.
- `t`: Listar los archivos de un archivador existente.
- `v`: Esto simplemente listará los archivos mientras se agregan o se extraen de un archivador, o, en conjunción con la opción `t` (ver arriba), imprime un listado largo de archivo en vez de uno corto.
- `f <archivo>`: Crear, un archivador de nombre `<archivo>`, extraer del archivador `<archivo>` o listar los archivos del archivador `<archivo>`. Si se omite este parámetro, el archivo predeterminado será `/dev/rmt0`, que generalmente es el archivo especial asociado con el ***streamer***. Si el parámetro `archivo` es `-` (un guión, la entrada o la salida (dependiendo de si está creando un archivador o extrayendo de uno) será asociado con la entrada estándar o la salida estándar.
- `z`: Le dice a `tar` que el archivador a crear debe comprimirse con `gzip`, o que el archivador del que se quiere extraer está comprimido con `gzip`.
- `j`: Igual que `z`, pero el programa usado para la compresión es `bzip2`.
- `p`: Cuando se extraen archivos de un archivador, preservar todos los atributos del archivo, incluyendo pertenencia, último tiempo de acceso, y así sucesivamente. Muy útil para los volcados del sistema de archivos.
- `r`: Agregar la lista de archivos dada en la línea de comandos a un archivador existente. Note que el archivo al cual quiere agregar archivos ¡**no** debe estar comprimido!

- `A`: Añadir los archivadores que se dan en la línea de comandos al que se da con la opción `f`. Al igual que con la opción `r`, los archivadores no deben estar comprimidos para que esto funcione.

Hay muchas, muchas, muchas otras opciones, para una lista completa querrá referirse a la página Man de `tar`. Vea, por ejemplo, la opción `d`. Ahora, sigamos con un poquito de práctica. Digamos que quiere crear un archivador con todas las imágenes en `/shared/images`, comprimirlo con `bzip2`, que va a llamar `images.tar.bz2` y ubicarlo en su directorio personal. Entonces, ingresará:

```
 #
 # Nota: ¡debe encontrarse en el directorio desde el
 #   que quiere archivar los archivos!
 #
$ cd /shared
$ tar cjf ~/images.tar.bz2 images/
```

Como puede ver, aquí hemos usado tres opciones: `c` le dijo a `tar` que queríamos crear un archivador, `j` le dijo que lo queríamos comprimir con `bzip2`, y `f ~/images.tar.bz2` le dijo que el archivador se iba a crear en nuestro directorio personal, con el nombre `images.tar.bz2`. Ahora queremos verificar si el archivador es válido. Simplemente lo podemos hacer listando sus archivos:

```
 #
 # Volver a nuestro directorio personal
 #
$ cd
$ tar tjvf images.tar.bz2
```

Aquí, le dijimos a `tar` que liste (`t`) los archivos del archivador `images.tar.bz2` (`f images.tar.bz2`), le advertimos que el archivador estaba comprimido con `bzip2` (`j`), y que queríamos un listado largo (`v`). Ahora, digamos que borró el directorio de las imágenes. Afortunadamente, su archivador está intacto, y ahora lo quiere extraer de nuevo a su lugar original, en `/shared`. Pero como no quiere arruinar su comando `find` para las imágenes nuevas, necesita conservar todos los atributos del archivo:

```
 #
 # cambiar al directorio donde quiere extraer
 #
$ cd /shared
$ tar jxpf ~/images.tar.bz2
```

¡Y eso es todo!

Ahora, digamos que quiere extraer sólo el directorio `images/cars` del archivador, y nada más. Entonces puede ingresar esto:

```
$ tar jxf ~/images.tar.bz2 images/cars
```

Si esto le preocupa, que no lo haga: si intenta respaldar archivos especiales, `tar` los tomará como lo que son, archivos especiales, y no volcará su contenido. Así que sí, se puede poner `/dev/mem` en un archivador **:-)** ¡Ah!, y también trata correctamente a los vínculos, así que tampoco se preocupe por esto. Para los vínculos simbólicos, también mire la opción `h` en la página Man.

5.6. bzip2 y gzip: comandos de compresión de datos

Puede ver que ya hemos hablado de estos dos programas cuando tratábamos con `tar`. A diferencia de `WinZip` bajo *Windows*, el archivador y la compresión se hacen usando dos utilitarios separados – `tar` para el archivador, y los dos programas que presentaremos ahora para la compresión de datos, `bzip2` y `gzip`. También existen otros utilitarios de compresión para *GNU/Linux* tales como `zip`, `arj` o `rar` (pero no se usan con frecuencia).

Al principio, `bzip2` había sido escrito como un reemplazo de `gzip`. Sus relaciones de compresión generalmente son mejores, pero por otra parte, toma más memoria. La razón por la cual todavía está aquí `gzip` es que todavía es más usado que `bzip2`.

Ambos comandos tienen una sintaxis similar:

```
gzip [opciones] [archivo(s)]
```

Si no se especifica un nombre de archivo, tanto `gzip` como `bzip2` esperarán datos de la entrada estándar y enviarán los resultados a la salida estándar. Por lo tanto, puede usar ambos programas en tuberías. También ambos programas tienen un conjunto de opciones en común:

- `-1`, ..., `-9`: Configuran el nivel de compresión. A mayor número, mejor compresión, pero mejor también significa más lenta: “dar para recibir”.
- `-d`: Descomprimir el(los) archivo(s). Esto es equivalente a usar `gunzip` o `bunzip2`.
- `-c`: Volcar el resultado de la compresión/descompresión de los archivos dados en la línea de comandos a la salida estándar.

> **Aviso**
>
> Predeterminadamente, tanto `gzip` como `bzip2` borran el o los archivos que han comprimido (o descomprimido) si no usa la opción `-c`. Con `bzip2` lo puede evitar usando la opción `-k` pero, `gzip` ¡no tiene tal opción!

Ahora, algunos ejemplos. Digamos que quiere comprimir todos los archivos que terminan con `.txt` en el directorio corriente usando `bzip2`, entonces usará:

```
$ bzip2 -9 *.txt
```

Digamos que quiere compartir su archivado de imágenes con alguien, pero dicha persona no tiene `bzip2`, sólo tiene `gzip`. Ud. no necesita descomprimir el archivador y volver a comprimirlo, simplemente puede descomprimirlo a la salida estándar, usar una tubería, comprimir la entrada estándar y volver a direccionar la salida al archivador nuevo:

```
bzip2 -dc images.tar.bz2 | gzip -9 >images.tar.gz
```

Y eso es todo. Podría haber ingresado `bzcat` en lugar de `bzip2 -dc`. Hay un equivalente para `gzip` pero su nombre es `zcat`, no **gzcat**. También tiene `bzless` (y `zless`, respectivamente) si quiere ver un archivo comprimido directamente, sin tener que descomprimirlo. Como ejercicio, intente encontrar el comando que tendría que ingresar para ver los archivos comprimidos sin descomprimirlos, y sin usar `bzless` o `zless` **:-)**

5.7. Mucho, mucho más...

Hay tantos comandos que un libro detallando todo acerca de ellos sería del tamaño de una enciclopedia. Este capítulo no ha cubierto ni un décimo del tema, sin embargo Ud. puede hacer mucho con lo que aprendió aquí. Si lo desea, puede leer algunas páginas Man: sort(1), sed(1) y zip(1) (sí, es lo que piensa: puede extraer o hacer archivadores `.zip` con *GNU/Linux*), convert(1), y así sucesivamente. La mejor manera de acostumbrarse a estas herramientas es practicar y experimentar con ellas, y es probable que les encuentre un montón de usos, incluso algunos inesperados. ¡Que se divierta! **:-)**

Capítulo 6. Control de procesos

6.1. Un poco más sobre los procesos

En *Los procesos*, página 25, mencionamos que era posible monitorear los procesos; eso es lo que trataremos luego. Para comprender las operaciones que vamos a hacer aquí, es útil saber un poco más acerca de ellas.

6.1.1. El árbol de procesos

Al igual que con los archivos, todos los procesos que corren en un sistema *GNU/Linux* están organizados en forma de árbol. La raíz de este árbol es `init`. Cada proceso tiene un número (su PID, *Process ID*, Identificador de proceso), junto con el número de su proceso padre (PPID, *Parent Process ID*, Identificador del proceso padre). El PID de `init` es 1, y también su PPID: `init` es su propio padre.

6.1.2. Las señales

Cada proceso en *Unix* puede reaccionar a las señales que se le envían. Existen 64 señales diferentes. Las 32 señales "más altas" (33 a 64) son señales de tiempo real, y están fuera del alcance de este capítulo. Para cada una de estas señales, el proceso puede redefinir su propio comportamiento, excepto para dos de ellas: la señal número 9 (`KILL`), y la señal número 19 (`STOP`).

La señal 9 termina un proceso irrevocablemente, sin darle tiempo de finalizar adecuadamente. Esta es la señal que se deberá enviar a un proceso cuando el mismo está trabado o exhibe otros problemas.Se encuentra disponible una lista completa de la señales usando el comando `kill -l`.

6.2. Obtener información sobre los procesos: ps y pstree

Estos dos comandos muestran una lista de los procesos presentes en el sistema, de acuerdo con los criterios que Ud. configura.

6.2.1. ps

Al enviar este comando sin un argumento se mostrarán solo los procesos iniciados por Ud. en la terminal que está utilizando:

```
$ ps
  PID TTY          TIME CMD
 5162 ttya1    00:00:00 zsh
 7452 ttya1    00:00:00 ps
```

Al igual que muchos utilitarios *Unix*, `ps` tiene una gran cantidad de opciones, y sólo veremos las más comunes:

- `a`: también muestra los procesos iniciados por los otros usuarios;
- `x`: también muestra los procesos sin terminal de control alguna o con una terminal de control diferente a la que Ud. está utilizando;
- `u`: muestra, para cada proceso, el nombre del usuario que lo inició y la hora a la cual fue iniciado.

Hay muchas otras opciones. Refiérase a la página Man para más información. ps(1).

La salida de este comando está dividida en campos diferentes: el que más le interesará es el campo `PID`, que contiene el identificador del proceso. El campo `CMD` contiene el nombre del comando ejecutado.

Una forma muy común de invocar a ps es la siguiente:

```
$ ps ax | less
```

Esto le da una lista de todos los procesos que se están ejecutando corrientemente, entonces puede identificar uno o más procesos que estén causando problemas y, subsecuentemente, puede "matarlos".

6.2.2. pstree

El comando pstree muestra los procesos en forma de estructura de árbol. Una ventaja es que Ud. puede ver inmediatamente quien es el proceso padre de otro: cuando quiere eliminar toda una serie de procesos, y si son todos padres e hijos, es suficiente matar al ancestro común. Puede usar la opción -p, que muestra el PID de cada proceso, y la opción -u que muestra el nombre del usuario que inició el proceso. Como generalmente la estructura de árbol es grande, es más fácil invocar a pstree de la siguiente manera:

```
$ pstree -up | less
```

Esto le da una visión general de toda la estructura de árbol de los procesos.

6.3. Envío de señales a los procesos: kill, killall y top

6.3.1. kill, killall

Estos dos comandos se usan para enviar señales a los procesos. El comando kill necesita el número de un proceso como argumento, mientras que el comando killall necesita el nombre de un comando.

Los dos comandos opcionalmente pueden recibir el número de una señal como argumento. Predeterminadamente, ambos envían la señal 15 (TERM) a el o los procesos relevantes. Por ejemplo, si quiere matar el proceso con PID 785, Ud. ingresa el comando:

```
$ kill 785
```

Si quiere enviarle la señal 9, entonces ingresa:

```
$ kill -9 785
```

Supongamos que quiere matar un proceso del cual Ud. conoce el nombre del comando. En vez de encontrar el número de proceso usando ps, puede matar el proceso directamente:

```
$ killall -9 netscape
```

Pase lo que pase, sólo matará a sus propios procesos (a menos que Ud. sea root), por lo que no debe preocuparse acerca de los procesos "del vecino" que tienen el mismo nombre, ellos no serán afectados.

6.3.2. top

top es un programa todo en uno: simultáneamente cumple las funciones de ps y kill. Es un comando de modo consola, por lo que debe iniciarlo desde una terminal, como se muestra en Figura 6-1.

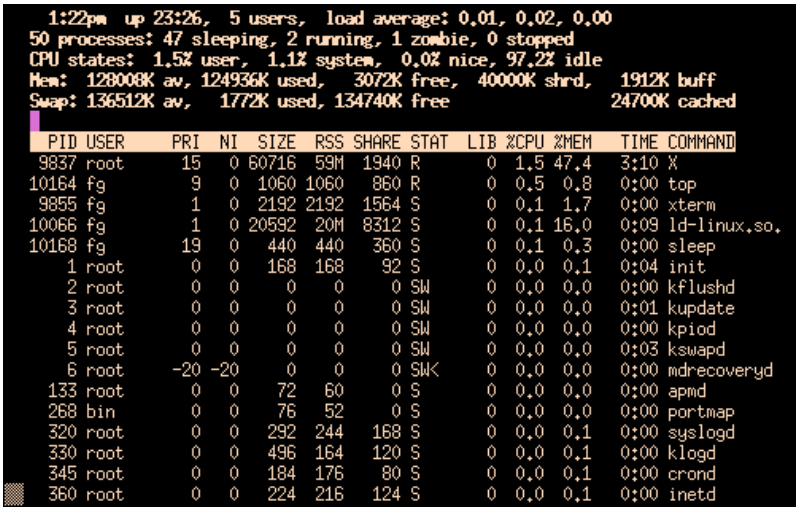

Figura 6-1. Ejemplo de ejecución de `top`

El programa se controla por completo con el teclado. Puede acceder a la ayuda presionando **h**, aunque esta está en inglés. Aquí tiene algunos de los comandos que puede usar.

- **k**: este comando se usa para enviar una señal a un proceso. Luego, `top` le preguntará por el PID del proceso, seguido del número de la señal a enviar (predeterminadamente, la número 15);
- **M**: este comando se usa para ordenar el listado de los procesos de acuerdo a la memoria que usan (campo `%MEM`);
- **P**: este comando se usa para ordenar el listado de procesos de acuerdo al tiempo de CPU que consumen (campo `%CPU`; este es el método de ordenamiento predeterminado);
- **u**: este comando se usa para mostrar los procesos de un usuario en particular, `top` le preguntará de cual. Debe ingresar el **nombre** del usuario, no su UID. Si no ingresa nombre alguno, se mostrarán todos los procesos;
- **i**: este comando actúa como un interruptor; predeterminadamente se muestran todos los procesos, incluso los que están dormidos; este comando asegura que se muestran sólo los procesos que están en curso de ejecución (los procesos cuyo campo `STAT` indica `R`, *`running`*, ejecutando) y no los otros. Una nueva llamada a este comando lo lleva a la situación previa.

II. Linux en profundidad

Capítulo 7. Organización del árbol de archivos

Hoy día, un sistema *Unix* es grande, muy grande. Esto es particularmente cierto con *GNU/Linux*: la profusión de software disponible lo harían un sistema inmanejable si no hubieran guías para la ubicación de los archivos en la estructura del árbol.

Respecto a eso, la norma reconocida es FHS (*Filesystem Hierarchy Standard*, Norma para la jerarquía del sistema de archivos), que, al momento de la escritura de este manual está en la versión 2.1. El documento que describe la norma está disponible en diferentes formatos en la Internet en la dirección http://www.pathname.com/fhs/. Este capítulo sólo da un breve sumario, pero debería ser suficiente para que Ud. sepa en que directorio debería buscar (o poner) un archivo dado.

7.1. Datos compartibles y no compartibles, estáticos y no estáticos

Sobre un sistema *Unix* los datos se pueden clasificar de acuerdo a estos dos criterios. Ud. puede haber adivinado lo que ambos significan: los datos compartibles son los datos que pueden ser comunes a varias máquinas en una red, mientras que los datos no compartibles no pueden serlo. Los datos estáticos no deben modificarse en el uso normal, mientras que los no estáticos pueden modificarse en el uso normal. A medida que exploremos la estructura del árbol, clasificaremos los diferentes directorios en cada una de estas categorías.

Note que estas clasificaciones son sólo recomendaciones. Ud. no está obligado a seguirlas, aunque adoptarlas le será de gran ayuda para administrar su sistema. Tenga presente también, que la distinción entre estático y no estático sólo se aplica al uso del sistema y no a la configuración del mismo. Si Ud. instala un programa, obviamente tendrá que modificar directorios "normalmente" estáticos, como `usr` por ejemplo.

7.2. El directorio raíz: /

El directorio raíz contiene toda la jerarquía del sistema. No se puede clasificar ya que sus sub-directorios pueden, o no, ser estáticos o compartibles. Aquí tiene una lista de los directorios y sub-directorios principales, junto con sus clasificaciones:

- `/bin`: archivos binarios esenciales del sistema. Este directorio contiene los comandos básicos que usarán todos los usuarios y son necesarios para la operación del sistema: `ls`, `cp`, *login*, etc. Estático, no compartible.
- `/boot`: contiene los archivos que necesita el administrador de arranque de *GNU/Linux* (*GRUB* o *LILO* para las plataformas **Intel**, *SILO* para las plataformas *Sparc*). Este puede, o no, contener al núcleo: si el núcleo no está aquí, debe estar ubicado en el directorio raíz. Estático, no compartible.
- `/dev`: archivos de los dispositivos del sistema (`dev` por *DEVices*, Dispositivos). Estático, no compartible.
- `/etc`: este directorio contiene todos los archivos de configuración específicos a la máquina. Estático, no compartible.
- `/home`: contiene todos los directorios personales de los usuarios del sistema. Este directorio puede, o no, ser compartible (algunas redes grandes lo hacen compartible por NFS). No estático, compartible.
- `/lib`: este directorio contiene las bibliotecas esenciales al sistema y los módulos del núcleo, en `/lib/modules`. Todas las bibliotecas que necesitan los binarios presentes en los directorios `/bin` y `/sbin` se deben ubicar aquí, junto con el vinculador `ld.so`. Estático, no compartible.
- `/mnt`: directorio que contiene los puntos de montaje para los sistemas de archivos temporales. No estático, no compartible.
- `/opt`: este directorio contiene los paquetes que no son necesarios para la operación del sistema. Se recomienda poner los archivos estáticos (binarios, bibliotecas, páginas de manual, etc.) de tales paquetes en

el directorio `/opt/nombre_del_paquete` y sus archivos de configuración específicos para la máquina en `/etc/opt`.

- `/root`: directorio personal "del que Todo lo Puede". No estático, no compartible.
- `/usr`: ver la sección siguiente. Estático, compartible.
- `/sbin`: contiene los binarios del sistema esenciales para el arranque del mismo, sólo utilizables por `root`. Un usuario no privilegiado también puede ejecutarlos pero no llegará muy lejos. Estático, no compartible.
- `/tmp`: directorio destinado a contener archivos temporales que pueden crear ciertos programas. No estático, no compartible.
- `/var`: ubicación para los datos que los programas pueden modificar en tiempo real (ej: el servidor de correo electrónico, los programas de auditoría, el servidor de impresión, etc.) Todo `var` es no estático, pero sus diferentes sub-directorios pueden ser compartibles, o no.

7.3. /usr: el grandote

El directorio `/usr` es el directorio principal de almacenamiento de las aplicaciones. Todos los archivos binarios en este directorio no deben ser necesarios para el arranque o mantenimiento del sistema, ya que por lo general, la jerarquía `/usr` está en un sistema de archivos separado. Debido a que su tamaño es generalmente importante, `usr` tiene su propia jerarquía de sub-directorios. Mencionaremos sólo algunos:

- `/usr/X11R6`: toda la jerarquía de *X Window System*. Todos los binarios necesarios para la operación de *X* (incluyendo los servidores *X*) y todas las bibliotecas necesarias se deben ubicar aquí. El directorio `/usr/X11R6/lib/X11` contiene todos los aspectos de la configuración de *X* que no varían de una máquina a otra. La configuración específica de cada máquina está en `/etc/X11`.
- `/usr/bin`: este directorio contiene la gran mayoría de los programas binarios del sistema. **Cualquier** programa binario que no sea necesario para el mantenimiento del sistema y no es un programa de administración del sistema se debe ubicar en este directorio, excepto los programas que instale Ud. mismo, que deben estar en `/usr/local`.
- `/usr/lib`: este directorio contiene todas las bibliotecas necesarias para emplear los programas ubicados en `/usr/bin` y `/usr/sbin`. También hay un vínculo simbólico `/usr/lib/X11` que apunta al directorio que contiene las bibliotecas de *X Window System*, `/usr/X11R6/lib` (si *X Window System* está instalado, por supuesto).
- `/usr/local`: en este directorio Ud. debería instalar sus aplicaciones personales. El programa de instalación habrá creado la jerarquía necesaria: `lib/`, `man/`, etc.
- `/usr/share`: este directorio contiene todos los datos necesarios para las aplicaciones de `/usr`, y todos los datos independientes de la plataforma. Entre otras cosas, Ud. encontrará la información de la zona y la localización (`zoneinfo` y `locale`).

También están los directorios `/usr/share/doc` y `/usr/share/man` que contienen, respectivamente, la documentación de las aplicaciones y las páginas Man del sistema.

7.4. /var: datos modificables durante el uso

El directorio `/var` contiene todos los datos operativos de los programas que están corriendo en el sistema. A diferencia de los datos de trabajo en `/tmp` estos datos deben quedar intactos en caso de volver a arrancar. Hay muchos sub-directorios, y algunos son muy útiles:

- `/var/log`: contiene los archivos de auditoría del sistema;

- `/var/spool`: contiene los archivos de trabajo de los demonios del sistema. Por ejemplo, `/var/spool/lpd` contiene los archivos de trabajo del servidor de impresión y `/var/spool/mail` los archivos de trabajo del servidor de correo electrónico (es decir, todo el correo que llega a su sistema y sale del mismo).
- `/var/run`: este directorio se usa para mantener la pista de todos los procesos que está usando el sistema, de forma tal que Ud. pueda actuar sobre estos procesos en caso de un cambio de ***nivel de ejecución*** del sistema (ver el capítulo *Los archivos de arranque: init sysv*, página 91).

7.5. /etc: los archivos de configuración

El directorio `/etc` es uno de los directorios esenciales de cualquier sistema *Unix*. Contiene todos los archivos básicos de configuración del sistema. ¡**Nunca** lo borre para ganar espacio! De la misma forma, recuerde que si quiere extender su estructura del árbol sobre varias particiones, no debe poner `/etc` en una partición separada: es necesario para la inicialización del sistema.

Algunos archivos importantes son:

- `passwd` y `shadow`: estos dos archivos, son archivos de texto que contienen a todos los usuarios del sistema y sus contraseñas cifradas. `shadow` sólo está presente si Ud. usa las contraseñas *shadow*, lo cual es la opción de instalación predeterminada;
- `inittab`: es el archivo de configuración del programa `init`, que juega un rol fundamental cuando se arranca el sistema, como veremos más adelante;
- `services`: este archivo contiene una lista de los servicios de red existentes;
- `profile`: este es el archivo de configuración del *shell*, aunque ciertos *shells* usan otros. Por ejemplo, *Bash* usa `bashrc`;
- `crontab`: archivo de configuración de `cron`, el programa responsable de la ejecución periódica de comandos.

También hay ciertos sub-directorios para los programas que necesitan una gran cantidad de archivos de configuración. Por ejemplo, esto se aplica a *X Window System* que tiene todo el directorio `/etc/X11`.

Capítulo 8. Sistemas de archivos y puntos de montaje

La mejor forma de entender “como funciona” es ver un caso práctico, que es lo que vamos a hacer aquí. Suponga que Ud. recién ha comprado un disco rígido nuevo, todavía sin partición alguna. Su partición **Linux-Mandrake** está llena a más no poder, y en vez de comenzar desde cero, Ud. decide mover toda una sección de la estructura de árbol a su disco rígido nuevo. Como este disco rígido nuevo es muy grande, Ud. decide mover al mismo su directorio más grande: `/usr`. Pero primero, un poquito de teoría.

8.1. Principios

Como ya hemos mencionado en la **Guía de Instalación**, cada disco rígido se divide en varias particiones, y cada una de ellas contiene un sistema de archivos. Mientras *Windows* asocia una letra a cada uno de estos sistemas de archivos (bueno, en realidad sólo a los que reconoce), *GNU/Linux* tiene una estructura de árbol de archivos única, y cada sistema de archivos está ***montado*** en algún lugar de la estructura del árbol.

Así como *Windows* necesita una “unidad C:”, *GNU/Linux* tiene que poder montar la raíz de su sistema de archivos (`/`) en algún lugar, de hecho en una partición que contiene al ***sistema de archivos raíz***. Una vez que está montada la raíz, puede montar otros sistemas de archivos en la estructura de árbol, sobre diferentes ***puntos de montaje*** en la misma. Cualquier directorio bajo la raíz puede oficiar de punto de montaje. Note que también puede montar el mismo sistema de archivos varias veces.

Esto permite una gran flexibilidad en la configuración. Por ejemplo, en el caso de un servidor *web*, es común dedicar toda una partición al directorio que contiene los datos del servidor *web*. El directorio que generalmente los contiene es `/var/www`, que, por lo tanto, oficiará de punto de montaje para la partición. Puede ver en Figura 8-1 y Figura 8-2 la situación del sistema antes y después de montar el sistema de archivos.

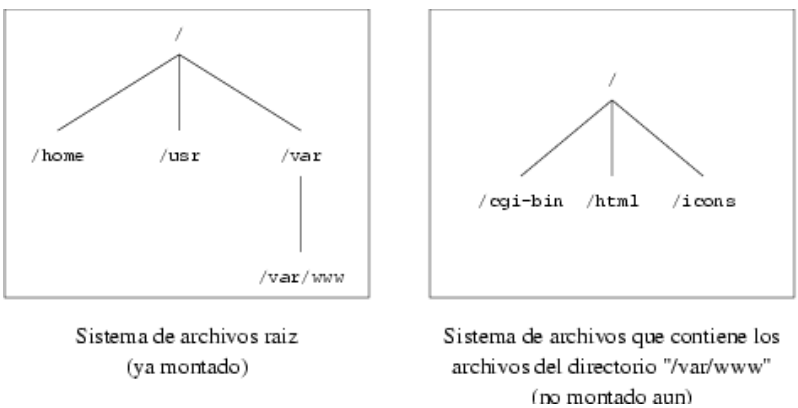

Figura 8-1. Un sistema de archivos todavía no montado

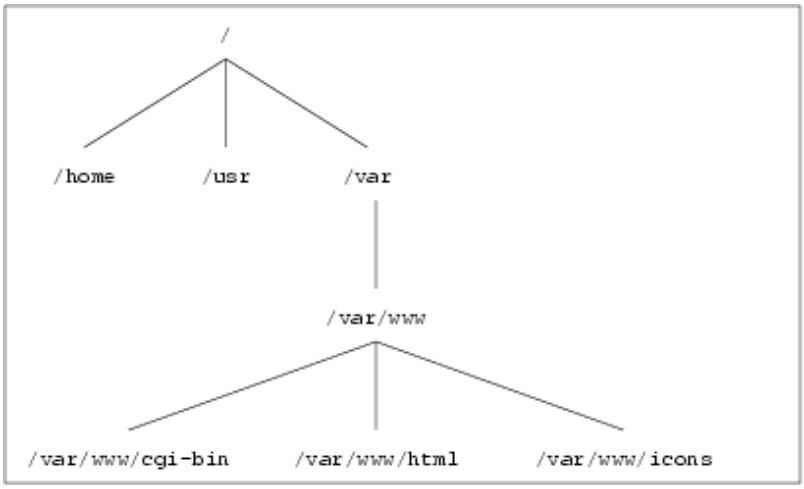

Figura 8-2. Ahora el sistema de archivos está montado

Como puede imaginar, esto presenta un número de ventajas: la estructura de árbol será siempre la misma, ya sea que esta se extiende sobre un sistema de archivos solo o sobre varias docenas[1]. Siempre es posible mover físicamente una parte clave de la estructura de árbol a otra partición cuando empieza a faltar espacio, que es lo que vamos a hacer aquí.

Hay dos cosas que Ud. debe saber sobre los puntos de montaje:

1. el directorio que oficia de punto de montaje debe existir,
2. y este directorio **preferentemente debería estar vacío**: si un directorio elegido como punto de montaje ya contiene archivos y sub-directorios, los mismos sencillamente serán “ocultados” por el sistema de archivos recién montado, pero no se podrá acceder más a ellos hasta que libere el punto de montaje.

8.2. Particionar discos rígidos y formatear particiónes

Acerca de los principios referidos arriba y para nuestro interés en esta sección, hay dos cosas por notar: un disco rígido se divide en particiones y cada una de estas particiones alberga un sistema de archivos. Ahora, por este instante, su disco rígido totalmente nuevo no tiene ni uno ni lo otro, así que es aquí por donde debe comenzar, con la partición en primer lugar. Para poder hacer eso, debe ser `root`.

Primero, tiene que saber el “nombre” de su disco rígido (es decir, el archivo que lo designa. Supongamos que configura a su disco nuevo como esclavo en la interfaz IDE primaria, entonces el nombre del mismo será `/dev/hdb`.[2]. Por favor, debe referirse a la sección “Administrar sus particiones” de la **Guía del Usuario**, la cual explica como particionar un disco. Note que *DiskDrake* también creará por Ud. los sistemas de archivo.

8.3. Los comandos mount y umount

Ahora que se ha creado el sistema de archivos, puede montar la partición. Inicialmente, la misma estará vacía. El comando para montar sistemas de archivos es `mount` , y su sintaxis es la siguiente:

---

1. *GNU/Linux* puede administrar hasta 64 sistemas de archivos montados simultáneamente

2. La manera de encontrar el nombre de un disco rígido se explica en la **Guía de Instalación**.

```
mount [opciones] <-t tipo> [-o opciones de montaje]
  <dispositivo> <punto de montaje>
```

En este caso, nosotros queremos montar nuestra partición sobre `/mnt` (o cualquier otro punto de montaje que Ud. haya elegido – no se olvide que debe existir); el comando para montar nuestra partición creada recientemente es el siguiente:

```
$ mount -t ext2 /dev/hdb1 /mnt
```

La opción `-t` se usa para especificar el tipo de sistema de archivos que se supone alberga la partición. Entre los sistemas de archivos que encontrará con mayor frecuencia, están ext2fs (el sistema de archivos de *GNU/Linux*), VFAT (para todas las particiones *DOS*/*Windows*: FAT 12, 16 o 32) e ISO-9660 (sistema de archivos de CD-ROM). Si no especifica tipo alguno, `mount` intentará adivinar el sistema de archivos que alberga la partición leyendo el super-bloque. Es poco común que falle en esta operación.

La opción `-o` se usa para especificar una o más opciones de montaje. Estas opciones dependen del sistema de archivos usado. Debe referirse a la página Man mount(8) para más detalles.

Ahora que ha montado su partición nueva, necesita copiar todo el directorio `/usr` en la misma:

```
$ (cd /usr && tar cf - .) | (cd /mnt && tar xpvf -)
```

Ahora que se copiaron los archivos, podemos desmontar nuestra partición. Para hacerlo, el comando es `umount`. La sintaxis es simple:

```
umount <punto de montaje|dispositivo>
```

Entonces, para desmontar nuestra partición, podemos ingresar:

```
$ umount /mnt
```

o bien:

```
$ umount /dev/hdb1
```

Como esta partición se va a "convertir" en nuestro directorio `/usr`, necesitamos informarle esto al sistema. Para hacerlo, editamos:

8.4. El archivo /etc/fstab

El archivo `/etc/fstab` permite la automatización de ciertos sistemas de archivos, en particular en el arranque del sistema. Contiene una serie de líneas que describen los sistemas de archivos, sus puntos de montaje y otras opciones. Aquí tiene un ejemplo de un archivo `/etc/fstab`:

```
/dev/hda1   /           ext2    defaults        1 1
/dev/hda5   /home       ext2    defaults        1 2
/dev/hda6   swap        swap    defaults        0 0
/dev/fd0    /mnt/floppy auto    sync,user,noauto,nosuid,nodev,unhide 0 0
/dev/cdrom  /mnt/cdrom  auto    user,noauto,nosuid,exec,nodev,ro 0 0
none        /proc       proc    defaults        0 0
none        /dev/pts    devpts  mode=0622       0 0
```

Una línea contiene, en orden:

- el dispositivo que alberga al sistema de archivos,
- el punto de montaje,

- el tipo de sistema de archivos,
- las opciones de montaje,
- el `flag` del utilitario de copia de respaldo `dump`,
- el orden de verificación por `fsck` (*FileSystem CheckK*, Verificación del sistema de archivos),

Siempre hay una entrada para el sistema de archivos raíz. Las particiones *swap* son especiales ya que no son visibles en la estructura de árbol, y el campo de punto de montaje para estas particiones contiene la palabra clave `swap`. Estudiaremos el sistema de archivos `/proc` con mayor detalle en *El sistema de archivos /proc*, página 85. Otro sistema de archivos especial es `/dev/pts`.

Volvamos al tema. Ud. ha movido toda la jerarquía `/usr` a `/dev/hdb1` y quiere entonces que esta partición se monte como `/usr` en el arranque. En ese caso, necesita agregar una entrada al archivo `etc/fstab` como la siguiente:

```
/dev/hdb1        /usr          ext2    defaults  1 2
```

Ahora la partición será montada en cada arranque. También se verificará la misma, si es necesario.

Hay dos opciones especiales: `noauto` y `user`. La opción `noauto` especifica que el sistema de archivos no debe montarse en el arranque sino que debe montarse explícitamente. La opción `user` especifica que cualquier usuario puede montar y desmontar el sistema de archivos. Estas opciones se usan típicamente para el CD-ROM y para la disquetera. Hay otras opciones, y `/etc/fstab` incluso tiene su propia página Man (fstab(5)).

La última, pero no menos importante, de las ventajas de este archivo es que simplifica la sintaxis del comando `mount`. Para montar un sistema de archivos señalado en este, puede referenciar el punto de montaje o el dispositivo. Así, para montar un disquete, puede ingresar:

```
$ mount /mnt/floppy/
```

o bien:

```
$ mount /dev/fd0/
```

Para finalizar con nuestro ejemplo de mover una partición: hemos copiado la jerarquía `/usr` y completado `/etc/fstab` de forma tal que la partición nueva se monte en el arranque. Pero por ahora, ¡todavía los archivos antiguos de `/usr` están allí! Por lo tanto, necesitamos borrarlos para liberar espacio (que era, después de todo, nuestro objetivo primario). Para hacerlo, primero necesita pasar a modo de usuario único (ejecutando el comando `telinit 1` en la línea de comandos), y luego:

- borrar todos los archivos del directorio `/usr` (es decir, del “antiguo”, ya que el “nuevo” todavía no está montado): `rm -Rf /usr/*`;
- montar el “nuevo” `/usr`: `mount /usr/`

y eso es todo. Ahora, regrese al modo multiusuario (`telinit 3` o `telinit 5`), y si no tiene otra tarea administrativa por hacer, ahora debería terminar la sesión de `root`.

8.5. Una nota acerca de la característica Supermount

Los núcleos recientes tal y como se envían con **Linux-Mandrake**, traen una característica interesante para los usuarios que usan disquetes y CDs con frecuencia. Instalada (o no) de acuerdo con el nivel de seguridad que eligió, esta monta y desmonta los soportes automáticamente a medida que se insertan o remueven. Esto es bastante útil ya que no tiene la necesidad de ejecutar `mount` o `umount` cada vez.

Si alguna vez desea cambiar ese comportamiento, y volver al viejo y querido montado manual de los soportes removibles, simplemente debe ejecutar, como `root`, el comando

```
supermount -i disable
```

Si desea habilitar la característica `supermount`, ejecute:

```
supermount -i enable
```

Para más información sobre el comando `supermount`, consulte la página Man (supermount(8)).

Capítulo 9. El sistema de archivos de Linux

La **Guía del Usuario** puede haber introducido los conceptos de posesión de archivos y permisos de acceso, pero la verdadera comprensión del ***sistema de archivos*** de *Unix* (y esto también se aplica al *ext2fs* de *GNU/Linux*) requiere que volvamos a definir el concepto de archivo en sí mismo. Una razón es que:

9.1. Todo es un archivo

Aquí, "todo" **realmente** significa todo. Un disco rígido, una partición en un disco rígido, un puerto paralelo, una conexión a un sitio *web*, una placa *Ethernet*, todos estos son archivos. Incluso los directorios son archivos. *GNU/Linux* reconoce muchos tipos de archivos además de los archivos regulares y los directorios. Note que aquí por tipo de archivo no nos referimos al tipo de **contenido** de un archivo: para *GNU/Linux* y cualquier sistema *Unix*, un archivo, ya sea una imagen GIF, un archivo binario o lo que sea, sólo es un flujo de bytes. Diferenciar a los archivos de acuerdo a su contenido es algo dejado a las aplicaciones.

9.1.1. Los diferentes tipos de archivos

Si recuerda bien, cuando Ud. hace un `ls -l`, el caracter antes de los derechos de acceso identifica el tipo de un archivo. Ya hemos visto dos tipos de archivos: los archivos regulares (`-`) y los directorios (`d`). También puede encontrarse con estos otros tipos si se desplaza por el árbol de archivos y lista el contenido de los directorios:

1. ***Archivos de modo caracter*** Estos archivos son o bien archivos especiales del sistema (tal como `/dev/null`, que ya hemos visto), o bien periféricos (puertos serie o paralelo), que comparten la particularidad de que su contenido (si es que tienen alguno) no se conserva en memoria [1]. Dichos archivos se identifican con la letra `'c'`.
2. ***Archivos de modo bloque*** Estos archivos son periféricos, y, a diferencia de los archivos de modo caracter, su contenido, **está** "bufereado"[2]. Los archivos que entran en esta categoría son, por ejemplo, los discos rígidos, las particiones de un disco rígido, las unidades de disquete, las unidades de CD-ROM y así sucesivamente. Los archivos `/dev/hda`, `/dev/sda5` son un ejemplo de archivos de modo bloque. En la salida de `ls -l`, estos están identificados por la letra `'b'`.
3. ***Vínculos simbólicos*** Estos archivos son muy comunes, y se usan ampliamente en el procedimiento de inicio del sistema de **Linux-Mandrake** (vea el capítulo *Los archivos de arranque: init sysv*, página 91). Como su nombre lo indica, su propósito es vincular archivos de forma simbólica, lo que significa que dichos archivos pueden o no apuntar a un archivo existente. Esto se explicará más adelante en este capítulo. Generalmente (y equivocadamente, como veremos más adelante) se los conoce como ***"soft links"*** (en inglés), y están identificados por una `'l'`.
4. ***Tuberías nombradas*** En caso que se lo pregunte, sí, estos son muy similares a las tuberías usadas en los comandos del *shell*, pero con la particularidad que estas, en realidad, tienen nombre. Siga leyendo para aprender más. Sin embargo, son muy raras, y es muy poco probable que vea una durante su viaje por el árbol de archivos. Sólo en caso de que los vea, la letra que las identifica es `'p'`. Para aprender más acerca de ellas eche un vistazo a *Tuberías "anónimas" y tuberías nombradas*, página 79.
5. ***Sockets*** Este es el tipo de archivo para todas las conexiones de red. Pero sólo unos pocos tienen nombre. Más aun, hay distintos tipos de sockets y sólo se puede vincular uno, pero esto va más allá del alcance de este libro. Dichos archivos se identifican con la letra `'s'`.

1. Se dice de ellos "no-bufereados", que proviene del inglés `unbuffered`

2. Conservado en memoria

Aquí tiene un ejemplo de cada archivo:

```
$ ls -l /dev/null /dev/sda  /etc/rc.d/rc3.d/S20random /proc/554/maps \
  /tmp/ssh-reina/ssh-510-agent
crw-rw-rw-    1 root     root       1,   3 may  5  1998 /dev/null
brw-rw----    1 root     disk       8,   0 may  5  1998 /dev/sda
lrwxrwxrwx    1 root     root           16 dic  9 19:12 /etc/rc.d/rc3.d/S20random
-> ../init.d/random*
pr--r--r--    1 reina    reina           0 dic 10 20:23 /proc/554/maps|
srwx------    1 reina    reina           0 dic 10 20:08 /tmp/ssh-reina/ssh-510-agent=
$
```

9.1.2. I-nodos

Los i-nodos son, junto con el paradigma “Todo es un archivo”, la parte fundamental de cualquier sistema de archivos *Unix*. La palabra ***i-nodo*** es una abreviación de *Information NODE* (NODO de Información).

Los i-nodos se almacenan en el disco en una **tabla de i-nodos**. Existen para todos los tipos de archivos que se pueden almacenar en un sistema de archivos, y esto incluye a los directorios, las tuberías nombradas, los archivos de modo caracter, y así sucesivamente. Esto nos lleva a esta otra frase famosa: “El i-nodo es el archivo”. Los i-inodos también son la forma en la que *Unix* identifica de forma unívoca a un archivo.

Sí, leyó bien: en *Unix*, Ud. **no identifica a un archivo por su nombre**, sino por un número de i-nodo[3]. La razón para esto es que un mismo archivo puede tener varios nombres, o incluso ninguno. En *Unix*, un nombre de archivo es simplemente una entrada en un i-nodo de directorio. Tal entrada se denomina **vínculo**. Veamos a los vínculos con más detalle.

9.2. Los vínculos

La mejor forma de comprender qué hay detrás de esta noción de vínculo es por medio de un ejemplo. Creemos un archivo (regular):

```
$ pwd
/home/reina/ejemplo
$ ls
$ touch a
$ ls -il a
  32555 -rw-rw-r--    1 reina       reina              0 sep 10 08:12 a
```

La opción `-i` del comando `ls` imprime el número de i-nodo, que es el primer campo de la salida. Como puede ver, antes de crear el archivo `a`, no había archivo alguno en el directorio. El otro campo de interés es el tercero, que es el contador de vínculos del archivo (bueno, de hecho, del i-nodo).

De hecho, el comando `touch a` puede separarse en dos acciones distintas:

---

3. Importante: note que los números de i-nodo son únicos **para cada sistema de archivos**, lo cual significa que puede existir un i-nodo con el mismo número en otro sistema de archivos Esto nos lleva a la diferencia entre i-nodos “en disco” e i-nodos “en memoria”. Aunque los i-nodos “en disco” pueden tener el mismo número si se encuentran en sistemas de archivo diferentes, los i-nodos “en memoria” tienen un número único a través de todo el sistema. Una solución para obtener la unicidad es, por ejemplo, hacer un hash de el número de i-nodo “en disco” contra el identificador del dispositivo de bloques.

- creación de un i-nodo, al cual el sistema le atribuyó el número 32555, y cuyo tipo es el de un archivo regular,
- creación de un vínculo a este i-nodo, llamado `a`, en el directorio corriente, `/home/reina/ejemplo`. Por lo tanto, el archivo `/home/reina/ejemplo/a` es un vínculo al i-nodo numerado 32555, y por el momento es sólo uno: el contador de vínculos muestra un 1.

Pero ahora, si ingresamos:

```
$ ln a b
$ ls -il a b
  32555 -rw-rw-r--    2 reina       reina              0 sep 10 08:12 a
  32555 -rw-rw-r--    2 reina       reina              0 sep 10 08:12 b
$
```

habremos creado otro vínculo al mismo i-nodo. Como puede ver, no hemos creado archivo alguno denominado `b`, sino que sólo hemos agregado otro vínculo al i-nodo numerado 32555 en el mismo directorio y lo denominamos `b`. Puede ver en la salida de `ls -l` que el contador de vínculos para el i-nodo ahora es 2, y ya no es 1.

Ahora, si hacemos:

```
$ rm a
$ ls -il b
  32555 -rw-rw-r--    1 reina       reina              0 sep 10 08:12 b
$
```

vemos que incluso cuando hemos borrado el “archivo original”, el i-nodo todavía existe. Pero ahora el único vínculo a él es el archivo denominado `/home/reina/ejemplo/b`.

Por lo tanto, bajo *Unix* un archivo no tiene nombre alguno; en su lugar, tiene uno o más ***vínculo***s en uno o más directorios.

También los directorios se almacenan en i-nodos, pero su contador de vínculos, contrariamente a todos los otros tipos de archivos, es el número de sub-directorios que contiene. Existen al menos dos vínculos por directorio: el directorio en sí mismo (`.`) y su directorio padre (`..`).

Ejemplos típicos de archivos que no están vinculados (es decir, no tienen un nombre) son las conexiones de red: nunca verá el archivo correspondiente a su conexión con el sitio *web* de Linux-Mandrake (http://www.linux-mandrake.com/) en su árbol de archivos, sin importar que directorio intente. Similarmente, cuando usa una ***tubería*** en el *shell*, el archivo que corresponde a la misma existe, pero no está vinculado.

9.3. Tuberías “anónimas” y tuberías nombradas

Volvamos al ejemplo de las tuberías, ya que es sumamente interesante además de ser una buena ilustración de la noción de vínculos. Cuando usa una tubería en una línea de comandos, el *shell* crea la tubería para Ud. y la opera de tal manera que el comando antes de la misma escribe en ella, mientras que el comando después de la misma lee de ella. Todas las tuberías, ya sean anónimas (como las que usa el *shell*) o nombradas (ver abajo), funcionan según el principio FIFO (*First In, First Out*, Primero en Llegar, Primero en Salir). Ya hemos visto ejemplos de como usar las tuberías en el *shell*, pero tomemos uno en pos de nuestra ilustración:

```
$ ls -d /proc/[0-9] | head -6
/proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
```

Una cosa que no notará en este ejemplo (porque ocurre muy rápido para que uno lo pueda ver) es que las escrituras en las tuberías son bloqueantes. Esto significa que cuando el comando `ls` escribe en la tubería, está bloqueado hasta que un proceso del otro lado lea sobre la misma. Para poder visualizar el efecto, puede crear tuberías nombradas, que al contrario de las usadas por el *shell*, tienen nombres (es decir, están vinculadas, mientras que las del *shell* no lo están)[4]. El comando para crear dichas tuberías es `mkfifo`:

```
$ mkfifo un_tubo
$ ls -il
total 0
    169 prw-rw-r--    1 reina       reina              0 sep 10 14:12 un_tubo|
  #
  # Ud. puede ver que el contador de vínculos es 1, y que la salida muestra
  # que el archivo es una tubería ('p').
  #
  # Aquí también puede usar ln:
  #
$ ln un_tubo el_mismo_tubo
$ ls -il
total 0
    169 prw-rw-r--    2 reina       reina              0 sep 10 15:37 un_tubo|
    169 prw-rw-r--    2 reina       reina              0 sep 10 15:37 el_mismo_tubo|
$ ls -d /proc/[0-9] >un_tubo
  #
  # El proceso está bloqueado, ya que no hay quien lea en el otro extremo.
  # Teclee C-z para suspender el proceso...
  #
zsh: 3452 suspended  ls -d /proc/[0-9] > un_tubo
  #
  # ...Luego póngalo en 2do. plano:
  #
$ bg
[1]  + continued  ls -d /proc/[0-9] > un_tubo
  #
  # ahora lea del tubo...
  #
$ head -6 <el_mismo_tubo
  #
  # ...el proceso que escribe termina
  #
[1]  + 3452 done       ls -d /proc/[0-9] > un_tubo
/proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
#
```

Similarmente, también las lecturas son bloqueantes. Si ejecutamos los comandos anteriores en orden inverso, observaremos que `head` se bloquea, esperando que algún proceso le de algo para leer:

```
$ head -6 <un_tubo
#
# El programa se bloquea, suspenderlo: C-z
#
zsh: 741 suspended  head -6 < un_tubo
```

4. Existen otras diferencias entre los dos tipos de tuberías, pero las mismas están fuera del alcance de este libro.

```
#
# Ponerlo en segundo plano...
#
$bg
[1]  + continued  head -6 < un_tubo
#
# ...Y darle algo de comer :)
#
$ ls -d /proc/[0-9] >el_mismo_tubo
$ /proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
[1]  + 741 done       head -6 < un_tubo
$
```

En el ejemplo previo también se puede ver un efecto no deseado: el comando `ls` terminó antes que el comando `head` tomara el relevo. La consecuencia es que Ud. volvió al prompt inmediatamente, pero `head` sólo se ejecutará después. Por lo tanto sólo efectuó su salida después que Ud. volvió al prompt.

9.4. Los archivos “especiales”: modo bloque y caracter

Como ya se mencionó, dichos archivos son archivos creados por el sistema o bien, representan periféricos en su máquina. También hemos mencionado que el contenido de los archivos en modo bloque está guardado en memoria mientras que el de los de modo caracter no lo está. Para ilustrar esto, inserte un disquete en la disquetera e ingrese el comando siguiente dos veces:

```
$ dd if=/dev/fd0 of=/dev/null
```

Ud. puede observar lo siguiente: mientras que, la primera vez que se lanzó el comando, se leyó todo el contenido del disquete, la segunda vez no se accedió a la disquetera en absoluto. Esto se debe simplemente a que el contenido de la disquetera se guardó en memoria la primera vez que se lanzó el comando – y entre tanto Ud. no cambie el disquete, el mismo permanece allí.

Pero ahora, si quiere imprimir un archivo grande de esta forma (sí, va a funcionar):

```
$ cat /a/big/printable/file/somewhere >/dev/lp0
```

el comando tardará el mismo tiempo si lo lanza una vez, dos veces, o cincuenta veces. Esto se debe a que `/dev/lp0` es un archivo de modo caracter y su contenido no se conserva en memoria.

El hecho de que los archivos de modo bloque se conserven en memoria tiene un efecto secundario interesante: no sólo se conservan las lecturas sino también las escrituras. Esto permite que las escrituras en el disco sean asíncronas: cuando Ud. escribe un archivo en disco, la operación de escritura en sí misma no es inmediata. Sólo ocurrirá cuando *GNU/Linux* lo decida.

Finalmente, cada archivo especial tiene un número ***mayor*** y uno ***menor***. Aparecen en la respuesta de `ls -l`, en lugar del tamaño, debido a que el tamaño para este tipo de archivos es irrelevante:

```
$ ls -l /dev/hda /dev/lp0
brw-rw----    1 root     disk       3,   0 may  5  1998 /dev/hda
crw-rw----    1 root     daemon     6,   0 may  5  1998
```

Aquí, los números mayor y menor de `/dev/hda` son 3 y 0, respectivamente, mientras que para `/dev/lp0` son 6 y 0, respectivamente. Note que estos números son únicos por categoría de archivo, lo que significa que puede haber un archivo de modo caracter con 3 por mayor y 0 por menor (de hecho, este archivo existe: `/dev/ttyp0`), y similarmente sólo puede haber un archivo de modo bloque con 6 por mayor y 0 por menor.

Estos números existen por una razón simple: le permiten a *GNU/Linux* asociar las operaciones adecuadas para estos archivos (es decir, con los periféricos a los cuales se refieren estos archivos). No se puede controlar una disquetera de la misma manera que, digamos, un disco rígido SCSI.

9.5. Los vínculos simbólicos y la limitación de los vínculos “duros”

Aquí tenemos que enfrentar una incomprensión muy común, aun entre usuarios de *Unix*, que principalmente se debe al hecho de que los vínculos tal y como los hemos visto (erróneamente llamados vínculos “duros”) sólo están asociados a archivos regulares (y hemos visto que este no es el caso – e incluso que los vínculos simbólicos están “vinculados”). Pero esto requiere que expliquemos primero qué son los vínculos simbólicos (En inglés los vínculos simbólicos se denominan “softlinks”, o más comúnmente “symlinks”).

Los vínculos simbólicos son archivos de un tipo particular que sólo contienen una cadena de caracteres arbitraria, que puede, o no, apuntar a un nombre de archivo existente. Cuando se menciona un vínculo simbólico en la línea de comandos o en un programa, de hecho se accede al archivo al que apunta, si es que existe. Por ejemplo:

```
$ echo Hola >miarchivo
$ ln -s miarchivo mivinculo
$ ls -il
total 4
    169 -rw-rw-r--    1 reina       reina              6 sep 10 21:30 miarchivo
    416 lrwxrwxrwx    1 reina       reina              6 sep 10 21:30 mivinculo
-> miarchivo
$ cat miarchivo
Hola
$ cat mivinculo
Hola
```

Puede ver que el tipo de archivo para `mivinculo` es `'l'`, por *Link* (Vínculo). Los derechos de acceso para un vínculo simbólico son insignificantes: siempre serán `rwxrwxrwx`. También puede ver que este **es** un archivo diferente de `miarchivo`, ya que su número de i-nodo es diferente. Pero se refiere al archivo `miarchivo` de manera simbólica, por lo tanto cuando ingresa `cat mivinculo`, en realidad estará imprimiendo el contenido del archivo `miarchivo`. Para demostrar que un vínculo simbólico contiene una cadena de caracteres arbitraria, podemos hacer lo siguiente:

```
$ ln -s "No soy un archivo existente" otrovinculo
$ ls -il otrovinculo
    418 lrwxrwxrwx    1 reina       reina             20 sep 10 21:43 otrovinculo
-> No soy un archivo existente
$ cat otrovinculo
cat: otrovinculo: No existe el fichero o el directorio
$
```

Pero los vínculos simbólicos existen porque superan varias de las limitaciones de los vínculos normales (“duros”):

- no se puede crear un vínculo a un i-nodo en un directorio que está en un sistema de archivos diferente a dicho i-nodo. La razón es simple: el contador de vínculos se almacena en el i-nodo en sí mismo, y los i-nodos no pueden compartirse entre los sistemas de archivos. Los vínculos simbólicos sí lo permiten;
- no se pueden vincular dos directorios, debido a que el contador de vínculos para un directorio tiene un uso especial como hemos visto. Pero Ud. puede hacer que un vínculo simbólico apunte a un directorio y usarlo como si realmente fuera un directorio.

Por lo tanto los vínculos simbólicos son muy útiles en muchas circunstancias, y muy a menudo, la gente tiende a usarlos para vincular archivos entre sí, incluso cuando podría haberse usado un vínculo normal. No obstante, una ventaja de los vínculos normales es que Ud. no pierde el archivo si borra el "original".

Finalmente, si ha observado atentamente, sabrá que el tamaño de un vínculo simbólico es simplemente el tamaño de la cadena de caracteres.

9.6. Los atributos de los archivos

De la misma forma en que FAT tiene atributos de archivo (archivo, archivo de sistema, invisible), *ext2fs* tiene los suyos propios, pero son diferentes. Hablaremos de ellos aquí en pos de la integridad, pero no son muy usados. Sin embargo, si realmente quiere un sistema sumamente seguro, siga leyendo.

Hay dos comandos para manipular los atributos de los archivos: `lsattr(1)` y `chattr(1)`. Probablemente ya haya adivinado, `lsattr` muestra los atributos (del inglés *LiSt*), mientras que `chattr` los cambia (del inglés *CHange*). Estos atributos sólo se pueden aplicar a los directorios y a los archivos regulares. Ellos son los siguientes:

1. `A` (*no Access time*, sin tiempo de *A*cceso): Si un archivo o directorio tiene este atributo activo, cuando sea accedido, ya sea para lectura o para escritura, no se actualizará su última fecha de acceso. Esto puede ser útil, por ejemplo, para archivos o directorios que se acceden para escritura muy a menudo, especialmente debido a que este parámetro es el único que cambia en un i-nodo cuando se abre como sólo de lectura.
2. `a` (*append only*, Sólo para *a*djuntar): Si un archivo tiene este atributo activo y se abre para escritura, la única operación posible será agregar datos a su contenido previo, pero no renombrar ni borrar el archivo existente. Sólo `root` puede activar o desactivar este atributo.
3. `d` (*no dump*, sin respal*d*o): `dump (8)` es el utilitario *Unix* estándar para copias de seguridad. Vuelca cualquier sistema de archivos para el cual el contador de respaldo está en 1 en `/etc/fstab` (vea el capítulo *Sistemas de archivos y puntos de montaje*, página 71). Pero si un archivo o un directorio tiene este atributo activo, a diferencia del resto, no será tomado en cuenta cuando esté en progreso un respaldo. Note que para los directorios, esto también incluye a todos los archivos y sub-directorios que contienen.
4. `i` (*immutable*, *i*nmutable): Un archivo o directorio que tiene este atributo activo sencillamente no puede modificarse en absoluto: no se puede renombrar, no se puede crear algún otro vínculo al mismo[5] y no puede borrarse. Sólo `root` puede activar o desactivar este atributo. Note que esto también impide cambios al tiempo de acceso, por lo tanto, no necesita activar también el atributo `A` cuando se activa este.
5. `s` (*secure deletion*, seguridad para la acción de borrado): Cuando se borra un archivo o directorio con este atributo activo, los bloques que estaba ocupando en el disco se sobreescriben con ceros.
6. `S` (*Synchronous mode*, modo *S*incrónico): Cuando un archivo o directorio tiene este atributo activo, todas las modificaciones sobre el mismo son sincrónicas y se escriben en el disco inmediatamente.

Por ejemplo, podría querer activar el atributo `'i'` en los archivos esenciales del sistema para evitar malas sorpresas. También considere el atributo `'A'` en las páginas Man por ejemplo: esto evita un montón de operaciones de disco y, en particular, prolonga la duración de la batería en las portátiles.

5. debe asegurarse de entender que significa "agregar un vínculo" tanto para un archivo como para un directorio **:-)**

Capítulo 10. El sistema de archivos /proc

El sistema de archivos `/proc` es algo específico de *GNU/Linux*. El mismo es un sistema de archivos virtual, y como tal, no ocupa lugar en su disco. Es una forma muy conveniente de obtener información sobre el sistema, ya que la mayoría de los archivos de este directorio son legibles (bueno, con un poco de ayuda). De hecho, muchos programas obtienen información de los archivos de `/proc`, la formatean a su manera y luego la muestran. Este es el caso de todos los programas que muestran información sobre los procesos, y ya hemos visto algunos de ellos (`top`, `ps` y otros). `/proc` también es una buena fuente de información sobre su hardware, y similarmente, unos cuantos programas sólo son interfaces de la información contenida en `/proc`.

También hay un sub-directorio especial, `/proc/sys`. Este permite cambiar algunos parámetros del núcleo en tiempo real, o consultarlos.

10.1. Información sobre los procesos

Si Ud. lista el contenido del directorio `/proc`, verá muchos directorios cuyo nombre es un número. Estos son los directorios que contienen información sobre todos los procesos que están corriendo en el sistema en ese momento:

```
$ ls -d /proc/[0-9]*
/proc/1/    /proc/302/  /proc/451/  /proc/496/  /proc/556/  /proc/633/
/proc/127/  /proc/317/  /proc/452/  /proc/497/  /proc/557/  /proc/718/
/proc/2/    /proc/339/  /proc/453/  /proc/5/    /proc/558/  /proc/755/
/proc/250/  /proc/385/  /proc/454/  /proc/501/  /proc/559/  /proc/760/
/proc/260/  /proc/4/    /proc/455/  /proc/504/  /proc/565/  /proc/761/
/proc/275/  /proc/402/  /proc/463/  /proc/505/  /proc/569/  /proc/769/
/proc/290/  /proc/433/  /proc/487/  /proc/509/  /proc/594/  /proc/774/
/proc/3/    /proc/450/  /proc/491/  /proc/554/  /proc/595/
```

Note que como usuario no privilegiado, Ud. (lógicamente) sólo puede mostrar la información relacionada con sus propios procesos, pero no con los de los otros usuarios. Entonces, seamos `root` y veamos que información está disponible acerca del proceso 127:

```
$ su
Password:
$ cd /proc/127
$ ls -l
total 0
-r--r--r--    1 root     root            0 dic 14 19:53 cmdline
lrwx------    1 root     root            0 dic 14 19:53 cwd -> //
-r--------    1 root     root            0 dic 14 19:53 environ
lrwx------    1 root     root            0 dic 14 19:53 exe -> /usr/sbin/apmd*
dr-x------    2 root     root            0 dic 14 19:53 fd/
pr--r--r--    1 root     root            0 dic 14 19:53 maps|
-rw-------    1 root     root            0 dic 14 19:53 mem
lrwx------    1 root     root            0 dic 14 19:53 root -> //
-r--r--r--    1 root     root            0 dic 14 19:53 stat
-r--r--r--    1 root     root            0 dic 14 19:53 statm
-r--r--r--    1 root     root            0 dic 14 19:53 status
$
```

Cada directorio contiene las mismas entradas. Aquí tiene una descripción breve de algunas de ellas:

1. `cmdline`: este (pseudo-)archivo contiene toda la línea de comandos usada para invocar al proceso. No tiene formato: no hay un espacio entre el programa y sus argumentos, y tampoco hay un salto de línea al final. Para poder verlo, puede usar: `perl -pl -e 's,\00, ,g' cmdline`.

2. `cwd`: este vínculo simbólico apunta al directorio de trabajo corriente ("current working directory" en inglés, de allí el nombre) del proceso.
3. `environ`: este archivo contiene todas las variables de entorno definidas por este proceso, de la forma `VARIABLE=valor`. Al igual que con `cmdline`, la salida no tiene formato alguno: no hay saltos de línea para separar las diferentes variables, y tampoco al final. Una solución para verlo: `perl -pl -e 's,\00,\n,g' environ`.
4. `exe`: este es un vínculo simbólico que apunta al archivo ejecutable correspondiente al proceso en curso de ejecución.
5. `fd`: este sub-directorio contiene la lista de los "descriptores" de archivo abiertos actualmente por el proceso. Vea abajo.
6. `maps`: cuando Ud. muestra el contenido de esta tubería nombrada (por ejemplo, con `cat`), puede ver las partes del espacio de direccionamiento del proceso que en ese momento están proyectadas sobre un archivo. Los campos, de izquierda a derecha, son: el espacio de direccionamiento asociado a esta proyección, los permisos asociados a esta proyección, el desplazamiento desde el comienzo del archivo donde comienza la proyección, el dispositivo en el cual se encuentra el archivo proyectado, el número de i-nodo del archivo, y finalmente el nombre del archivo en sí mismo. Vea mmap(2).
7. `root`: este es un vínculo simbólico que apunta al directorio raíz usado por el proceso. Generalmente, será `/`, pero vea chroot(2).
8. `status`: este archivo contiene información diversa sobre el proceso: el nombre del ejecutable, su estado corriente su PID y su PPID, sus UID y GID reales y efectivos, su uso de memoria, y otra información.

Si listamos el contenido del directorio `fd`, siempre para nuestro proceso 127, obtenemos lo siguiente:

```
$ ls -l fd
total 0
lrwx------    1 root     root           64 dic 16 22:04 0 -> /dev/console
l-wx------    1 root     root           64 dic 16 22:04 1 -> pipe:[128]
l-wx------    1 root     root           64 dic 16 22:04 2 -> pipe:[129]
l-wx------    1 root     root           64 dic 16 22:04 21 -> pipe:[130]
lrwx------    1 root     root           64 dic 16 22:04 3 -> /dev/apm_bios
lr-x------    1 root     root           64 dic 16 22:04 7 -> pipe:[130]
lrwx------    1 root     root           64 dic 16 22:04 9 ->
/dev/console
$
```

De hecho, esta es la lista de los descriptores de archivo que abrió el proceso. Cada descriptor abierto está materializado por un vínculo simbólico cuyo nombre es el número del descriptor, y que apunta al archivo abierto por este descriptor[1]. También puede notar los permisos sobre los vínculos simbólicos: este es el único lugar donde los derechos tienen sentido, ya que representan los permisos con los cuales se abrió el archivo correspondiente al descriptor.

10.2. Información sobre el hardware

Aparte de los directorios asociados a los diferentes procesos, `/proc` también contiene una miríada de información sobre el hardware presente en su máquina. Un listado de los archivos del directorio `/proc` da lo siguiente:

```
$ ls -d [a-z]*
apm      dma          interrupts  loadavg  mounts      rtc       swaps
```

1. Si recuerda lo que se mencionó en la sección *Redirecciones y tuberías*, página 36, sabrá el significado de los descriptores 0, 1 y 2.

```
bus/     fb           ioports     locks    mtrr        scsi/     sys/
cmdline  filesystems  kcore       meminfo  net/        self/     tty/
cpuinfo  fs/          kmsg        misc     partitions  slabinfo  uptime
devices  ide/         ksyms       modules  pci         stat      version
$
```

Por ejemplo, si observamos el contenido de `/proc/interrupts`, podemos ver la lista de las interrupciones que el sistema está usando en ese momento, junto con el periférico que las está ocupando. Similarmente, `ioports` contiene la lista de los rangos de direcciones de entrada/salida ocupados en ese momento, y finalmente, `dma` hace lo mismo para los canales DMA. Por lo tanto, si desea solucionar un conflicto, observe el contenido de estos tres archivos:

```
$ cat interrupts
           CPU0
  0:     127648          XT-PIC  timer
  1:       5191          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  5:       1402          XT-PIC  xirc2ps_cs
  8:          1          XT-PIC  rtc
 10:          0          XT-PIC  ESS Solo1
 12:       2631          XT-PIC  PS/2 Mouse
 13:          1          XT-PIC  fpu
 14:      73434          XT-PIC  ide0
 15:      80234          XT-PIC  ide1
NMI:          0
$ cat ioports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
0300-030f : xirc2ps_cs
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(auto)
1050-1057 : ide0
1058-105f : ide1
1080-108f : ESS Solo1
10c0-10cf : ESS Solo1
10d4-10df : ESS Solo1
10ec-10ef : ESS Solo1
$ cat dma
 4: cascade
$
```

O, más simplemente, use el comando `lsdev` el cual obtiene información de estos tres archivos y la ordena por periférico, lo cual es, indudablemente, más conveniente[2]:

2. `lsdev` es parte del paquete `procinfo`.

```
$ lsdev
Device            DMA   IRQ  I/O Ports
----------------------------------
cascade             4     2
dma                            0080-008f
dma1                           0000-001f
dma2                           00c0-00df
ESS                            1080-108f 10c0-10cf 10d4-10df 10ec-10ef
fpu                           13  00f0-00ff
ide0                          14  01f0-01f7 03f6-03f6 1050-1057
ide1                          15  0170-0177 0376-0376 1058-105f
keyboard                       1  0060-006f
Mouse                         12
pic1                           0020-003f
pic2                           00a0-00bf
rtc                            8  0070-007f
serial                         03f8-03ff
Solo1                         10
timer                          0  0040-005f
vga+                           03c0-03df
xirc2ps_cs                     5  0300-030f
$
```

Una lista exhaustiva de los archivos presentes sería demasiado larga, sin embargo aquí tiene la descripción de algunos:

- `cpuinfo`: este archivo contiene, como su nombre (en inglés) lo indica, información sobre el(los) procesador(es) presente(s) en su máquina.
- `modules`: este archivo contiene una lista de los módulos que el núcleo está usando en ese momento, junto con el conteo del uso para cada uno. De hecho, esta es la misma información que reporta el comando `lsmod`.
- `meminfo`: este archivo contiene información sobre el uso de la memoria en el momento que Ud. muestra su contenido. Una información ordenada más claramente está disponible con el comando `free`.
- `apm`: si Ud. tiene una portátil, al mostrar el contenido de este archivo Ud. verá el estado de su batería. Puede ver si está conectada la alimentación externa, la carga actual de su batería, y la vida útil de la batería si el *BIOS* APM de su portátil lo soporta (desafortunadamente, este no es el caso general). Este archivo en sí mismo no es muy legible, por lo tanto Ud. querrá usar el comando `apm` en su lugar, que proporciona la misma información en un formato legible (si Ud. comprende el inglés...).
- `bus`: este sub-directorio contiene información sobre todos los periféricos que se encuentran en los diferentes buses de su máquina. Por lo general, la información es poco legible, y en su mayoría se trata y se reformatea con utilitarios externos: `lspcidrake`, `lspnp`, etc.

10.3. El sub-directorio /proc/sys

El rol de este sub-directorio es reportar los diferentes parámetros del núcleo, y permitir cambiar en tiempo real algunos de ellos. A diferencia de todos los demás archivos en `/proc`, algunos archivos de este directorio se pueden escribir, pero solo `root` puede hacerlo.

Una lista de los directorios y archivos presentes sería demasiado larga, y además dependería en gran parte de su sistema, y la mayoría de los archivos sólo serán útiles para aplicaciones muy especializadas. Sin embargo, aquí tiene tres usos comunes de este sub-directorio:

1. Autorizar el ruteo: Aunque el núcleo predeterminado de **Linux-Mandrake** puede enrutar, Ud. debe autorizarlo explícitamente. Para ello, tiene que ingresar el comando siguiente como `root`:

```
$ echo 1 >/proc/sys/net/ipv4/ip_forward
```

Reemplace el 1 por un 0 si desea prohibir el ruteo.

2. Prevenir la usurpación de la dirección IP: (*IP Spoofing*, en inglés) consiste en hacerle creer a uno que un paquete que viene del mundo externo viene de la interfaz por la cual llega. Esta técnica es muy usada por los crackers[3], pero Ud. puede hacer que el núcleo prevenga este tipo de intrusión. Sólo debe ingresar:

   ```
   $ echo 1 >/proc/sys/net/ipv4/conf/all/rp_filter
   ```

   y este tipo de ataque se vuelve imposible.

3. Incrementar el tamaño de la tabla de archivos abiertos y la tabla de i-nodos: Bajo *GNU/Linux* el tamaño de la tabla de archivos abiertos y la tabla de i-nodos es dinámico. Para un uso normal los valores predeterminados son suficientes, pero pueden ser insuficientes si su máquina es un servidor importante (por ejemplo, un servidor de bases de datos). Justamente el primer obstáculo es el hecho de que los procesos no pueden abrir más archivos porque la tabla está llena, por lo tanto Ud. debe incrementar el tamaño de la misma. Paralelamente, Ud. también debe incrementar el tamaño de la tabla de i-nodos. Estas dos líneas resolverán el problema:

   ```
   $ echo 8192 >/proc/sys/fs/file-max
   $ echo 16384 >/proc/sys/fs/inode-max
   ```

Para que estos parámetros se apliquen cada vez que arranque el sistema, puede agregar todas estas líneas al archivo `/etc/rc.d/rc.local` para evitar tener que volver a ingresarlas cada vez, pero otra solución es completar `/etc/sysctl.conf`, vea sysctl.conf(5).

3. ¡Y no por los hackers!

Capítulo 11. Los archivos de arranque: init sysv

En la tradición *Unix*, hay dos esquemas de arranque del sistema: el esquema *BSD* y el esquema "*System V*", ambos toman su nombre del sistema *Unix* que los implementó primero (*Berkeley Software Distribution* y **AT&T** *Unix System V*, respectivamente). El esquema *BSD* es el más simple, pero el esquema *System V*, aunque es menos fácil de entender (lo cual cambiará una vez que termine de leer este capítulo), definitivamente es más flexible de usar.

11.1. Al comienzo estaba init

Cuando el sistema arranca, luego de que el núcleo configuró todo y montó la raíz del sistema de archivos, inicia el programa `/sbin/init` [1]. `init` es el padre de todos los procesos del sistema, y es el responsable de llevar al sistema al ***nivel de ejecución*** (*runlevel*) deseado. En la próxima sección estudiaremos los distintos niveles de ejecución.

El archivo de configuración de `init` es `/etc/inittab`. Este archivo tiene su propia página Man (inittab(5)), pero aquí describiremos sólo algunos de los elementos de configuración.

La primer línea que debería ser el foco de su atención es esta:

```
si::sysinit:/etc/rc.d/rc.sysinit
```

Esta instrucción le dice a `init` que `/etc/rc.d/rc.sysinit` debe ejecutarse en la inicialización del sistema antes que cualquier otra cosa. Para determinar el nivel de ejecución predeterminado, `init` busca entonces la línea que contiene la palabra clave `initdefault`:

```
id:5:initdefault:
```

En este caso, `init` sabe que el nivel de ejecución predeterminado es 5. También sabe que para entrar en el nivel 5, debe ejecutar el comando siguiente:

```
l5:5:wait:/etc/rc.d/rc 5
```

Como puede ver, la sintaxis para cada uno de los niveles de ejecución es similar.

`init` también es responsable de reiniciar (`respawn`) ciertos programas que sólo él es capaz de reiniciar. Este es el caso, por ejemplo, de todos los programas de conexión que corren en cada una de las 6 terminales virtuales[2]. Para la segunda consola virtual, esto da:

```
2:2345:respawn:/sbin/mingetty tty2
```

11.2. Los niveles de ejecución

Todos los archivos relacionados con el arranque del sistema están ubicados en el directorio `/etc/rc.d`. Aquí tiene la lista de los mismos:

---

1. Ahora Ud. puede ver por qué poner `/sbin` en un sistema de archivos que no sea la raíz es un muy mala idea **:-)**

2. Por lo que Ud. puede, si quiere, agregar o quitar consolas virtuales modificando este archivo, hasta un máximo de 64, siguiendo la sintaxis. Pero no se olvide que *X* ¡también corre en una consola virtual! Entonces, por lo menos deje una libre para *X*.

```
$ ls /etc/rc.d
init.d/  rc.local*    rc0.d/  rc2.d/  rc4.d/  rc6.d/
rc*      rc.sysinit*  rc1.d/  rc3.d/  rc5.d/
```

En primer lugar, como hemos visto, se ejecuta el archivo `rc.sysinit`. Este es el archivo responsable de poner en su lugar la configuración básica de la máquina: tipo de teclado, configuración de ciertos dispositivos, verificación del sistema de archivos, etc.

Luego se ejecuta el script `rc`, con el nivel de ejecución deseado como argumento. Como hemos visto, el nivel de ejecución es un simple entero, y para cada nivel de ejecución `<x>` definido, debe haber un directorio `rc<x>.d` correspondiente. Entonces, en una instalación típica de **Linux-Mandrake**, puede ver que están definidos 6 niveles de ejecución:

- 0: Detención de la máquina por completo;
- 1: modo ***monousuario***; para ser usado en el caso de serios problemas o para la recuperación del sistema.
- 2: modo ***multi-usuario***, sin soporte para redes;
- 3: modo multi-usuario, con soporte para redes;
- 4: No usado;
- 5: Como 3, pero con la ejecución de la interfaz gráfica de conexión;
- 6: Volver a iniciar.

Observemos, por ejemplo, el contenido del directorio `rc5.d`:

```
$ ls rc5.d
K15postgresql@  K60atd@      S15netfs@   S60lpd@         S90xfs@
K20nfs@         K96pcmcia@   S20random@  S60nfs@         S99linuxconf@
K20rstatd@      S05apmd@     S30syslog@  S66yppasswdd@   S99local@
K20rusersd@     S10network@  S40crond@   S75keytable@
K20rwhod@       S11portmap@  S50inet@    S85gpm@
K30sendmail@    S12ypserv@   S55named@   S85httpd@
K35smb@         S13ypbind@   S55routed@  S85sound@
```

Como puede ver, todos los archivos de este directorio son vínculos simbólicos, y todos tienen una forma muy específica. Su forma general es

```
<S|K><orden><nombre_del_servicio>
```

. La `S` significa *`Start`* (arrancar) el servicio, y la `K` significa *`Kill`* (detener), el servicio. Los scripts se ejecutan por número de orden ascendente, y si dos scripts tienen el mismo número, se aplica el orden alfabético. También podemos ver que cada vínculo simbólico apunta a scripts ubicados en `/etc/rc.d/init.d` (excepto `local`), script que es responsable de controlar un servicio específico.

Cuando el sistema entra en un nivel de ejecución dado, comienza por ejecutar los vínculos `K` en orden: `rc` busca donde apunta el vínculo, luego llama al script correspondiente con un argumento solo `stop` (detener). Luego ejecuta los scripts `S`, todavía usando el mismo método, excepto por el hecho de que el script se llama con el argumento `start` (iniciar).

Por lo tanto, sin mencionar a todos los scripts, podemos ver que cuando el sistema entra en el nivel de ejecución 5, primero ejecuta `K15postgresql`, es decir, `/etc/rc.d/init.d/postgresql stop`. Luego `K20nfs`, luego `K20rstatd`, hasta el último; acto seguido, ejecuta todos los srcipts `S`: primero `S05ampd`, que entonces invoca a `/etc/rc.d/init.d/apmd start`, y así sucesivamente.

Armado con todo esto, Ud. puede crear su propio nivel de ejecución completo en pocos minutos, o evitar el arranque o la detención de un servicio borrando el vínculo simbólico correspondiente (también hay programas que son una interfaz para hacer esto, en particular *`DrakXServices`* y `chkconfig`; el primero es un programa gráfico).

III. Usos avanzados

Capítulo 12. Impresión

Este capítulo está dividido en dos partes: *Instalando y administrando impresoras*, página 95, dedicado a las personas que administran su máquina; y *Imprimiendo documentos*, página 101, que explica como usar una herramienta de impresión avanzada: *XPP*.

12.1. Instalando y administrando impresoras

Comenzando con la versión 7.2, **Linux-Mandrake**, está utilizando el sistema de impresión nuevo basado en *CUPS* (http://www.cups.org/) [1]. Esta es una herramienta muy potente que se basa en la administración y configuración descentralizada, haciendo que todas las impresoras que están en una red local estén disponibles para todos los usuarios.

12.1.1. Instale cups y navegue por su interfaz `web`

Debido a que ahora *CUPS* es el administrador de impresión predeterminado de **Linux-Mandrake**, todos los paquetes necesarios deberían instalarse de manera predeterminada. Si este no es el caso, debe asegurarse que están instalados al menos los paquetes `cups`, `cups-drivers` y `xpp`.

> **Nota:** Básicamente hay dos maneras de administrar sus impresoras con *CUPS*: una interfaz *web* y una aplicación para *KDE* denominada *Kups*. Ya hemos hablado sobre esta última en la **Guía del Usuario** y ahora nos concentraremos en la interfaz basada en *web* ya que la misma es accesible desde cualquier plataforma.

Desde su navegador *web* preferido, simplemente teclee http://localhost:631 (http://localhost:631/) en la ubicación o campo URL. Se mostrará el menú principal de *CUPS* (Figura 12-1).

---

1. *Common Unix Printing System* (Sistema de impresión común de *Unix*)

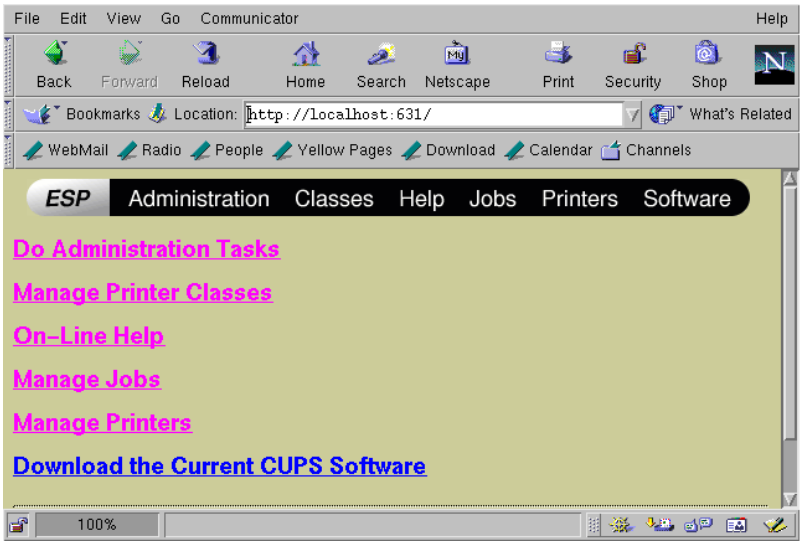

Figura 12-1. La página de bienvenida de CUPS

Ahora Ud. puede navegar por la interfaz de configuración como lo hace por un sitio *web*.

12.1.2. Configurar una impresora nueva

Dependiendo de si su LAN ya tiene máquinas con *CUPS* instalado y corriendo, Ud. debería ver o no una lista de impresoras bajo el vínculo Administrar Impresoras. Asumiremos que ahora Ud. está instalando una impresora conectada a su computadora personal aislada. Para configuraciones más complejas consulte la Ayuda en línea.

La página Administrar Impresoras (Figura 12-2) por ahora debería verse vacía.

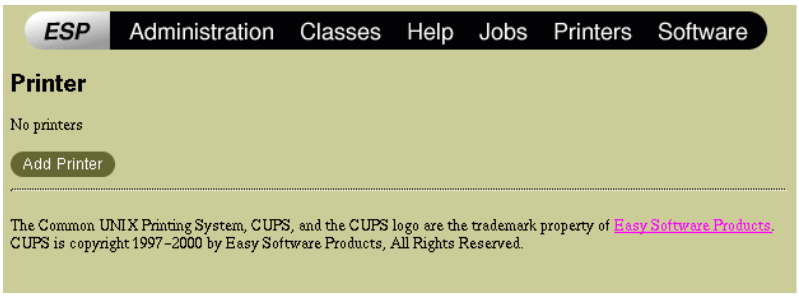

Figura 12-2. La lista de impresoras de CUPS vacía

Para configurar una impresora nueva, ahora haga click sobre el botón Agregar Impresora en la parte inferior de la página. Esto comenzará un procedimiento de cuatro etapas. Para ir de una etapa a la siguiente haga click sobre el botón Continuar luego de completar en la página todos los campos requeridos.

> **Nota:** La primera vez que Ud. quiere realizar una tarea administrativa con *CUPS*, este le pedirá la contraseña de `root` (Figura 12-3). Simplemente proporcione el nombre de conexión y la contraseña de `root` aquí.

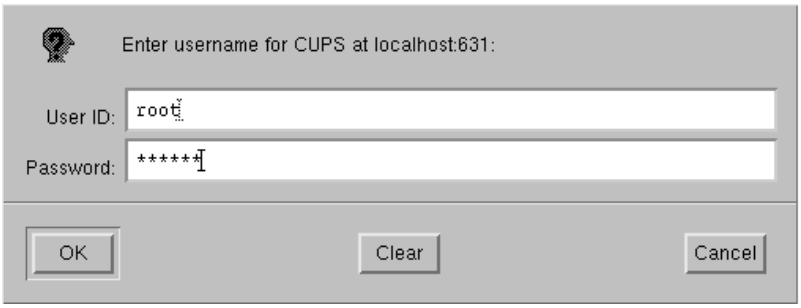

Figura 12-3. El diálogo de conexión de `CUPS`

12.1.2.1. Proporcionar información adicional acerca de la impresora

Este primer formulario presenta tres campos que Ud. puede llenar como le convenga para ayudar a los otros usuarios a conocer las impresoras físicas con las que tratan. El texto no tiene influencia sobre el comportamiento de la impresora, pero sin embargo debe llenarlos con cuidado, para evitar confusiones futuras.

Figura 12-4. Agregando una impresora nueva, etapa 1

Aquí sólo es necesario el nombre de la impresora.

12.1.2.2. Decir donde está conectada la impresora

Debe decirle a *`CUPS`* donde está ubicada físicamente la impresora. Si la impresora está conectada directamente a su computadora, entonces elija Puerto Paralelo #1, Puerto Serie, o USB dependiendo del tipo de su conexión.

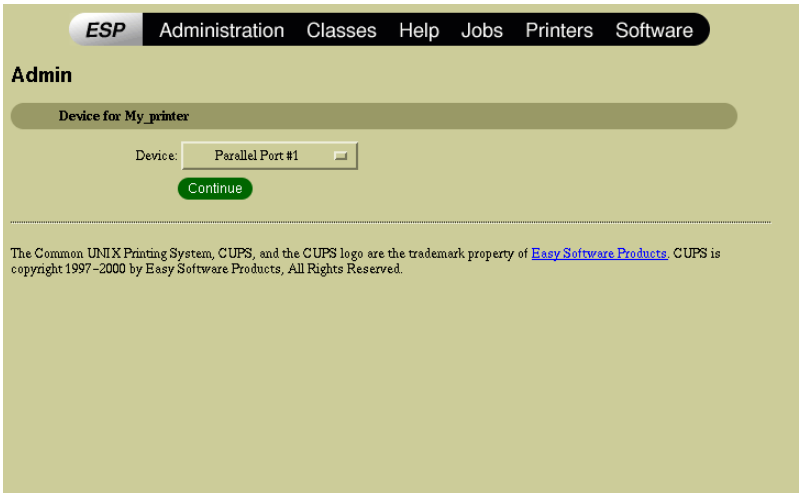

Figura 12-5. Agregando una impresora nueva, etapa 2

Están disponibles muchos tipos de conexión:

Ethernet

Para impresoras conectadas directamente a una red de área local.

LPD/LPR

Adecuado tanto para las impresoras que implementan este tipo de comportamiento directamente, como para las impresoras que están servidas por este tipo de cola de impresión. Generalmente, los servidores tipo *Unix* proporcionan este tipo de conexión.

Samba

Para impresoras servidas por servidores *Windows*. Note que para conectar a ese tipo de impresora, Ud. necesita instalar el paquete *Samba*.

12.1.2.3. Elija la marca de su impresora

Ahora es el momento de decirle a *CUPS* el tipo de impresora que Ud. está instalando. Simplemente debe marcar el nombre del fabricante en la lista.

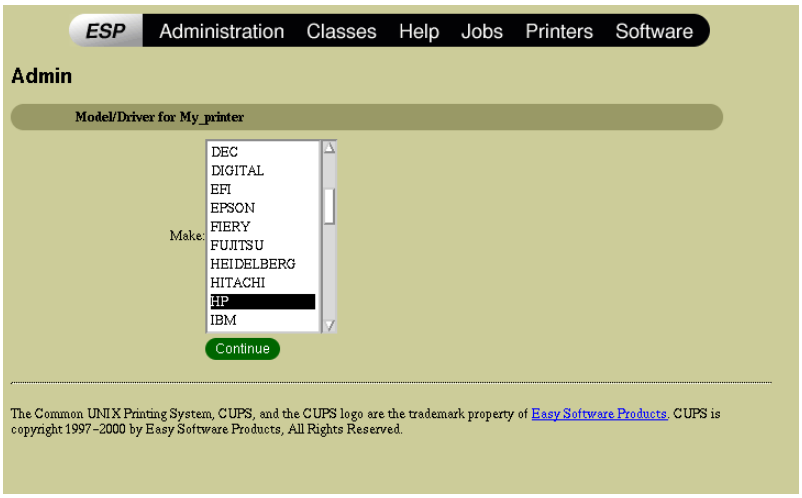

Figura 12-6. Agregando una impresora nueva, etapa 3

12.1.2.4. Elija el modelo de su impresora

Esta es la etapa final, la lista muestra ahora todos los modelos de ese fabricante específico de acuerdo a su elección previa. Aquí debe elegir su documento cuidadosamente.

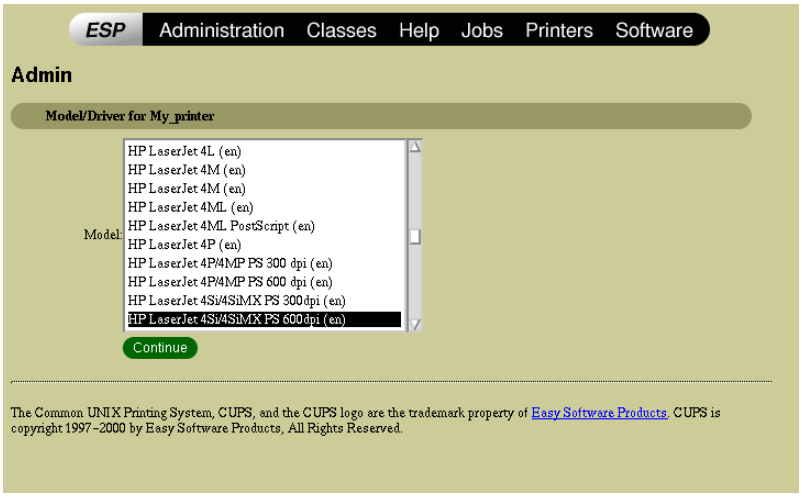

Figura 12-7. Agregando una impresora nueva, etapa 4

Si todo va bien, ahora Ud. debería ver su impresora nueva en la página Impresoras.

12.1.2.5. configuración final y prueba

Antes de probar la impresora, debe asegurarse que la configuración del tamaño del papel para esa impresora es la correcta. Vaya a la página de la impresora, y haga click sobre Configurar Impresora. Ud. ingresa a la página de parámetros de la impresora, se dirige a la sección `General` y elige el Tamaño de página apropiado. De hecho, algunas impresoras se niegan a imprimir si no tienen cargado el papel apropiado.

> **Aviso**
>
> Acerca de esta página de parámetros: siempre que Ud. cambie un parámetro en una sección, debe hacer click sobre el botón Continuar correspondiente para que sus cambios sean tomados en cuenta.

12.1.3. Una nota acerca de la seguridad

De forma predeterminada siempre que Ud. configura una impresora en su máquina, la misma está disponible para las demás personas que se encuentran en su red local. Si prefiere que las personas no puedan imprimir en la impresora suya, entonces debe editar el archivo de configuración de *CUPS* manualmente: `/etc/cups/cupsd.conf`. Simplemente debe reemplazar la línea

```
#BrowseInterval 30
```

por

```
BrowseInterval 0
```

También este archivo contiene un montón de opciones que le permiten un ajuste fino de su servidor de impresión. En particular, Ud. puede restringir el acceso desde máquinas o sub-redes específicas. Para más información, consulte los distintos comentarios dentro del archivo de configuración o consulte la ayuda en línea desde la interfaz *web*.

> **Sugerencia:** Siempre que Ud. realice cambios en el archivo de configuración, no olvide de volver a iniciar al demonio servidor de *CUPS* ejecutando:
>
> ```
> /etc/rc.d/init.d/cups restart
> ```

12.1.4. Administrar los trabajos de impresión

Esta característica es mayormente útil para impresoras muy ocupadas, pero puede llegar a necesitarla ocasionalmente para cancelar una impresión errónea de 10000 páginas por ejemplo. Siempre que Ud. envía un trabajo a una impresora puede consultar sus trabajos, y posiblemente todos los trabajos de todos los usuarios si Ud. es el administrador de la máquina que sirve esa impresora, mostrando luego la página específica a la impresora (Figura 12-8).

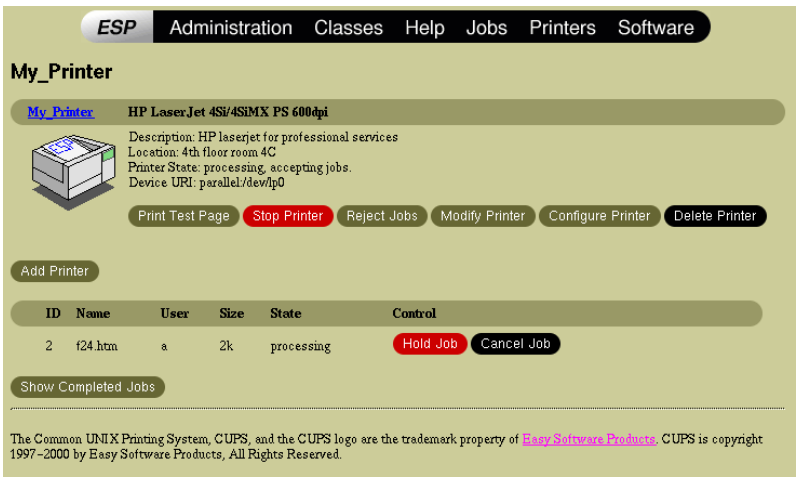

Figura 12-8. La página del estado de la impresora

Entonces Ud. puede realizar dos acciones diferentes sobre un trabajo específico:

- Suspender Trabajo: Para poner al trabajo en una lista de espera, sólo se imprimirá el mismo cuando Ud. regrese a este trabajo y presione el botón verde Reanudar Trabajo.
- Cancelar Trabajo: para cancelar definitivamente este trabajo (quitarlo de la cola).

Si desea hacer que su impresora no esté disponible temporalmente -por ejemplo, para cambiar el toner- simplemente puede hacer click sobre el botón Rechazar Trabajos. Luego, cuando la impresora vuelva a estar lista para aceptar trabajos, haga click sobre el botón Aceptar Trabajos.

> **Sugerencia:** Si está interesado en las capacidades de manejo de trabajos, debería echar un vistazo al programa `kups`.

12.2. Imprimiendo documentos

Reemplazando a los comandos de impresión tradicionales `lpr` y `lp` para imprimir documentos, ahora en **Linux-Mandrake** hay una bonita aplicación denominada *XPP* que permite a los usuarios imprimir archivos y configurar los parámetros de impresión para todas las impresiones. También se puede usar como el "comando de impresión" en las aplicaciones [2], de forma tal que Ud. puede acceder confortablemente a todas sus impresoras desde allí también.

12.2.1. Imprimir simplemente un archivo

Digamos que Ud. recién ha almacenado en su disco rígido una imagen de un sitio *web* y desea imprimirla. Primero lance *XPP* desde el menú Aplicaciones+Publicación→Panel de impresión X, aparecerá la ventana principal (Figura 12-9).

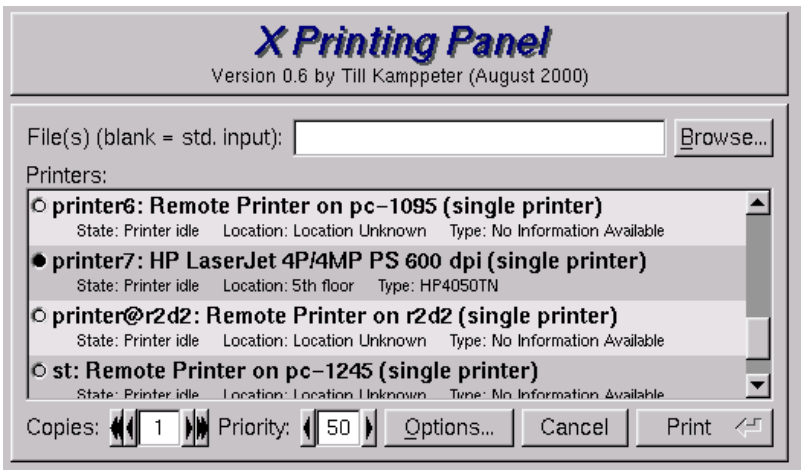

Figura 12-9. La ventana principal de `XPP`

Hay tres partes principales en esa ventana: un campo para especificar la ubicación de el o los archivos a imprimir, la lista de impresoras disponibles y un conjunto de aplicaciones en la parte inferior.

Para seleccionar el archivo a imprimir, haga click sobre el botón Browse..., este mostrará un diálogo (Figura 12-10) donde Ud. puede navegar por su disco y elegir el archivo que desea imprimir. Note que si Ud. sabe la ubicación también puede escribir la ruta completa a mano en el campo en blanco.

2. Para este propósito, también puede usar el comando `qtcups`

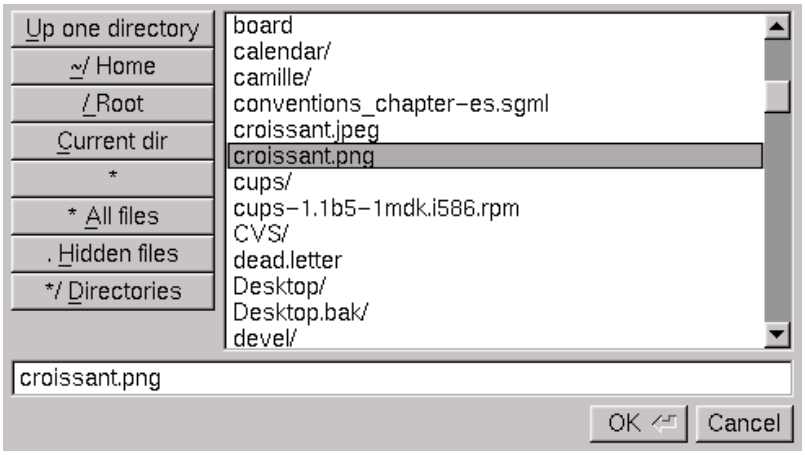

Figura 12-10. La selección de archivo de `XPP`

Luego asegúrese de que la impresora seleccionada (con un punto negro a la izquierda) es la correcta y haga click sobre el botón Imprimir.

> **Sugerencia:** Si desea usar *`XPP`* como el programa (filtro) de impresión para los trabajos de impresión que se envían desde otras aplicaciones tales como *`Netscape`* o muchas otras que lo permiten, simplemente debe cambiar el comando de impresión. Usualmente hay un campo en el diálogo de opciones de impresión que dice lpr, simplemente debe cambiarlo a `xpp`.

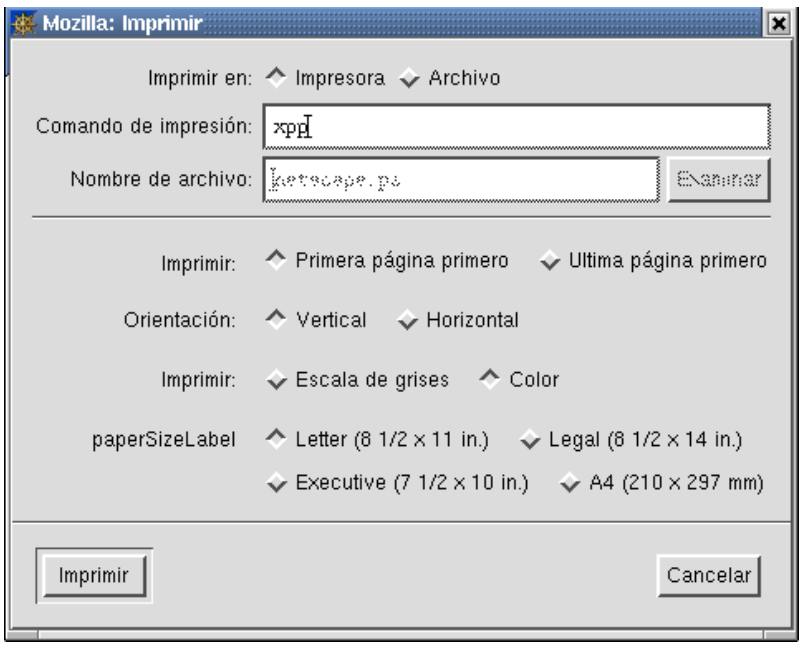

Figura 12-11. El diálogo de impresión de `Netscape`

Entonces cuando Ud. hace click sobre el botón Aceptar, aparecerá la ventana de *`XPP`*. Sólo necesita cambiar las opciones que desee, sin preocuparse acerca del archivo.

12.2.2. Configuración avanzada

`XPP` le permite un ajuste muy fino de sus impresiones. Antes que nada, hay opciones en la ventana principal que le permiten imprimir copias múltiples del mismo archivo (campo Copias) y cambiar la prioridad de su

trabajo de impresión. Esto último es útil para las impresoras muy ocupadas que usan muchos usuarios. Si Ud. necesita que se imprima un documento urgentemente, incremente el número de prioridad; si Ud. desea imprimir un documento que no necesita inmediatamente, disminuya el número de la prioridad.

Luego está el botón Options... que muestra un diálogo de opciones de solapas múltiples (Figura 12-12).

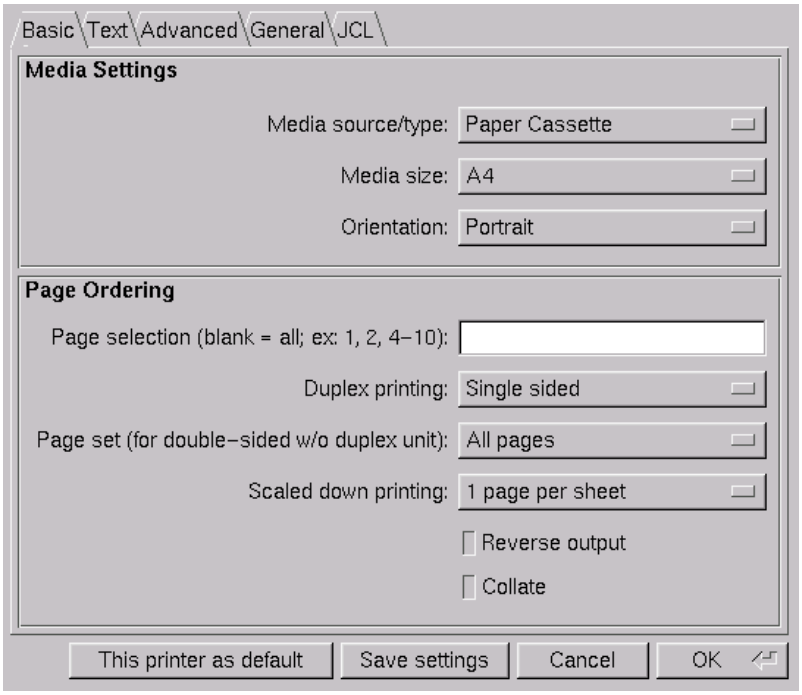

Figura 12-12. El diálogo de opciones básicas de `XPP`

La primera solapa se divide en dos secciones:

Configuraciones del papel

> Aquí puede elegir tres parámetros, dependiendo de la impresora, puede ser relevante o no. Primero la bandeja donde se encuentra el papel deseado o la alimentación manual, luego el tamaño del papel, y finalmente la orientación de la impresión.

Orden de las Páginas

> Estas opciones son muy útiles para definir la forma en que se imprime su documento y qué parte del mismo se imprime.
>
> - El ejemplo selección de página solo imprimiría las páginas 1, 2, 4, 5, 6, 7, 8, 9, 10 de su documento.
> - La opción Impresión Dúplex sólo es útil para las impresoras que la soportan. También se denomina impresión de ambos lados o de ambas caras.
> - Como se mencionó, la opción Conjunto de páginas le permite imprimir en dúplex incluso en impresoras que no soportan esta característica. Simplemente comience a imprimir las páginas pares, ponga el papel nuevamente en la bandeja (¡Tenga cuidado con la orientación!) e imprima sobre la otra cara las páginas impares.
> - Impresión reducida es una opción ecológica que permite la impresión de múltiples páginas por hoja de papel. Combinada con la impresión dúplex, debería ayudar a salvar el bosque tropical **:-)**
> - La opción Invertir la salida imprimirá las páginas comenzando por la última. Esto es útil para las impresoras que apilan el papel sólo boca para arriba, de forma tal que los documentos de múltiples páginas salen en el orden correcto.

- Finalmente, la opción Colacionar cambia el orden de las páginas cuando se imprimen múltiples copias del mismo archivo. Con la opción activa, el orden de un documento de tres páginas será 1,2,3,1,2,3; y sin la opción activa será 1,1,2,2,3,3.

Luego la solapa Texto (Figura 12-13) proporcionará opciones más adecuadas para cambiar la forma en que se imprimen los archivos de texto.

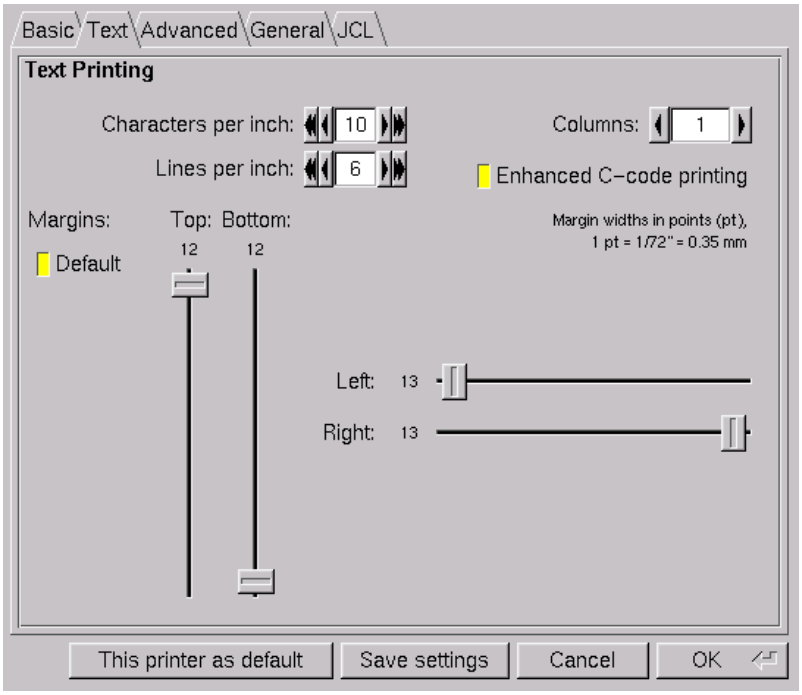

Figura 12-13. El diálogo de opciones de texto de `XPP`

Hay muchas opciones, cada una bastante fácil de entender. Sólo una nota acerca de la opción Impresión de código C mejorada, la cual imprimirá un encabezado para cada página y realizará el resaltado de la sintaxis para los listados de programas *C*.

La solapa Avanzadas (Figura 12-14) proporciona opciones para cambiar la apariencia visual de la página.

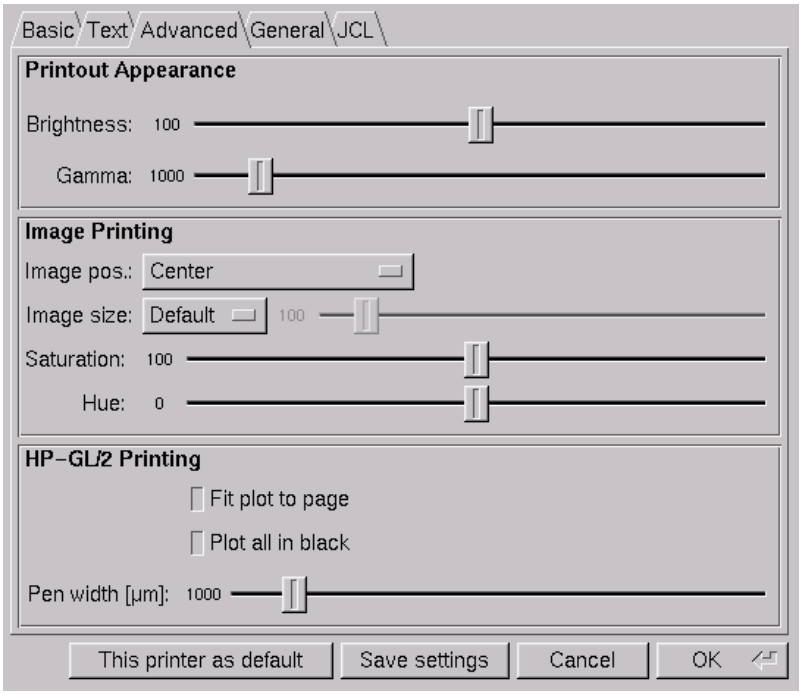

Figura 12-14. El diálogo de opciones avanzadas de `XPP`

Esta solapa está dividida en tres secciones:

Apariencia de la impresión

Estos dos parámetros, Brightness y gamma influenciarán principalmente las partes de tonos medios y las imágenes en la impresión (aclarar-obscurecer).

Impresión de imagen

Los primeros dos parámetros cambian la posición y el tamaño relativos de la imagen en la página; los últimos dos cambian la forma en la que se corrigen los colores.

Impresión HP-GL/2

Tres parámetros:

- Ajustar impresión a la página: Cambiar la escala de la impresión de forma tal que se ajuste exactamente al tamaño del papel.
- Imprimir todo en negro: Para imprimir sólo en blanco y negro.
- Ancho de pluma: para ajustar el ancho de la pluma “virtual” que imprime el documento.

Finalmente las solapas siguientes presentan opciones específicas a cada impresora: colores, resolución, medios tonos, etc.

Aviso

Cuando Ud. elige opciones que son incompatibles entre sí, las opciones problemáticas se muestran en rojo, de forma tal que Ud. pueda identificar el conflicto fácilmente. En esta situación, Ud. obtendrá un mensaje de error cuando haga click sobre Guardar opciones o Aceptar, primero debe corregir sus configuraciones.

Entonces cuando Ud. cambie las opciones hay dos posibilidades: o las guarda bien con el botón Guardar configuraciones de forma tal que las opciones se vuelvan a utilizar para las próximas impresiones, o bien simplemente hace click sobre Aceptar si estas opciones sólo son válidas para esa impresión en particular.

Capítulo 13. Solución de problemas

13.1. Introducción

Este capítulo lo guiará a través de algunas formas básicas de solucionar problemas, es decir: qué hacer cuando está todo mal o, mejor aun, qué hacer para estar **preparado** para cuando todo esté mal y como arreglarlo.

¿Cuantas veces se ha sentido tonto por no haber hecho copia de respaldo de ese archivito de configuración que se las ha arreglado para desconfigurar? Yo me he sentido así, muchas veces. En realidad, más veces de las que me gustaría admitir **:-)** ¿Cuantas veces ha perdido todas las preferencias de su programa después de instalar un software que no se comportó bien, o incluso después de borrar por accidente algún archivo de configuración? ¿Cuantas veces su computadora ha dejado de arrancar después de esas pruebas que estaba haciendo con el último núcleo?

Hay algunas personas que vuelven a compilar su núcleo o ajustan sus archivos de configuración cada día de la semana, cada semana del mes, o cada mes del año. Puede que Ud. no sea uno de ellos pero, créame, algún día va a querer, o necesitar, hacerlo; entonces asumamos que estos no son escenarios poco comunes en la vida diaria con *GNU/Linux*. Todos se pueden manejar sin problemas en absoluto si usa un poco de sentido común y sigue algunas prácticas y guías que le mostraremos. Esto lo ayudará cuando **esos** tiempos lleguen **:-)**

Ahora, pasemos a las cosas básicas que necesita para estar preparado...

13.2. Creando un disquete de arranque

La primerísima cosa que necesitará en caso de que su sistema no pueda arrancar más por cualquiera de las razones que expusimos con anterioridad, será un disquete de arranque. Debería haber creado uno durante el proceso de instalación. Un disquete de arranque le permitirá arrancar su sistema y poder deshacer, en cuestión de minutos, aquello que hizo que su sistema no pueda arrancar más.

> **Nota:** También puede usar el `Modo de Rescate` del CD-ROM de instalación de **Linux-Mandrake** para arrancar su máquina y realizar algunas tareas de mantenimiento, pero un disquete de arranque le puede resultar útil de todas formas (por ejemplo, si su máquina no soporta arrancar desde una unidad de CD-ROM).

Hay dos maneras de crear un disquete de arranque bajo **Linux-Mandrake**, usando la consola, y una gráfica. Para crear un disquete de arranque Ud. tiene que ser `root`. Si usa el método gráfico, se le pedirá la contraseña de `root` antes de continuar.

13.2.1. Usando la consola

Entonces, Ud. está en una consola, haga `su` para volverse `root` y teclee lo siguiente:

```
[root@localhost]# mkbootdisk -device /dev/fd0 2.2.17-21mdk
```

y presione **Intro**. Al hacerlo, le aparecerá algo como esto:

```
Insert a disk in /dev/fd0. Any information on the disk will be lost.
```

```
Press <Enter> to continue or ^C to abort:
```

Expliquemos el ejemplo dado. Entre otros parámetros, los dos que necesita `mkbootdisk` son `-device [dispositivo]`, que le dice a `mkbootdisk` el dispositivo sobre el cual queremos escribir el disquete de arranque. En nuestro ejemplo, hemos elegido `/dev/fd0`, el cual es la primer disquetera en el sistema. En un 99.9% de los casos eso debería funcionar, si no funciona en su caso, bueno, elija el dispositivo correcto para usar.

El otro parámetro necesario es `[versión_del_núcleo]`, el cual le dice a `mkbootdisk` el núcleo que queremos poner en el dispositivo elegido. En nuestro ejemplo, elegimos `2.2.17-21mdk`. Tendrá que cambiarlo de acuerdo al núcleo que está usando en su sistema. Por lo tanto, el ejemplo dado creará un disquete de arranque en `/dev/fd0` conteniendo el núcleo `2.2.17-21mdk`.

Por favor, note que esto creará un disquete de arranque que está basado en su núcleo corriente (en caso que Ud. así lo elija) con todos los módulos y cosas que tiene el núcleo. Si no desea incluir todas esas cosas en su disquete de arranque, o incluso si desea cambiar algo como, digamos, agregar un módulo para soportar unidades de cinta, es mejor que use nuestra herramienta gráfica *`drakfloppy`*.

13.2.2. Usando drakfloppy para crear un disquete de arranque

`drakfloppy` es una herramienta gráfica que le permite crear un disquete de arranque altamente personalizado. Para iniciar *`drakfloppy`* vaya al menú Configuración/Arranque e Init e inícielo desde allí. Se le pedirá su contraseña de `root`, a menos que ya sea `root`, como se ve en Figura 13-1:

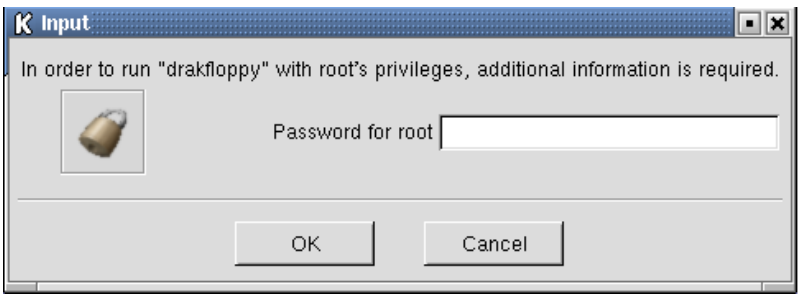

Figura 13-1. Ingrese su contraseña de `root`

Luego de eso, Ud. estará en frente de la ventana principal de *`drakfloppy`* (Figura 13-2). Si desea crear un disquete de arranque "predeterminado", es decir, uno que es igual al que resulta de experimentar el ejemplo de la última sección, sólo tiene que insertar un disquete en la disquetera apropiada, seleccionar esa disquetera desde la lista desplegable y presionar Aceptar.

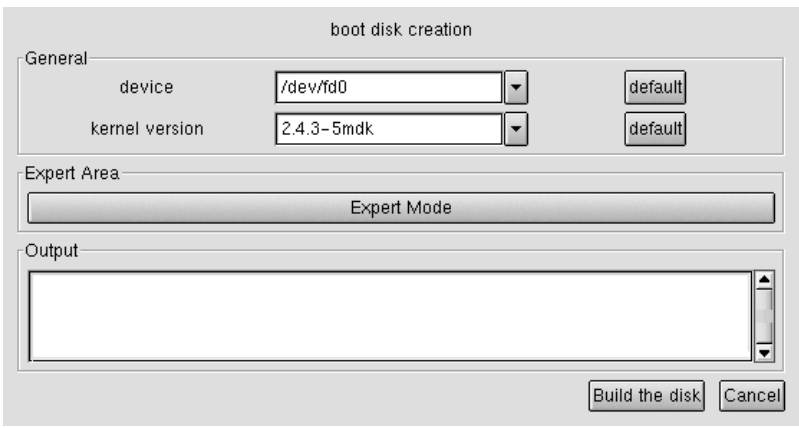

Figura 13-2. La ventana principal de `drakfloppy`

Si desea personalizar su disquete de arranque, tendrá que presionar el botón Mostrar modo experto y la ventana de *drakfloppy* cambiará como se muestra en Figura 13-3.

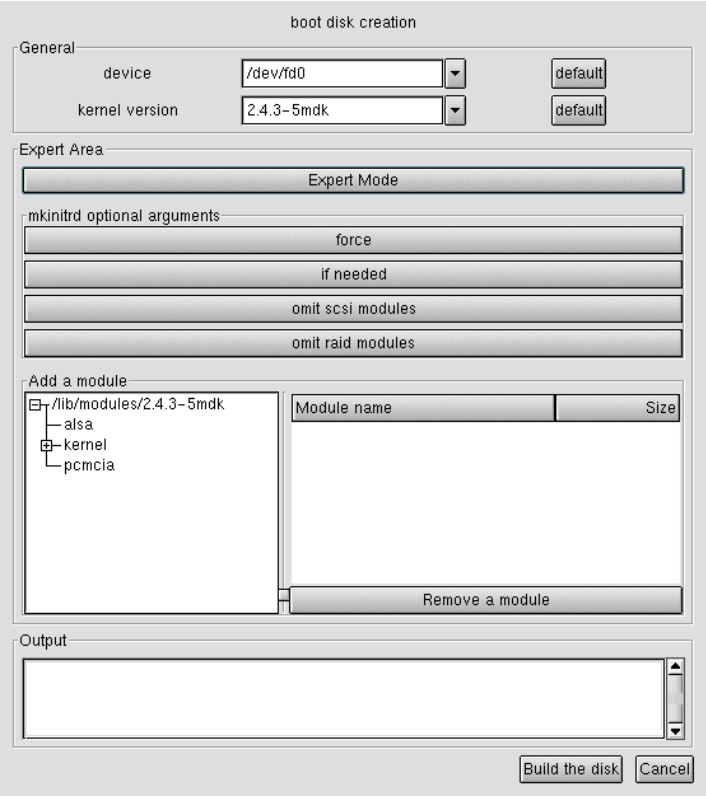

Figura 13-3. Haciendo un disquete de arranque personalizado

Para agregar algunos módulos tiene que presionar el botón Agregar y aparecerá la ventana que se muestra en Figura 13-3. En el ejemplo queremos usar el módulo de unidad de cinta IDE y pre-cargarlo. Las otras opciones se explican por sí mismas. Cuando termine de personalizar el disquete de arranque, presione Aceptar para crearlo.

13.2.3. Probando el disquete de arranque

Pruebe su disquete de arranque para asegurarse que **realmente funciona**. Hay pocas cosas más embarazosas que darse cuenta que el disquete no va a arrancar debido a errores en el disquete. Si el disquete arranca sin problemas entonces...

13.2.4. ¡Ya está!

¡Felicidades! Ya tiene la herramienta más importante para tratar de recuperar un sistema dañado: un disquete de arranque **:-)** Ahora, sigamos con algunas consideraciones importantes sobre la segunda herramienta más importante: las copias de respaldo.

13.3. Copia de respaldo

13.3.1. ¿Por qué hacer copia de respaldo?

Hacer copia de respaldo de su sistema es la **única** manera de poder repararlo si sufre un daño severo, o incluso si Ud. borrar por accidente algunos archivos importantes del sistema, o si alguien irrumpe en su sistema y borra intencionalmente algunos archivos. También debería hacer copia de respaldo de los datos que usa a diario (audio comprimido, imágenes, documentos de oficina, correo-e, libreta de direcciones, etc.) para estar seguro.

Debería hacer sus copias de respaldo usando un soporte apropiado y mantenerlas en un lugar seguro. Tal lugar debería estar fuera del lugar en el que Ud. trabaja usualmente, si es posible. Incluso puede tener dos copias de respaldo, uno en el lugar de trabajo, y otra fuera del mismo. En general, debería asegurarse que podrá recuperar dichas copias de respaldo si desea que todo esto realmente sirva para algo **:-)**

13.3.2. Preparando su sistema

Probablemente tiene todo lo que necesita ya instalado en su sistema, **y** un disquete de arranque (hizo uno, ¿cierto?). En realidad, puede hacer copias de respaldo usando sólo a `tar` y una herramienta de compresión tal como `gzip` o `bzip2`. Vea un ejemplo en *Ejemplo de copia de respaldo usando TAR*, página 115.

Como alternativa, puede usar programas de copia de respaldo especializados, tales como *`Taper`*, *`Time Navigator`*, etc.

13.3.3. Que incluir en la copia de respaldo

Bueno, esta puede ser la pregunta más difícil que cada administrador de sistemas se pregunta cuando llega la hora de hacer la copia de respaldo. La respuesta depende de cosas tales como: ¿sólo está respaldando sus datos personales, sus archivos de configuración, o todo su sistema? ¿Cuánto tiempo y/o espacio va a tomar? ¿Restaurará su copia de respaldo en la misma máquina/versión de sistema operativo, o en una diferente?

Debido a que esto es una guía de solución de problemas, trataremos de concentrarnos en hacer una copia de respaldo tal que nos permita restaurar rápidamente nuestro sistema al estado en el cual estaba antes que ocurra esa cosa terrible que lo inutilizó. Por supuesto, necesitará hacer copia de respaldo de sus datos personales si no desea perderlos, pero... esa es otra historia.

Como regla general, necesitará hacer copia de respaldo de los directorios siguientes: `/etc`, `/home`, `/root` y `/var`. Si hace una copia de respaldo completa de estos directorios, habrá guardado no sólo sus configuraciones, sino también sus datos (en caso que se esté preguntando dónde están sus datos, están en el directorio `/home/su_nombre_de_usuario/`). Por favor, tenga presente que esto puede tomar un tiempo **largo** en completarse, pero es la apuesta más segura.

Un esquema más sofisticado sería hacer copia de respaldo sólo de los archivos de configuración que han cambiado, dejando de lado los que no han cambiado. Esto llevaría más tiempo de “planificación”, pero llevará a tiempos de copia de respaldo más cortos (y también a tiempos de restauración más cortos) y a copias de respaldo que son “más fáciles” de portar de una máquina/versión de sistema operativo a otro.

Seguidamente, se le presentará una lista de los archivos a los cuales debería prestarles la mayor atención. Note que estas listas no son exhaustivas en absoluto, en especial si ha hecho un montón de cambios en su sistema[1]

En el directorio `/etc`:

`/etc/lilo.conf`

1. De cualquier manera, si ha hecho un montón de cambios, probablemente no necesitará estas listas **:-)**

Contiene la configuración del cargador de arranque *LILO*. Si, en vez de *LILO*, Ud. usa *GRUB*, los archivos a incluir en la copia de respaldo son los que están en el directorio `/boot/grub`.

`/etc/fstab`

Contiene la configuración de las tablas de partición de los discos y sus puntos de montaje asociados.

`/etc/modules.conf`

Contiene los módulos a cargar y sus parámetros de acuerdo al hardware de su sistema. Puede no ser de utilidad si se restaura en una máquina **muy** diferente, pero de todas formas puede proporcionar algunas pistas.

`/etc/isapnp.conf`

Contiene las configuraciones de `isapnp` si es que lo usa para configurar el hardware ISA *plug'n'play*.

> **Nota:** Con el núcleo 2.4.x puede no necesitar más este archivo, ya que el hardware *plug'n'play* se configura usando el sistema de archivos `DevFS`.

`/etc/X11/XF86Config`

Contiene las configuraciones de *X*. *X* es el núcleo gráfico de *GNU/Linux* y todos sus entornos de escritorio y administradores de ventanas.

`/etc/cups`

Contiene las configuraciones de *CUPS*. *CUPS* es el sistema de impresión predeterminado de **Linux-Mandrake**.

`/etc/printcap`

Si no utiliza *CUPS* y usa el sistema de impresión `lpr`, entonces tiene que incluir este archivo en lugar de `cups` para las configuraciones de su impresora.

`/etc/bashrc`

Configura al *shell Bash*, para todo el sistema.

`/etc/profile`

Configura el entorno del sistema y algunos programas que se ejecutan al iniciar el sistema.

`/etc/crontab`

Configura los jobs de cron a ejecutar periódicamente, por ejemplo, para las tareas de mantenimiento del sistema.

`/etc/rc.d/rc.local/*`

Configura los distintos niveles de ejecución del sistema. Usualmente, no necesitará hacer copia de respaldo de estos, pero si agregó algún nivel de ejecución personalizado o cambio uno de los predeterminados, necesitará incluirlos en la copia de respaldo.

`/etc/inittab`

Configura el nivel de ejecución predeterminado con el cual arranca su sistema.

`/etc/ssh`

Contiene las configuraciones de `ssh`. Si utiliza el acceso remoto seguro, es **sumamente** importante incluir este archivo.

Si tiene un servidor *web*, un servidor FTP, u otros servidores, también haga una copia de respaldo de sus respectivos archivos de configuración.

En el directorio `/root` y en el directorio personal de cada usuario `/home/nombre_de_usuario`, los directorios siguientes:

`~/.gnome/*`

Configuraciones para el entorno de escritorio *GNOME*.

`~/.kde/*`

Configuraciones para el entorno de escritorio *KDE*.

`~/.netscape/*`

Configuraciones para la familia de programas Netscape. Los marcadores de Navigator, los filtros de correo de Messenger, etc.

`~/nsmail/*`

Contiene **todos** sus mensajes de correo-e y de los grupos de noticias.(En caso que use Netscape). **Definitivamente** no quiere perder estos, ¿cierto?

`~/Mail/*`

Si usa *KMail* este directorio contiene **todos** sus mensajes de correo-e. **Definitivamente** no quiere perder estos, ¿cierto?

`~/.ssh/*`

Contiene las configuraciones personalizadas para el uso de `ssh`. Si utiliza a `ssh`, haga copia de respaldo de este...

Tampoco quisiera perder de vista a los archivos siguientes:

`~/.bash_profile`

Contiene las variables de entorno, los alias, y más configuraciones para el *shell Bash*.

`~/.bashrc`

Más configuraciones de *Bash*.

`~/.cshrc`

Contiene las variables de entorno, los alias, y más configuraciones para el *shell CSH*.

`~/.tcshrc`

Contiene las variables de entorno, los alias, y más configuraciones para el *shell tcsh*.

Por favor, note que no mencionamos todos y cada uno de los posibles archivos de configuración debido a que hubiésemos necesitado un libro entero sobre el tema **:-)** Por ejemplo, si Ud. no utiliza *Netscape* no necesita hacer copia de respaldo de los archivos y directorios relacionados con *Netscape*, si Ud. no utiliza `ssh` no necesita hacer copia de respaldo de las cosas relacionadas con `ssh`, y así sucesivamente.

En suma, haga copia de respaldo de todos los archivos de configuración de los programas que usa y de todos los archivos de configuración que ha modificado. También haga copia de seguridad de todos los archivos de datos personales (y de los usuarios de su sistema). No se va a arrepentir, confíe en mi **:-)**

13.3.4. ¿Dónde hacer copia de respaldo?

La otra gran pregunta a responder. Esto depende de cuanta información desea incluir en la copia de respaldo, cuan rápido desea hacer sus copias de respaldo, cuan fácil es el acceso al soporte de la copia de respaldo, y una larga lista de etcéteras.

En general, necesitará soportes que tengan a lo sumo tanta capacidad como la cantidad de información que desea incluir, y sean suficientemente rápidos como para que el proceso completo termine este siglo **:-)**

13.3.5. Soportes para copias de respaldo

Seguidamente, le proporcionaremos una descripción breve de las opciones de soportes de copia de respaldo disponibles. Estas varían en capacidad, confiabilidad, y velocidad. No se dan en orden particular alguno, simplemente como vienen a la mente **:-)** Por favor, note que el software que Ud. utiliza para hacer copia de respaldo puede o no soportarlos a todos.

Nota: Esta lista no pretende ser un análisis exhaustivo de los diferentes soportes de almacenamiento disponibles en el mercado. De hecho, algunas de las cosas que se escriben a continuación pueden cambiar en el futuro. Cosas tales como el tiempo de vida esperado del soporte fueron tomadas de los sitios *web* de los fabricantes y/o de la experiencia personal y de la comunidad. También, pueden haber muchos puntos de vista **personales** sobre temas tales como el precio o la velocidad, por ejemplo.

Disquete

Su capacidad va desde 1.44 MB[2]. Se pueden llevar de un lado a otro con facilidad pero, para las necesidades actuales, pueden no tener espacio suficiente. La mejor forma de llevar archivos pequeños. Lento. Barato. Disquetera estándar en virtualmente cada computadora que existe. Lectura/Escritura. El tiempo de vida esperado es de 4 o 5 años.

Nota: Por favor, tenga presente que los disquetes no son muy confiables.

Disquete LS120

Su capacidad es 120 MB. Dimensiones físicas idénticas al disquete pero con casi diez veces más la capacidad. No muy baratos. Necesita una disquetera especial pero esta disquetera también puede leer/escribir disquetes estándar. Puede ser un buen reemplazo para los disquetes pero su velocidad queda atrás de la de las unidades ZIP. Lectura/Escritura. El tiempo de vida esperado es más o menos el mismo que para los disquetes ZIP

Disquete ZIP

2. Bueno, en realidad se pueden formatear hasta 1.92 MB usando programas del tipo de *SuperFormat* y su disquetera estándar, pero esa es otra historia...

Su capacidad llega hasta 250 MB. Aunque no son tan delgados como los disquetes, también se pueden llevar de un lado a otro con facilidad, y son más adecuados para las necesidades de hoy día. Buen balance de características, aunque pueden ser un poco caros. Lectura/Escritura. El tiempo de vida esperado es de diez años para las unidades de 100 MB, tal vez sea mayor para las de 250 MB.

CD-R

Su capacidad llega hasta 700 MB estos días, aunque el estándar es 650 MB. Soporte muy barato y confiable. Hoy día muchos comentan que su capacidad no es suficiente, pero, hey, 650 MB me suenan bien **:-)**. Su característica más fuerte es que casi todas las computadoras de la Tierra tienen una unidad de CD-ROM, por lo tanto se pueden leer casi en cualquier lugar. Se pueden escribir sólo una vez. Se pueden leer tantas veces como lo desee (bueno, en realidad, tantas como pueda...). El tiempo de vida esperado es 20 años, tal vez más si se almacenan en un lugar seguro y no se leen muy seguido **:-)**

CD-RW

Las mismas consideraciones que con los CD-R, pero se pueden formatear y volver a escribir hasta 1000 veces. En general, es un soporte barato y confiable. El tiempo de vida esperado es 15 años, tal vez más si se si se almacenan en un lugar seguro y no se leen muy seguido **:-)**

Cinta

Su capacidad va desde 120 MB (¿alguien tiene unidades de cinta tan antiguas?) hasta varios gigabytes. Es un soporte caro y no muy confiable (hey, después de todo **son** cintas magnéticas). No obstante, su capacidad los hace ideal para hacer copias de respaldo de servidores y cosas por el estilo; si desea hacer copia de respaldo de todo su disco rígido en una sola pieza de soporte, la cinta es la única opción (por ahora, al menos...). Su desventaja más importante es que el acceso a la cinta es secuencial, y esto tiene un gran impacto en el desempeño, pero las unidades de cinta SCSI son lo suficientemente rápidas para las necesidades de hoy día y, hey, **tienen** muchos gigabytes de lugar para almacenar sus archivos. Lectura/Escritura. El tiempo de vida esperado es 30 años para las tecnologías de cinta nuevas.

Disco rígido

Ud. se debe estar preguntando: ¿Este tipo tomó whiskey en el desayuno? No, no tomé whiskey en el desayuno, es que hoy día, los precios de los discos han caído de forma tal que también pueden ser considerados seriamente como soporte para copias de respaldo. Son relativamente baratos, tienen amplio espacio (hasta 40 GB al momento de escribir este manual), y son muy confiables y el soporte más rápido de todos los que se presentan en esta lista. Si tiene un sistema portátil esto puede no ser una opción[3], pero en sus sistemas de escritorio agregar una segunda unidad de disco sólo para propósitos de copia de respaldo puede ser una buena opción. En realidad, puede no necesitar agregar una unidad de disco rígido nueva y hacer copias de respaldo en el único disco rígido que tiene; pero esto puede no ser una idea muy buena ya que no lo protegerá si se le rompe el disco rígido.

Otros soportes removibles

Existen otros soportes removibles (El *ORB* de Castlewood, y el *JAZ* de IOMEGA me vienen a la mente) que tienen un buen balance de precio/características y son adecuados para hacer copias de respaldo. Algunos incluso fueron publicitados como “reemplazos de su disco rígido” (*JAZ* por ejemplo) pero cuando se usan como discos rígidos pueden no durar mucho debido a restricciones de diseño (**No** son unidades de disco rígido). Sin embargo, en estos temas *Your Mileage May Vary* (Su Experiencia Puede Ser Distinta), simplemente asegúrese de elegir con sabiduría (con sentido común) de acuerdo a sus necesidades, y ... ¡buena suerte! **:-)**

3. Si tiene una portátil relativamente nueva, puede tener espacio para poner dentro una segunda unidad de disco rígido. También, usando USB puede conectar unidades de disco rígido USB externas.

Directorios remotos

Bueno, estrictamente hablando, estos pueden no ser considerados como “soporte”, pero mencionaremos algo sobre ellos porque son una buena opción para copia de respaldo si no tiene cientos de megabytes para incluir en la copia de respaldo.

Si su ISP le proporciona algo de espacio, Ud. puede usar ese espacio para colocar sus archivos junto con sus páginas *web* **:-)**. En la *web* puede encontrar muchas ofertas de servicios de almacenamiento remoto en línea. Si tiene una red con dos o más máquinas puede hacer copias de respaldo en alguna máquina “remota” en la red (por supuesto, otra máquina de la que Ud. está intentando hacer copia de respaldo...)

En realidad, el hecho de hacer copias de respaldo “remotas” puede ser un agujero de seguridad, entonces no mantenga en una copia de respaldo remota sus archivos ultra secretos ni sus archivos más importantes. Piense que, en caso de una falla severa del sistema, tal vez ni pueda conectarse a ese sitio remoto para recuperar los archivos...

Por favor, tenga presente que también puede combinar soportes de copia de respaldo de acuerdo a su estrategia para realizar las copias de respaldo, por ejemplo: cintas y CD-R, disco rígido y cintas, disco rígido y CD-R, etc.

13.3.6. ¿Cuándo hacer copia de respaldo?

Hay muchas políticas para las agendas de copia de seguridad. Aquí le presentaremos algunas. Por favor, tenga presente que esas no son obligatorias, ni son las mejores, ni son las únicas. Simplemente son guías que querría seguir al planificar su propia agenda de copia de seguridad.

Hay muchas estrategias de copia de respaldo allí afuera, ellas dependen del soporte que Ud. utilice, de cuan seguido cambian sus datos, y de cuan críticos son sus datos para Ud. o para su organización. Por ejemplo, una estrategia dice que debería hacer una copia de respaldo completa cada fin de semana, y una incremental (sólo las cosas que cambiaron) cada día; luego hacer una copia de respaldo completa cada mes y guardar la última en al menos dos lugares. Esta estrategia puede resultar ser útil para una empresa pero no para una computadora personal. Para sus copias de respaldo personales puede pensar en algo como esto: hacer una copia de respaldo semanal en su disco rígido y cada mes transferir esas copias a CD-R o cinta.

13.3.7. Ejemplo de copia de respaldo usando TAR

Seguidamente, le presentaremos un pequeño script de copia de respaldo que usa a `tar` para hacer una copia de respaldo completa de su directorio personal.

Aviso

Necesitará permiso de lectura sobre los archivos y directorios que va a incluir en la copia de respaldo, caso contrario la operación de copia de respaldo fallará.

```
#!/bin/bash

# Crear una copia de respaldo comprimida de su directorio personal en el archivo
# backup.tar.gz o backup.tar.bz2 dependiendo del esquema de compresión usado.

BACKUP_DIRS=$HOME

# Quitar comentario de la línea siguiente si desea comprimir con GZIP
#tar cvzf backup.tar.gz $BACKUP_DIRS
```

```
# Aquí comprimimos con BZIP2...
tar cvyf backup.tar.bz2 $BACKUP_DIRS
```

Como puede ver este es un script de copia de respaldo **muy** simple que sólo hace una copia de respaldo de su directorio personal y pone el resultado en el mismo directorio. Vamos a mejorarlo un poquito...

```
#!/bin/bash

# Crear una copia de respaldo comprimida de todos los directorios especificados
# y poner el archivo resultante en un directorio de nuestra preferencia.

BACKUP_DIRS="$HOME /etc /etc/rc.d"
BACKUP_FILENAME=‘fecha‘+%b%d%Y
BACKUP_DEST_DIR=$HOME

# Quitar comentario de la línea siguiente para usar GZIP, dejarlo para usar BZIP2
#tar cvzf $BACKUP_DEST_DIR/$BACKUP_FILENAME.tar.gz $BACKUP_DIRS

# Aquí usamos BZIP2 para comprimir...
# Comentar la línea siguiente para usar GZIP, quitar comentario para usar BZIP2
tar cvyf $BACKUP_DEST_DIR/$BACKUP_FILENAME.tar.bz2 $BACKUP_DIRS
```

Como puede ver en este último ejemplo, hemos añadido algunos directorios más a nuestra copia de respaldo, y hemos usado un esquema de nombres para agregar la fecha de la copia de seguridad al nombre del archivo resultante.

Por supuesto, puede mover el archivo `tar.bz2` o `tar.gz` resultante a cualquier soporte que desee. Incluso puede hacer copia de respaldo directamente sobre el soporte que desea si lo monta y cambia la variable `BACKUP_DEST_DIR` del script adecuadamente. Siéntase libre de mejorar este script y hacerlo tan flexible como desee **:-)**

Para restaurar las copias de respaldo hechas de esta forma, por favor mire en *Ejemplo de restauración usando TAR*, página 116

13.4. Restaurar

La restauración de la copia de seguridad depende del programa, soporte, y agenda que Ud. utilizó para hacerlo. Aquí no cubriremos todos los casos de restauración, sino que sólo mencionaremos que Ud. se tiene que asegurar de restaurar los archivos y/o directorios en los mismos lugares donde se encontraban cuando hizo la copia de respaldo para poder recuperar sus configuraciones, y sus archivos de datos.

13.4.1. Ejemplo de restauración usando TAR

Ahora, veremos un pequeño script para restaurar la copia de respaldo que hicimos con `tar` usando el script que se presentó antes en *Ejemplo de copia de respaldo usando TAR*, página 115

Aviso

Necesita permisos de escritura sobre los archivos y directorios que va a restaurar, de no tenerlos, la operación de restauración fallará.

```
#!/bin/bash

# Extraer una copia de respaldo comprimida de todos los directorios
```

```
# especificados poniendo los archivos en sus lugares originales.

BACKUP_SOURCE_DIR=$HOME
RESTORE_FILENAME=$1

# Quite el comentario de la línea siguiente si está comprimido con GZIP
#tar xvzf $BACKUP_SOURCE_DIR/$RESTORE_FILENAME

# Restaurar una copia de respaldo comprimida con BZIP2...
tar xvyf $BACKUP_SOURCE_DIR/$RESTORE_FILENAME
```

Como puede ver, este script es bastante simple. Todo lo que tenemos que hacer es pasarle el nombre del archivo de la copia de respaldo que deseamos restaurar como parámetro (sólo el nombre del archivo, no la ruta completa), y el script restaura los archivos de la copia de respaldo en sus ubicaciones originales.

13.5. Mi sistema se congela al arrancar

Puede ocurrir que su sistema se “cuelga” durante el arranque. De ser así, no entre en pánico, simplemente siga leyendo **:-)**

> **Nota:** Las secciones que siguen no se presentan en orden particular alguno.

13.5.1. El sistema se cuelga durante el arranque

Si su sistema se cuelga durante la etapa `Rebuilding RPM database` (Reconstruyendo la base de datos de los RPM) o `Finding module dependencies` (Encontrando las dependencias de los módulos), simplemente presione **CTRL**+**C**. Al hacerlo, el sistema va a saltear este paso y continuar arrancando. Una vez que arrancó, ejecute `rpm -rebuilddb` como `root` si la colgadura ocurrió durante la etapa `Rebuilding RPM database`. Si ocurrió durante la etapa `Finding module dependencies` es muy probable que haya pasado por una actualización del núcleo, pero no lo haya hecho adecuadamente. Verifique si los archivos en los directorios `/boot` y `/lib/modules` coinciden con la versión corriente del núcleo (es decir, tienen el mismo número de versión). Si no coinciden, por favor lea *Compilando e instalando núcleos nuevos*, página 129, para hallar una forma de solucionar esto.

Si el arranque se cuelga en `RAMDISK: Compressed image found at block 0` (RAMDISK: Se encontró imagen comprimida en el bloque 0) Ud. ha arruinado la imagen `initrd`. O bien trata de arrancar otra entrada de `lilo.conf` o arranca un sistema de emergencia y quita o cambia la sección `initrd=` en `/etc/lilo.conf`

13.5.2. Falla la verificación de sistema de archivos al arrancar

Si, por cualquier razón, no ha apagado su máquina de manera apropiada, el sistema ejecutará una verificación del sistema de archivos de rutina durante el arranque próximo. A veces puede fallar al hacer esto por sí solo y lo llevará a una consola. Ejecute `e2fsck -py [dispositivo]` donde `[dispositivo]` es el nombre de la partición sobre la cual falló la prueba automática. La opción `-p` le dice a `e2fsck` que haga todas las reparaciones necesarias sin preguntar, `-y` asume que la respuesta a todas las preguntas es `sí`. Cuando la fase de verificación y reparación termina, presione **CTRL**+**D** para dejar la consola de emergencia. El sistema volverá a arrancar.

Si este error se repite con regularidad, podrían haber sectores defectuosos en su disco. Ejecute `e2fsck -c [dispositivo]` para verificar esto. Este comando marcará automáticamente cualquier sector defectuoso y, por lo tanto, evitará que el sistema de archivos almacene datos en estos sectores. `e2fsck` verifica el sistema de archivos automáticamente sólo si no ha sido desmontado de manera apropiada durante el apagado previo del sistema; o si se ha alcanzado la cantidad máxima de montajes (`maximal mount count`). Para forzar una verificación, use la opción `-f`.

13.6. Volver a instalar el cargador de arranque

A veces Ud. comete un error y borra el sector de arranque maestro (MBR) de su disco, o algún programa que se porta mal lo hace, o incluso Ud. tiene arranque dual con *Windows* y se pesca un virus que lo hace. Entonces, Ud. dice, no voy a poder arrancar más mi máquina, ¿cierto? **¡equivocado! :-)** Hay muchas maneras de recuperar el cargador de arranque.

13.6.1. Usando un disquete de arranque

Esto no es sorpresa para nadie: Para recuperar su cargador de arranque Ud. **necesitará** un disquete de arranque. Sin un disquete de arranque de algún tipo Ud. puede estar completamente perdido[4]. Ha hecho un disquete de arranque, ¿cierto?

Bueno, simplemente ponga el disquete en la disquetera y vuelva a arrancar su computadora. Lo que haga luego, depende de si usa *LILO* o *GRUB*. Sin importar que cargador de arranque use, todos los comandos se deben ejecutar como `root`.

13.6.1.1. Con LILO

Si usa *LILO*, simplemente necesita ingresar lo siguiente en la línea de comandos: `/sbin/lilo`. Esto volverá a instalar a *LILO* en el sector de arranque de su disco y eso corregirá el problema.

13.6.1.2. Con GRUB

Si usa *GRUB* las cosas son un poquito diferentes que cuando usa *LILO*... pero no se asuste, estamos acá para ayudarlo **:-)**

> **Nota:** El ejemplo siguiente asumirá que está intentando instalar a *GRUB* en el MBR de su primer unidad de disco rígido IDE, y que el archivo `stage1` está en el directorio `/boot/grub/`.

Primero, invoque al shell de *GRUB* ejecutando el comando: `grub`. Una vez allí, ingrese el comando siguiente: `root (hd0,0)`; esto le dirá a *GRUB* que los archivos que necesita están en la primer partición (`0`) de su primer unidad de disco rígido (`hd0`). Luego ejecute el comando siguiente: `setup (hd0)`; esto instalará a *GRUB* en el MBR de su primer disco rígido. ¡Ya está!

También puede intentar usar `grub-install /dev/hda` para instalar a *GRUB* en el MBR de su primer disco rígido, pero el método que se describe arriba es el preferido.

13.6.1.3. ¡Terminó!

4. A menos que haga una copia de respaldo de su MBR, más sobre eso luego...

Bueno, esto es todo lo que hay que saber acerca de volver a instalar su cargador de arranque.

13.6.2. Reparando un Super-Bloque dañado

El super-bloque es el primer bloque de cada partición ext2fs. El mismo contiene datos importante acerca del sistema de archivos, como el tamaño, espacio libre, etc. (Es similar al método usado por las particiones FAT). Una partición con un super-bloque dañado no se puede montar. Afortunadamente, ext2fs mantiene varias copias de respaldo del super-bloque desparramadas sobre la partición.

Arranque su sistema con el disquete de arranque que creó antes (**Ha** creado uno, ¿cierto?). Por lo general, las copias de respaldo se ubican al comienzo de cada bloque de 8 KB (8192 bytes). Entonces, la próxima copia de respaldo está en el byte número 8193. Para restaurar el super-bloque a partir de esta copia, ejecute `e2fsck -b 8193 /dev/hda4`; cambie `hda4` para reflejar el nombre de su partición dañada. Si ocurre que ese bloque también está dañado, intente el siguiente en el byte número 16384, y así hasta que encuentra uno que sirva. Vuelva a arrancar su sistema para activar los cambios.

13.7. Niveles de ejecución

13.7.1. Descripción breve de que son los niveles de ejecución

Un nivel de ejecución es una configuración del software del sistema que permite que existan ciertos procesos. Para cada nivel de ejecución se definen los procesos permitidos en el archivo `/etc/inittab`. Hay ocho niveles de ejecución definidos: `0`, `1`, `2`, `3`, `4`, `5`, `6`, `S`. También puede crear su nivel de ejecución propio. Para una descripción más detallada sobre los niveles de ejecución, por favor refiérase a *Los archivos de arranque: init sysv*, página 91.

13.7.2. ¿Cómo me pueden ayudar los niveles de ejecución?

Arrancar en un nivel de ejecución diferente puede ayudarlo a resolver ciertos problemas, por ejemplo: suponga que ha hecho un cambio en su configuración de *X* tal que el sistema se ha vuelto inútil, y Ud. arranca en *X* **predeterminadamente**. De ser así, puede arrancar temporalmente en una consola, arreglar el error y volver a arrancar en *X*. Veamos como hacer eso.

De manera predeterminada *GNU/Linux* arranca o en el nivel de ejecución `3` (la consola), o en el nivel de ejecución `5` (*X*). El nivel de ejecución predeterminado se define en el archivo `/etc/inittab`. Busque una entrada como `id:3:initdefault:` (si su sistema arranca en una consola) o una como `id:5:initdefault:` (si su sistema arranca en *X*).

Si desea arrancar en un nivel de ejecución distinto al que está definido en `/etc/inittab`, tiene que definir a dicho nivel de ejecución en el *prompt* de arranque. Bajo *LILO*, teclee `linux init 3` para arrancar en la consola o `linux init 5` para arrancar en *X*. Bajo *GRUB*, presione la tecla **E** dos veces, agregue `init 3` para arrancar en la consola, o `init 5` para arrancar en *X*, presione la tecla **Intro** y luego la tecla **B** para arrancar.

13.8. Recuperando archivos borrados

En esta sección mencionamos algunas maneras de recuperar archivos y directorios borrados. Por favor, tenga presente que las herramientas de recuperación no son mágicas, y sólo funcionarán dependiendo de cuan recientemente borró el archivo que está intentando recuperar.

Puede estar pensando “Bueno, borré accidentalmente este archivo, ¿cómo puedo recuperarlo?”. No tema, hay algunos utilitarios diseñados para el sistema de archivos de *GNU/Linux*, ext2, que le permiten recuperar los archivos y directorios borrados. Sin embargo, estos utilitarios no recuperarán los archivos que Ud. borró hace unos meses, debido al uso del disco, el espacio marcado como “libre” se escribirá con otra cosa; entonces, la **mejor** manera de protegerse contra los borrados accidentales o no tan accidentales es hacer copias de respaldo como se describe arriba.

Bueno, sigamos con las herramientas para recuperar sus archivos borrados. Una de ellas es *Recover*. Es una herramienta “interactiva”. Si Ud. es el dueño orgulloso de un MandrakeLinux - Edición PowerPack Deluxe, ya tiene esta herramienta en el CD-ROM “contribs”. De no ser así, puede encontrarla en el sitio web de RPMFind (http://www.rpmfind.net). Vaya allí y transfiera el RPM. Una vez que tiene el RPM, tiene que instalarlo. Luego, ejecute `recover [opciones_del_comando]` y responda a las preguntas que le formula. Las preguntas son para configurar una franja de tiempo para buscar archivos y directorios borrados para minimizar el tiempo que lleva hacer la búsqueda.[5]

Una vez que la herramienta termina su búsqueda, le preguntará donde desea grabar los archivos y directorios recuperados. Escoja un directorio de su preferencia, y tendrá todos los archivos y directorios recuperados en el mismo. Note que no podrá recuperar los nombres de los archivos, sólo sus contenidos, pero puede inspeccionarlos o intentar cambiarles el nombre varias veces hasta que obtenga el nombre adecuado. Esto es mejor que nada **:-)**

13.9. Recuperando cuando se congela el sistema

Cuando su computadora está “congelada”, no responderá más a los comandos y los dispositivos de entrada como el teclado y el ratón parecen estar bloqueados. Este es un escenario del peor caso y puede significar que Ud. tiene un error muy severo ya sea en su configuración, su software o su hardware. Aquí le mostraremos como manejar esta situación molesta.

En caso que su sistema se congele, su prioridad máxima debería ser intentar apagarlo de manera apropiada. Asumamos que está bajo *X*, de ser así, intente estos pasos de manera consecutiva:

- Intente terminar el servidor *X* presionando simultáneamente **ALT**+ **CTRL**+ **BACKSPACE**.
- Intente cambiar a otra consola con **ALT**+**CTRL**+**F2**. Si tiene éxito, ingrese como `root` y ejecute el comando: `kill -15 $(pidof X)` o el comando `kill -9 $(pidof X)`, si el primer comando no surte efecto alguno. (Verifique con `top` para ver si *X* todavía se está ejecutando).
- Si es parte de una red local, intente conectarse a su máquina con `ssh` desde otra máquina. Es aconsejable conectarse con `ssh` en la máquina remota como un usuario no privilegiado y luego usar `su` para volverse `root`.
- Si el sistema no responde a alguno de estos pasos, tiene que ir a través de la secuencia “SysRq” (“System Request”, Pedido del Sistema). LA secuencia “SysRq” involucra presionar tres teclas a la vez, la tecla **ALT** izquierda, la tecla **SysRq** (marcada **PrintScreen** en teclados antiguos) y una tecla de alguna letra.
  - **ALT Izq.**+**SysRq**+**r** pone al teclado en modo “raw” (crudo). Ahora intente presionar **ALT**+ **CTRL**+ **BACKSPACE** otra vez, para terminar el servidor *X*. Si eso no funciona, continúe.

5. También puede buscar **todos** los archivos borrados, pero llevará más tiempo...

- **ALT Izq,**+**SysRq**+**s** intenta escribir en el disco todos los datos no guardados (“sincronizar” el disco).
- **ALT Izq.**+**SysRq**+**e** envía una señal de terminación a todos los procesos, excepto a `init`.
- **ALT Izq.**+**SysRq**+**i** envía una señal de matar a todos los procesos, excepto a `init`.
- **ALT Izq.**+**SysRq**+**u** intenta volver a montar todos los sistemas de archivos como de sólo lectura. Esto quita la “marca de sucio” y evitará una verificación del sistema de archivos al volver a arrancar.
- **ALT Izq.**+**SysRq**+**b** vuelve a arrancar el sistema. También podría presionar el botón de “reset” de su máquina.

  Nota: Recuerde que esto es una secuencia, es decir, Ud. tiene que presionar una combinación después de la otra en el orden correcto: c**R**udo, **S**incronizar, t**E**rminar, destru**I**r, **U**mount (desmontar), re**B**oot (volver a arrancar)[6]. Encontrará más acerca de esta característica en `/usr/src/linux/Documentation/sysrq.txt`.

- Si nada de lo anterior ayuda, cruce los dedos y presione el botón de “reset” en su máquina. Si tiene suerte, *GNU/Linux* sólo ejecutará una verificación del disco al momento del arranque.

Por todos los medios, intente averiguar las causas por las cuales su máquina se congela porque pueden causar un daño severo al sistema de archivos. También puede querer considerar el uso de `ReiserFS`, un sistema de archivos con registro que se incluye en **Linux-Mandrake** desde 7.0, el cual maneja tales fallas con mayor gracia. Sin embargo, para reemplazar a ext2fs con `ReiserFS` se necesita volver a formatear a sus particiones.

13.10. Terminando aplicaciones que no se portan bien

Bueno, esto no es tan difícil después de todo **:-)**. En realidad, no es común que necesite hacer esto pero en caso que lo necesite... Tiene varias formas de hacerlo. Puede hacerlo buscando el PID del programa que se rebeló y usar `kill` para terminarlo, o puede usar la herramienta `xkill` u otras herramientas gráficas como las que le muestran el árbol de procesos.

13.10.1. Desde la consola

La primer cosa a hacer para terminar un programa que se está portando mal es encontrar su PID, o identificador de proceso. Para hacerlo, ejecute el comando siguiente desde una consola: `ps aux | grep netscape`, suponiendo que *Netscape* es el programa rebelde. Obtendrá algo como lo siguiente:

```
Darth    3505  7.7 23.1 24816 15076 pts/2   Z    21:29   0:02 /usr/lib/netscape
```

Esto nos dice, entre otras cosas, que *Netscape* fue iniciado por el usuario Darth y tiene `3505` como PID.

Ahora que tenemos el PID del programa rebelde, podemos proceder a ejecutar el comando `kill` para terminarlo. Entonces, ejecutamos lo siguiente: `kill -9 3505`, ¡y ya está! *Netscape* será terminado. Note que esto **sólo** se debe usar cuando el programa deja de responder a su entrada. **No** lo use como la forma común para salir de las aplicaciones.

En realidad lo que hemos hecho fue enviar la señal `KILL` al proceso número `3505`. El comando `kill` acepta otras señales además de `KILL`, entonces Ud. puede tener un control mayor sobre sus procesos. Para más información, vea la página Man de `kill` y *Control de procesos*, página 61.

6. En inglés hay una frase mnemónica: “Raising Skinny Elephants Is Utterly Boring” (Criar elefantes delgados es terriblemente aburrido) **:-)**

13.10.2. Usando XKILL

Linux-Mandrake incluye una herramienta gráfica para terminar procesos: `xkill`. La misma es útil para terminar aplicaciones gráficas rebeldes con un sólo clic. Todo lo que tiene que hacer es hacer clic sobre el icono de `xkill` en el escritorio y, cuando el puntero de su ratón cambia a una calavera con dos huesos cruzados debajo, muévalo sobre la ventana de la aplicación que desea terminar y haga clic sobre dicha ventana. ¡Ya está!

> **Sugerencia:** También puede acceder a `xkill` con el atajo de teclado siguiente: **CTRL**+**ALT**+**ESC** (al menos desde *KDE*).

13.10.3. Usando otras herramientas gráficas

También puede usar una de las herramientas gráficas de estado de los procesos (como *KPM*, *KSySGuard*, y *GTOP* por nombrar algunas) que le permiten apuntar al nombre del proceso y con un clic solo enviar una señal a ese proceso o simplemente terminar ese proceso.

13.11. Herramientas de solución de problemas específicas de Mandrake

Bueno, en realidad cada herramienta de administración (*Control Center*, *RpmDrake*, *DrakGW*, *DrakNet*, *MandrakeUpdate*, etc.) es una herramienta potencial de solución de problemas **:-)** Puede usar todas estas herramientas para revertir los cambios en la configuración, añadir o quitar programas, actualizar su sistema con los últimos arreglos de **MandrakeSoft**, etc.

13.12. Pensamientos finales

Bien, como ha visto hay muchas más formas de recuperarse de una emergencia además de volver a instalar todo el sistema **:-)**[7]. Seguro, necesita algo de experiencia para aplicar las técnicas que se describen en este capítulo, pero con un poquito de práctica puede adquirirla. Sin embargo, esperamos que nunca necesite realmente dominar estas técnicas **:-)** ... aunque no hace mal conocerlas. Esperamos que las instrucciones y ejemplos dados sean de utilidad cuando esté necesitado. ¡Buena suerte recuperándose de una emergencia!

7. La forma común de arreglar las cosas en ciertos otros sistemas operativos...

Capítulo 14. Algunas palabras acerca del Núcleo 2.4

Probablemente el cambio del núcleo 2.2 al núcleo 2.4 pasará inadvertido para la mayoría de los usuarios finales de *Linux*. Sin embargo es una pequeña revolución para la gente involucrada en la programación de *Linux* y para los usuarios avanzados. Por lo tanto, no podíamos resistir el hecho de agregar un capítulo nuevo a nuestro **Manual de Referencia** para presentar al usuario las mejoras más importantes así como también algunos trucos que pueden resultar útiles para problemas específicos.

14.1. Linux 2.4 - ¿qué es lo que tiene para Ud.?

Linux-Mandrake 8.0 es la primer versión de LM basada en el núcleo 2.4.x.

¡Genial! ¡Uno, Dos, Hurra! ¡Es lo que todos estábamos esperando! Y así sucesivamente...

Bueno, hemos absuelto esta parte, por lo tanto vamos a verlo desde otro lugar:

¿Una versión nueva del núcleo? ¿Y qué? ¿Por qué tendría que importarme?

La respuesta a esta pregunta es diferente para los usuarios hogareños, de oficina, ISPs, usuarios de portátiles, gente interesada en *Linux* embebido...

Además, la mayoría de las mejoras en el núcleo 2.4 se ocultarán detrás de algún programa de configuración gráfico, y por lo tanto, serán “invisibles”. Por ejemplo, la combinación de *Resource management* (Administración de recursos), y *Device filesystem* (Sistema de archivos de dispositivos) eventualmente llevarán a un soporte más robusto de Enchufar-y-funcionar (*Plug-and-work*) en todas las distribuciones *GNU/Linux*. Cuando esto ocurra mucha gente celebrará los programas de configuración gráficos, mientras que el “héroe verdadero” permanecerá anónimo.

Otros cambios pueden no afectarlo en absoluto, al menos no por ahora: los procesadores de 2 GHz todavía no están disponibles, 64 GB de memoria RAM todavía son algo caros, los archivos más grandes que 2 GB todavía son escasos, y la posibilidad de tener *PC*s hogareñas con 16 tarjetas *Ethernet*, 10 controladoras IDE, etc. también es bastante baja. Las grandes mejoras en el manejo de los dispositivos de bloques (dispositivo de E/S “raw” (crudo), Administrador de Volúmenes Lógicos (LVM), mejor soporte para RAID) serán importantes para las versiones nuevas de nuestros productos “corporativos” y sus relacionados, pero es poco probable que afecten a los usuarios hogareños, los usuarios de escritorio de oficina o los servidores para oficinas pequeñas.

Dicho esto, abajo tiene los cambios que afectarán a muchos usuarios de **Linux-Mandrake**:

14.1.1. Redes: ¡seguridad, estabilidad, y velocidad!

Linux siempre ha tenido soporte de redes realmente bueno por razones obvias: *Linux* nació en la Internet, y la mayor parte del desarrollo también se hace por medio de la Internet. No obstante, la capa de redes se ha vuelto a escribir por completo, y el núcleo 2.4 trae algunos cambios casi “revolucionarios”:

- Soporte para redes a través de USB, Firewire y *Ethernet* de 1 GHz. (¿cuándo será que finalmente agreguen señales de humo?)
- Mejor soporte para PPP: la capa PPP por RDSI se combinó con PPP de la capa de dispositivos serie, PLIP mejorado, y la combinación del soporte de PPPoE en el núcleo.
- La funcionalidad de cortafuegos y enmascarado de IP ha vuelto a ser re-escrita por completo. Netfilter (iptables) aporta mejoras de seguridad importantes en comparación con *iphains* usado en 2.2. Además, iptables permite una configuración mucho más fácil y potente de la traducción de direcciones de red (NAT),

proxies transparentes, y re-direccionamiento (Piense acerca de *clustering* de servidores con carga balanceada, es decir, ¡reemplazar a un servidor Web con cuatro de forma transparente!). Resumiendo, Jay (el director de nuestro equipo de seguridad y autor de Bastille Linux (http://www.bastille-linux.org/)) ha estado soñando con iptables por años. (http://securityportal.com/cover/coverstory20010122.printerfriendly.html)

Además, 2.4 ofrece optimizaciones para las peculiaridades que se encuentran en las capas de red de otros sistemas operativos, un manejo mejor de los sockets de red, un soporte mejorado para sistemas multi-procesador, e incluso una estabilidad mejor a la que ya estamos acostumbrados (¿es posible esto?).

14.1.2. PCMCIA, USB, Fire-wire y soporte para ISA PnP

En los núcleos 2.2, PCMCIA e ISA-PnP han sido soportados por paquetes externos, y los núcleos 2.2 en las distribuciones LM 7.x incluían soporte de USB extraído de núcleos superiores. Esto era realmente útil, pero la calidad de los paquetes externos y los parches nunca pueden igualar a la calidad del núcleo de *Linux* oficial. *Linux* 2.4 finalmente incluye soporte nativo para estos dispositivos. Combinado con un sistema central de "administración de recursos", el soporte para PCMCIA, USB, FireWire e ISA-PnP debería mejorar drásticamente en el futuro.

14.1.3. Vídeo y multimedia.

Un soporte mejor para el "frame-buffer", y la adición de DRI (*Direct Rendering Infrastructure*, Infraestructura de Render Directa) hacen a *Linux* 2.4 aun más atractivo para los juegos (¡aceleración de vídeo!) y los dispositivos embebidos. Es posible que la muerte de SVGAlib llegue a su *PC* más adelante este año. **:-)**

Además, el soporte para tarjetas de sonido, y para tarjetas de TV/radio ha estado mejorando sin parar durante el ciclo de desarrollo de 2.4. Otra vez, la mayoría de estos cambios ya habían sido extraídos de núcleos superiores a los núcleos 2.2 en las distribuciones LM 7.x, y los usuarios "mimados" de LM pueden no notar una diferencia importante en este área.

14.1.4. Sistemas de archivo

El cambio más notorio en el soporte de sistemas de archivos locales es la inclusión de sistemas de archivos con registro ReiserFS en el núcleo *Linux* oficial. Otra vez, ReiserFS ya estaba presente en LM 7.x, pero estaba marcado "experimental". Por el lado de las "redes", *Linux* 2.4 finalmente soporta NFSv3, y también se mejoró el soporte para SMB. Desafortunadamente, no se ha integrado sistema de archivos distribuido alguno en el árbol del núcleo oficial, y el soporte para NTFS todavía está marcado como "experimental".

14.1.5. Una nota sobre los Parches

Estos no son el tipo de parches que uno se pone cuando se lastima, sino un parche que Ud. pone en su pileta de natación cuando encuentra un hueco. Para los núcleos tiene un significado más amplio, generalmente es algo de código adicional que le da al núcleo funcionalidad extra. Aquí damos la lista de parches más importantes agregada por **MandrakeSoft** al núcleo estándar mantenido por Linus Torvalds.

Alsa

La Arquitectura de Sonido de *Linux* Avanzada es básicamente un controlador de sonido. Funciona sobre el soporte básico de sonido de núcleo, y mejora la calidad del sonido. También proporciona una interfaz común para las aplicaciones de forma tal que el soporte de sonido pueda ser más homogéneo bajo *Linux*

FreeS/WAN

FreeS/WAN de *Linux* es una implementación de IPSEC & IKE para *Linux*.

IPSEC significa *Internet Protocol SECurity* (Seguridad del Protocolo de Internet). Usa cifrado fuerte para proporcionar servicios tanto de autenticación como de cifrado. La autenticación asegura que los paquetes son del remitente correcto y no han sido alterados durante el tránsito. El cifrado evita que se lea el contenido de los paquetes sin autorización.

Estos servicios le permiten construir túneles seguros a través de redes poco seguras. La máquina portal de IPSEC cifra todo lo que pasa por la red poco segura y la máquina portal del otro lado del túnel lo descifra. El resultado es un Red Privada Virtual (*Virtual Private Network*, o VPN). Esta es una red que es en efecto privada aunque incluye máquinas en varios sitios diferentes conectadas por medio de la Internet poco segura.

Lm_sensors

Este es un parche del proyecto Lm_sensors para monitorear el estado del hardware de los sistemas *Linux* que contienen hardware de monitoreo del estado del hardware como por ejemplo el LM78 y el LM75 (y muchos otros).

Pcmcia-CS

Card Services para *Linux* es un paquete de soporte completo de PCMCIA. El mismo incluye un conjunto de módulos del núcleo cargables que implementan una versión de la API de los PCMCIA 2.1 Card Services, un conjunto de controladores de cliente para tarjetas específicas, y un demonio administrador de tarjetas que puede responder a eventos de inserción y remoción de tarjetas, cargando y descargando los controladores bajo demanda. Este soporta el "intercambio en caliente" (*hot swapping*) de las tarjetas PCMCIA, de forma tal que las tarjetas se pueden insertar y expulsar en cualquier momento. El paquete actual soporta muchas tarjetas *Ethernet*, módems y tarjetas serie, varios adaptadores SCSI, y algunas tarjetas de memoria SRAM y *FLASH*. Todas las controladoras PCMCIA comunes están soportadas, de forma tal que debería correr prácticamente en todas las portátiles capaces de ejecutar *Linux*.

Controladores misceláneos

Los controladores para soportar partes de hardware muy recientes, que por ahora no han sido incluidos en el núcleo. También hay algunos controladores especiales tales como *SuperMount* que permite el montaje automático de los dispositivos removibles en el momento que se accede a los mismos.

Finalmente debemos remarcar que **Linux-Mandrake** viene no sólo con un núcleo sino con varios de ellos:

kernel22-*

Estos son los núcleos construidos a partir de la revisión 2.2.

kernel

El núcleo estándar instalado de manera predeterminada en todas las máquinas estándar.

kernel-smp

Este núcleo está compilado especialmente para soportar máquinas con procesadores múltiples. Si al momento de instalar el sistema se detecta más de un procesador se instala este núcleo de manera predeterminada.

kernel-linus

El núcleo estándar tal cual está mantenido y distribuido por Linus Torvalds, sin parche alguno.

14.1.6. ¿Qué diablos es `kapm-idled`?

Si ya echó un vistazo a la carga de su CPU, debe haber notado una tarea denominada `kapm-idled` que ocupa la mayor cantidad de los recursos de la CPU. Y se debe haber preguntado "¡Qué es esa cosa rara horrible que se está comiendo a mi CPU!"

Bueno, de hecho la misma es sólo una tarea que contabiliza el tiempo durante el cual la CPU está ociosa. De aquí el nombre. Esta se usa para las necesidades de el Manejo Avanzado de los Procesos (*`Advanced Process Management`*), y de hecho protege a su CPU en vez de hacerle daño.

14.1.7. El núcleo funciona en todos lados, pero...

Al final, permítanos mencionar una característica que puede no parecer importante a los poseedores de *`PC`*: El núcleo de *`Linux`* ha sido portado a casi todo tipo de hardware que existe hoy día, comenzando con dispositivos pequeños con *`Linux`* embebido, hasta las supercomputadoras IBM. Las distribuciones **Linux-Mandrake** no lo usan por completo, y nunca lo harán, debido a que se deben portar otros programas, o al menos recompilar y probar para arquitecturas diferentes también. En especial, los programas de instalación y configuración del hardware DrakeX se tienen que adaptar para cada plataforma nueva lo cual cuesta tiempo y dinero.

Sin embargo, el hecho que el núcleo ha sido portado a tantas arquitecturas significa que el código del mismo ha sido revisado una y otra vez, ¡lo cual resulta en núcleos mucho más robustos para todas las arquitecturas! Además, esto también significa que la distribución **Linux-Mandrake** (LM) se puede portar rápidamente a otras arquitecturas, siempre y cuando haya un mercado para esto. Esto ya ha sido probado cuando se portó a *`Sparc`* (Corporate Server) y a *`Alpha`* (LM 7.1), y portar a otras arquitecturas es sólo un tema de interés comercial. A pesar que LM 8.0 actualmente sólo existe en versión para *`PC`* (compatible con *`Intel`*), el soporte para i64 estará disponible tan pronto como esta plataforma esté disponible en el mercado, y es probable que también se porte a *`PowerPC`* si esta plataforma continúa siendo tan exitosa como lo fue el año pasado.

14.2. Consejos útiles para sacarle más jugo a su núcleo

14.2.1. initrd: Controladores necesarios al momento de arrancar

El `initrd` (boot loader Initialized Ram Disk) contiene módulos controladores necesarios para que se levante el sistema. Ellos son por lo general controladores para acceso a disco (IDE SCSI RAID).

Generalmente no necesita cambiar su `initrd`, pero en caso que tenga una configuración especial de hardware (por ejemplo RAID) o cambios en el tipo de discos, puede necesitar actualizar su `initrd`. Todo está hecho de forma tal que los requisitos especiales se detectarán y manejarán automáticamente. Pero en caso que esto falle o para casos muy especiales Ud. puede necesitar poner las manos en la masa.

Si su partición raíz (`/`) está bajo reiserfs es particularmente importante que su `initrd` contenga los módulos necesarios para acceder a esa partición.

14.2.1.1. El procedimiento

Básicamente, esto es lo que está pasando: *`LILO`* o *`GRUB`* tienen una opción `initrd=/boot/initrd.img` que especifica la ubicación del `initrd` a cargar en memoria. El cargador de arranque carga en memoria tanto el núcleo como el `initrd`. Luego, el núcleo puede usar los módulos contenidos en el `initrd` para sus necesidades.

Para crear el `initrd`, hay un programa especial denominado `mkinitrd` (vea `man mkinitrd`). El mismo toma los módulos especificados en `/etc/modules.conf` más algunos opcionales especificados en su línea de comandos y los empaca en un archivo dirigido al cargador de arranque.

14.2.1.2. Un ejemplo

Imaginemos que Ud. desea cambiar su sistema a una matriz RAID. Para poder arrancar, el núcleo necesita acceder a la matriz para montar los sistemas de archivos. Por lo tanto, Ud. tendrá que poner el módulo `aacraid` (por ejemplo) dentro de `initrd`. Todos los comandos siguientes se deben ejecutar como `root`.

1. asegúrese que la línea

   ```
   initrd=initrd.img
   ```

   está en el archivo de configuración del cargador de arranque.

2. cree e instale el `initrd`:

   ```
   $ /sbin/mkinitrd -with=aacraid /boot/initrd-2.4-mdk.img 2.4.2-7mdk
   mke2fs 1.19, 13-Jul-2000 for EXT2 FS 0.5b, 95/08/09
   $ ln -sf initrd-2.4-mdk.img /boot/initrd.img
   $
   ```

   El número de revisión "`2.4.2-7mdk`" debe corresponderse con el núcleo instalado en ese momento y en cualquier caso debe corresponderse con el directorio presente en `/lib/modules/`.

3. vuelva a arrancar **:-)**

14.2.2. Dispositivos USB

Básicamente, Ud. acaba de comprar un ratón USB nuevo y quiere hacerlo funcionar en su sistema. Esto es muy fácil. De hecho generalmente no tiene nada que hacer ya que el ratón USB se detecta y se instala

automáticamente. Sin embargo, puede ocurrir que falle la detección automática. De ser así, proceda de la manera siguiente.

Primero, edite el archivo `/etc/sysconfig/usb`, el mismo debería contener algo así:

```
USB=yes
MOUSE=no
KEYBOARD=no
STORAGE=no
VISOR=no
```

Antes que nada asegúrese que la primera línea es `USB=yes` y simplemente cambie `MOUSE=no` por `MOUSE=yes`. Luego ejecute `mousedrake` y elija allí el ratón apropiado.

14.2.3. Dispositivos de mano basados en USB

Esta sección será de interés para quienes posean dispositivos tales como el *Visor*, el cual sincroniza sus datos con una *PC* por medio de un vínculo USB.

El procedimiento es casi el mismo que para los ratones USB. Cambie el archivo `/etc/sysconfig/usb` de forma tal de obtener `VISOR=yes`.

Hecho esto, simplemente necesita decirle al sistema que su *Visor* está conectado en el puerto USB:

```
$ ln -s /dev/usb/ttyUSB0 /dev/visor
```

¡y eso es todo! su *Visor* está listo para la sincronización.

> **Nota:** Dependiendo de su configuración puede tener que usar `/dev/usb/ttyUSB1` en vez de `/dev/usb/ttyUSB0`.

Puede consultar el mini-COMO sobre el Handspring-Visor con Linux (http://www.calvin.edu/~rvbijl39/visor/beta/t1.html) para más información.

Capítulo 15. Compilando e instalando núcleos nuevos

Junto con la noción del montaje de los sistemas de archivos y la compilación de los fuentes, la compilación del núcleo es, sin lugar a dudas, el tema que causa la mayor cantidad de problemas a los principiantes. Generalmente no es necesario compilar un núcleo nuevo, ya que los núcleos que instala **Linux-Mandrake** contienen soporte para un número significativo de dispositivos (de hecho, más dispositivos que los que Ud. necesitará o pensará jamás), así como también un montón de *patches*, etc. Pero...

Podría ser, que quiera hacerlo, por el sólo hecho de ver "que es lo que hace". Además de hacer que su *PC* y su cafetera funcionen un poco más de lo normal, no mucho. Las razones por las cuales Ud. debería querer volver a compilar su propio núcleo varían desde desactivar una opción, hasta volver a construir un núcleo experimental completamente nuevo. De todas formas, el objetivo de este capítulo es que su cafetera debería seguir funcionando luego de la compilación.

También hay otros motivos válidos para volver a compilar el núcleo. Por ejemplo, Ud. leyó que el núcleo que está usando tiene un ***bug*** que afecta a la seguridad, un bug que se corrige en una versión más reciente; o bien, que un núcleo nuevo incluye soporte para un dispositivo que necesita. Por supuesto que en estos casos Ud. tiene la opción de esperar a las actualizaciones de los binarios, pero actualizar los fuentes del núcleo y volver a compilar el núcleo nuevo por Ud. mismo, es una solución más rápida.

Haga lo que haga, consiga algo de café.

15.1. Dónde conseguir los fuentes del núcleo

El sitio principal de alojamiento de los fuentes del núcleo es ftp.kernel.org, pero tiene una gran cantidad de sitios de réplica, todos se llaman ftp.xx.kernel.org, donde xx representa el código ISO del país. Para Argentina, este código es ar, por lo tanto el sitio de réplica preferido será la máquina *ftp.ar.kernel.org*. Ejemplos de otros códigos ISO de países de habla hispana son: es, España; mx, Méjico; ve, Venezuela; ec, Ecuador. Luego del anuncio oficial de la disponibilidad del núcleo, debería dejar pasar dos horas para que todos los sitios de réplica se actualicen.

En todos estos servidores FTP, los fuentes del núcleo están en el directorio `/pub/linux/kernel`. Luego, debe dirigirse al directorio con la serie que le interesa: seguramente será la `v2.4`[1]. No hay nada que le impida probar los núcleos de la versión 2.5[2], pero recuerde que estos son núcleos experimentales. El archivo que contiene los fuentes del núcleo se denomina `linux-<version.del.núcleo>.tar.gz`, por ejemplo `linux-2.4.2.tar.gz`.

También existen los *patches* (correcciones o parches) para aplicar a los fuentes del núcleo para actualizarlo incrementalmente: de esta manera, si ya tiene los fuentes del núcleo versión 2.4.2 y los quiere actualizar a 2.4.4, no necesita transferir todos los fuentes, sino que simplemente puede transferir los ***patches*** `patch-2.4.3.gz` y `patch-2.4.4.gz`. Como regla general, esta es una idea buena, ya que los fuentes actualmente ocupan más de 24 MB.

15.2. Extrayendo los fuentes, corrigiendo el núcleo (si es necesario)

1. Al momento de escribir este manual

2. O cualquier otra versión impar

Los fuentes del núcleo deberían ponerse en `/usr/src`. Por lo que debería ir a este directorio y luego extraer los fuentes allí:

```
$ cd /usr/src
$ mv linux linux.old
$ tar xzf /ruta/a/linux-2.4.2.tar.gz
```

El comando `mv linux linux.old` es necesario: esto es debido a que Ud. ya podría tener los fuentes de otra versión del núcleo. Este comando le asegurará que no escribirá sobre ellos. Una vez que el archivo se descompactó, tiene un sub-directorio `linux` con los fuentes del núcleo nuevo.

Ahora, los *patches*. Asumiremos que quiere "patchear" (o corregir) de la versión 2.4.2 a la 2.4.4 y que ha transferido los *patches* necesarios para hacer esto: debe dirigirse al directorio `linux` creado recientemente, luego aplique los *patches*:

```
$ cd linux
$ gzip -dc /ruta/al/patch-2.4.3.gz | patch -p1
$ gzip -dc /ruta/al/patch-2.4.4.gz | patch -p1
$ cd ..
```

Generalmente, para pasar de una versión 2.4.x a una versión 2.4.y es necesario que Ud. aplique todos los *patches* numerados 2.4.x+1, 2.4.x+2, ..., 2.4.y en orden. Para "revertir" desde 2.4.y hasta 2.4.x, repita exactamente el mismo proceso pero aplicando los *patches* en orden inverso con la opción `-R` desde `patch` (`R` significa Revertir). Entonces, para regresar del núcleo 2.4.4 al núcleo 2.4.2, Ud. haría lo siguiente:

```
$ gzip -dc /ruta/al/patch-2.4.4.gz | patch -p1 -R
$ gzip -dc /ruta/al/patch-2.4.3.gz | patch -p1 -R
```

Luego, en pos de la claridad (y para que Ud. sepa donde está), puede cambiarle el nombre a `linux` para reflejar la versión del núcleo y crear un vínculo simbólico:

```
$ mv linux linux-2.4.4
$ ln -s linux-2.4.4 linux
```

Ahora es tiempo de pasar a la configuración. Para esto debe estar en el directorio fuente:

```
$ cd linux
```

15.3. Configurando el núcleo

Primero, un pequeño truco: Ud. puede, si lo desea, personalizar la versión de su núcleo. La versión de su núcleo está determinada por las primeras cuatro líneas del archivo `Makefile`:

```
$ head -4 Makefile
VERSION = 2
PATCHLEVEL = 4
SUBLEVEL = 4
```

```
EXTRAVERSION =
```

Más adelante en el archivo `Makefile`, puede ver que la versión del núcleo se construye como:

```
KERNELRELEASE=$(VERSION).$(PATCHLEVEL).$(SUBLEVEL)$(EXTRAVERSION)
```

Todo lo que tiene que hacer es modificar uno de estos campos para cambiar su versión. Preferentemente, Ud. sólo cambiará `EXTRAVERSION`. Digamos que la configura como `-pepe`, por ejemplo. Entonces, la nueva versión de su núcleo será 2.4.4-pepe.

Ahora, sigamos con la configuración. Puede elegir entre:

- `make xconfig` para una interfaz gráfica,
- `make menuconfig` para una interfaz basada en `ncurses`, o
- `make config` para la interfaz más rudimentaria, línea por línea, sección por sección.

Mayormente la configuración del núcleo no está internacionalizada, todo está en inglés. Trataremos de ir sección por sección aunque puede omitir secciones e ir a la sección que desee si está usando `menuconfig` o `xconfig`. Las opciones pueden contestarse con **y** por Yes (funcionalidad incorporada, compilada en el núcleo), **m** por Module (funcionalidad compilada como un módulo), o **n** por No (no incluir en el núcleo).

Tanto `make xconfig` como `make menuconfig` tienen las opciones acomodadas por grupos jerárquicos. Por ejemplo, `Processor family` (Familia del procesador) va bajo `Processor type and features` (Características y tipo del procesador).

Para `xconfig`, el botón Main Menu es para volver al menú principal cuando se está en un grupo jerárquico, Next es para ir al siguiente grupo de opciones, y Prev es para volver al grupo anterior. Para `menuconfig`, use la tecla **Intro** para seleccionar una sección, y cambie el estado de las opciones con **y**, **m**, o **n** o, de lo contrario, presione la tecla **Intro** y elija sus opciones de entre las opciones múltiples disponibles. Con Exit saldrá de una sección y de la configuración si es que se encuentra en el menú principal. Y también está la ayuda con Help.

Aquí no vamos a enumerar todas las opciones, ya que hay varios cientos. Es más, si ha llegado a este capítulo, probablemente sepa lo que está haciendo. Por lo tanto, lo dejamos navegar por la configuración del núcleo y habilitar/deshabilitar las opciones que Ud. crea apropiadas. Sin embargo, hay algunos consejos para que Ud. no se encuentre con un núcleo que no pueda usar:

1. A menos que use un ramdisk inicial, ¡**nunca** compile los controladores necesarios para montar su sistema de archivos raíz (controladores de hardware y controladores de sistemas de archivos) como módulos! Y, si Ud. usa un ramdisk inicial, diga `Y` al soporte para ext2fs, ya que este es el sistema de archivos que usan los ramdisks.
2. Si tiene grabadoras de CD IDE en su sistema, compile el soporte para las unidades de CD-ROM IDE como un módulo; haga lo mismo con el soporte genérico para SCSI y con la emulación IDE SCSI. Si dice `Y` al soporte para CD-ROM IDE, no podrá usar sus grabadoras de CD como grabadoras, aunque todavía las podrá usar como unidades de CD-ROM normales.
3. Si tiene tarjetas de red en su sistema, compile los controladores de las mismas como módulos. De esta forma, Ud. puede definir qué tarjeta será la primera, cual la segunda, y así sucesivamente, poniendo alias apropiados en el archivo `/etc/modules.conf`. Si compila los controladores dentro del núcleo, el orden en el que se cargarán dependerá del orden de vinculación, el cual puede diferir de lo que Ud. desea.
4. Y finalmente: si no sabe lo que abarca una opción, ¡lea la ayuda! Si el texto de ayuda no logra inspirarlo, simplemente deje la opción como estaba.

También puede consultar el archivo `/usr/src/linux/Documentation/Configure.help` el cual le da el texto de ayuda para cada opción en orden de aparición. También encontrará en el encabezado del mismo vínculos a varias traducciones.

¡Y voilà! La configuración por fin está terminada. Guarde su configuración y salga.

15.4. Guardando y volviendo a usar los archivos de configuración de su núcleo

La configuración de núcleo se guarda en el archivo `/usr/src/linux/.config`. Es altamente recomendable que haga una copia de respaldo de este archivo, por ejemplo en el directorio `/root`, de tal forma que tenga la posibilidad no sólo de volver a usarlo más tarde, sino también guardar configuraciones para distintos núcleos, ya que sólo es cuestión de dar nombres diferentes a los archivos de configuración.

Una posibilidad es nombrar a los archivos de configuración basándose en la versión del núcleo. Digamos que Ud. modificó la versión de su núcleo como se mostró en *Configurando el núcleo*, página 130, entonces puede hacer:

```
$ cp .config /root/config-2.4.4-pepe
```

Si, por ejemplo, decide actualizar a 2.4.6, podrá volver a usar este archivo, ya que las diferencias de configuración entre estos dos núcleos serán muy pequeñas. Simplemente use la copia de respaldo:

```
$ cp /root/config-2.4.4-pepe .config
```

Pero el hecho de volver a copiar el archivo, no significa que el núcleo esté listo para volver a ser compilado. Tiene que volver a invocar a `make menuconfig` (o lo que sea que elija usar), ya que este proceso crea y/o modifica algunos archivos necesarios para poder compilar exitosamente.

Sin embargo, además del hecho molesto de volver a pasar por todos los menús, puede perderse alguna opción nueva interesante. Puede evitar esto usando `make oldconfig`. Esto tiene dos ventajas:

1. es rápido,
2. si aparece una opción nueva en el núcleo y no estaba presente en su archivo de configuración, el proceso se frenará y esperará a que Ud. ingrese su elección.

Luego, tiempo de compilar.

15.5. Compilar el núcleo y los módulos, instalar los módulos

Una pequeña nota antes de comenzar: si está volviendo a compilar un núcleo con una versión idéntica al que ya está presente en su sistema, primero debe borrar los módulos antiguos que se encuentran en su sistema. Por ejemplo, si está recompilando 2.4.10, debe borrar el directorio `/lib/modules/2.4.10`.

La compilación del núcleo y de los módulos, y la posterior instalación de los módulos se hace con una sola línea:

```
make dep bzImage modules modules_install
```

Un poco de vocabulario: `dep`, `bzImage`, etc., así como también `oldconfig` y otros que hemos usado arriba, se denominan **objetivos**. Si especifica varios objetivos para `make` como se muestra arriba, se ejecutarán los mismos en orden de aparición. Pero si uno de los objetivos falla, `make` no continuará más[3].

Veamos los objetivos diferentes y qué es lo que hacen:

- `dep`: esto computa las dependencias entre los distintos archivos fuente. Es necesario hacerlo cada vez que cambia su configuración, de otra forma puede ser que algunos archivos no se construyan y la compilación fallará.
- `bzImage`: esto construye el núcleo. Note que este objetivo sólo es válido para los procesadores *Intel*, y también lo es `zImage`. La diferencia entre `bzImage` y `zImage` es que el primero generará un núcleo que se cargará en la parte alta de la memoria. Este objetivo también genera el archivo `System.map` para este núcleo. Más adelante veremos para que se usa este archivo.
- `modules`: como su nombre (en inglés) lo indica, este objetivo generará los módulos para el núcleo que construyó recién. Si ha elegido no tener módulos, este objetivo no hará cosa alguna.
- `modules_install`: esto instalará los módulos. De manera predeterminada, los módulos se instalarán en el directorio `/lib/modules/<versión-del-núcleo>`. Este objetivo también computa las dependencias de los módulos (a diferencia de los núcleos de la versión 2.2.x).

Ahora todo está compilado y los módulos están instalados. Pero eso no es suficiente: también necesita instalar el núcleo en un lugar donde su cargador de arranque (ya sea *LILO* o *GRUB*) lo pueda encontrar. De esto se trata la sección siguiente.

15.6. Instalando el núcleo nuevo

El núcleo se encuentra en `arch/i386/boot/bzImage` (o `arch/i386/boot/zImage` si ha elegido `make zImage`). El directorio estándar en el cual se instalan los núcleos es `/boot`. También necesita copiar el archivo `System.map` para asegurarse que algunos programas (por ejemplo, `top`) funcionarán correctamente. Otra vez, consejos para una buena práctica: nombre a dichos archivos en base a la versión del núcleo. Asumamos que la versión de su núcleo es 2.4.4-pepe. La secuencia de comandos que Ud. tendrá que teclear es:

```
$ cp arch/i386/boot/bzImage /boot/vmlinux-2.4.4-pepe
$ cp System.map /boot/System.map-2.4.4-pepe
```

Después de esto, tiene que decirle al cargador de arranque acerca de su núcleo. Hay dos posibilidades: *GRUB* o *LILO*. Note que **Linux-Mandrake** ahora se proporciona con *GRUB* como cargador de arranque predeterminado.

15.6.1. Actualizando a grub

Obviamente, ¡retenga la posibilidad de iniciar su núcleo corriente! La forma más fácil de actualizar a *GRUB* es usar *DrakBoot* (vea el capítulo drakboot: cambiar su configuración de arranque de la **Guía del Usuario**). Alternativamente, Ud. puede editar manualmente el archivo de configuración de la manera siguiente.

Debe editar el archivo `/boot/grub/menu.lst`. Así es como luce un archivo `menu.lst` típico, luego de que Ud. ha instalado su distribución **Linux-Mandrake** y antes de la modificación:

```
timeout 5
```

3. En este caso, si falla, significa que hay un error en el núcleo... De ser así, por favor ¡háganoslo saber!

```
color black/cyan yellow/cyan
i18n (hd0,4)/boot/grub/messages
keytable (hd0,4)/boot/es-latin1.klt
default 0

title linux
kernel (hd0,4)/boot/vmlinuz root=/dev/hda5

title failsafe
kernel (hd0,4)/boot/vmlinuz root=/dev/hda5  failsafe

title Windows
root (hd0,0)
makeactive
chainloader  +1

title floppy
root (fd0)
chainloader +1
```

Este archivo está compuesto por dos partes: el encabezado con las opciones en común (las primeras cinco líneas), y las imágenes (o entradas), cada una correspondiente a un núcleo de *GNU/Linux* diferente o al sistema operativo (SO) que sea. `timeout 5` define el tiempo del que Ud. dispone para elegir una opción en particular antes que *GRUB* lance la predeterminada. Dicha opción predeterminada está definida por `default 0`, lo cual significa que la primera sección definida es la predeterminada. La línea `color` define los colores del menú; la línea `i18n` define donde se debe leer el mensaje de bienvenida: `(hd0,4)` significa que el mismo está ubicado en la cuarta partición del primer disco rígido. La opción `keytable (hd0,4)/boot/es-latin1.klt` le dice a *GRUB* el tipo de teclado usado cuando Ud. le suministra argumentos al momento del arranque. En este ejemplo es una distribución de teclado Español. Si no hay entrada alguna de este tipo, se asume un teclado qwerty común. Todos los `hd(x,y)` que ve se refieren a la partición número `y` del disco número `x` como lo ve el *BIOS*

Luego viene la sección de las diferentes imágenes. Aquí tenemos cuatro de ellas: `linux`, `failsafe`, `windows`, y `floppy`.

- La sección `linux` comienza por decirle *GRUB* el núcleo que se tiene que cargar (`kernel hd(0,4)/boot/vmlinuz`), seguido por las opciones a pasar a dicho núcleo. En este caso, `root=/dev/hda5` le dirá al núcleo que el sistema de archivos raíz está ubicado en `/dev/hda5`. De hecho, `/dev/hda5` es el equivalente de `hd(0,4)` de *GRUB*, pero nada le impide al núcleo de estar en una partición diferente que el sistema de archivos raíz.
- La sección `failsafe` se parece muchísimo a la anterior, a diferencia que nosotros le pasaremos un argumento al núcleo (`failsafe`) lo cual le instruye que se ponga en el modo “single” o “rescue”.
- La sección `windows` le indica a *GRUB* que simplemente cargue el sector de arranque de la primera partición, el cual probablemente contiene un sector de arranque de *Windows*.
- La sección `floppy`, simplemente arranca a su sistema desde un disquete en la primera disquetera, cualquiera sea el sistema que esté instalado sobre el mismo. Puede ser un disquete de arranque de *Windows*, o incluso un núcleo de *GNU/Linux* en un disquete.

> **Nota:** Dependiendo del nivel de seguridad que use en su sistema, algunas de las entradas descriptas aquí pueden no estar presentes en su archivo.

Ahora, vayamos al grano. Necesitamos agregar otra sección para decirle a *GRUB* acerca de nuestro núcleo nuevo. En este ejemplo, la misma se ubicará al comienzo de las otras entradas, pero nada le prohíbe ponerla en otro lugar:

```
title pepe
kernel (hd0,4)/boot/vmlinuz-2.4.4-pepe root=/dev/hda5
```

¡No se olvide de adaptar el archivo a su configuración! El sistema de archivos raíz de *GNU/Linux* aquí está en `/dev/hda5` pero bien podría estar en otro lugar de su sistema.

Y eso es todo. A diferencia de *LILO*, como veremos debajo, no queda cosa por hacer. Simplemente vuelva a iniciar su computadora y aparecerá la entrada nueva que definió recién. Sólo tiene que seleccionarla del menú y arrancará su núcleo nuevo.

Si Ud. compiló su núcleo con el *framebuffer*, probablemente querrá usarlo: en este caso, debe agregar una directiva al núcleo que le dice cual es la resolución en la que Ud. quiere arrancar. La lista de modos está disponible en el archivo `/usr/src/linux/Documentation/fb/vesafb.txt` (¡sólo en el caso del framebuffer VESA! De lo contrario, debe referirse al archivo correspondiente). Para el modo 800x600 en 32 bits[4], el número de modo es `0x315`, por lo cual Ud. debe agregar la directiva:

```
vga=0x315
```

y ahora su entrada se parece a lo siguiente:

```
title pepe
kernel (hd0,4)/boot/vmlinuz-2.4.4-pepe root=/dev/hda5 vga=0x315
```

Para mayor información, por favor consulte las páginas Info acerca de *GRUB* (`info grub`).

15.6.2. Actualizando a LILO

La manera más simple de actualizar a *LILO* es usar *DrakBoot* (ver drakboot: cambiar su configuración de arranque en la **Guía del Usuario**). Alternativamente, Ud. puede editar manualmente el archivo de configuración de la manera siguiente.

El archivo de configuración de *LILO* es `/etc/lilo.conf`. Un archivo `lilo.conf` típico luce así:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
vga=normal
default=linux
keytable=/boot/es-latin1.klt
lba32
prompt
timeout=50
message=/boot/message
image=/boot/vmlinuz-2.4.9-19mdk
        label=linux
        root=/dev/hda1
        read-only
other=/dev/hda2
        label=dos
```

4. 8 bits significa 2^8 colores, es decir 256; 16 bits significa 2^{16} colores, es decir 64k, es decir 65536; en 24 bits así como en 32 bits, el color se codifica en 24 bits, es decir 2^{24} colores posibles, en otras palabras exactamente 16M, o un poco más de 16 millones.

```
        table=/dev/hda
```

Un archivo `lilo.conf` consiste de una sección principal, seguida de una sección para arrancar cada sistema operativo. En el ejemplo anterior, la sección principal se compone de las directivas siguientes:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
vga=normal
default=linux
keytable=/boot/es-latin1.klt
lba32
prompt
timeout=50
message=/boot/message
```

La directiva `boot=` le dice a *LILO* donde instalar su sector de arranque; en este caso, es en el MBR (*Master Boot Record*, Registro Principal de Arranque) del primer disco rígido IDE. Si quiere hacer un disquete de arranque de *LILO*, simplemente reemplace `/dev/hda` con `/dev/fd0`. La directiva `prompt` le pide a *LILO* que muestre el menú al arrancar. Como se configura un tiempo de espera, *LILO* iniciará la imagen predeterminada después de 5 segundos (`timeout=50`). Si remueve la directiva `timeout`, *LILO* esperará hasta que Ud. haya tecleado algo.

Luego viene una sección `linux`:

```
image=/boot/vmlinuz-2.4.9-19mdk
        label=linux
        root=/dev/hda1
        read-only
```

Una sección para arrancar un núcleo de *GNU/Linux* siempre comienza con una directiva `image=`, seguida de la ruta completa a un núcleo *GNU/Linux* válido. Como cualquier sección, contiene una directiva `label=` como identificador único. La directiva `root=` le dice a *LILO* qué partición contiene el sistema de archivos raíz para este núcleo. El suyo puede ser distinto al ejemplo... La directiva `read-only` le ordena a *LILO* a montar este sistema de archivos raíz como sólo de lectura al arrancar: si esta directiva no está presente, recibirá un mensaje de advertencia.

Luego viene la sección *Windows*:

```
other=/dev/hda2
        label=dos
        table=/dev/hda
```

De hecho, *LILO* usa una sección que empiece con `other=` para arrancar cualquier sistema operativo que no sea *GNU/Linux*: el argumento de esta directiva es la ubicación del sector de arranque de dicho sistema operativo, y, en este caso, es un sistema *Windows*. Para encontrar el sector de arranque, ubicado al comienzo de la partición que alberga a este otro sistema, *GNU/Linux* también necesita saber la ubicación de la tabla de particiones que le permitirá ubicar la partición en cuestión, esto es lo que hace la directiva `table=`. La directiva `label=` identifica al sistema, como en la sección `linux` de arriba.

Ahora, es momento de agregar una sección para nuestro núcleo nuevo. Puede poner esta sección en cualquier lugar debajo de la sección principal, pero no “dentro” de otra sección. La misma lucirá así:

```
image=/boot/vmlinux-2.4.4-pep
        label=foo
        root=/dev/hda1
        read-only
```

Por supuesto, ¡Ud. debe adaptarlo a su configuración! Tomamos una situación diferente a la presentada con anterioridad para *GRUB* adrede...

Si compiló su núcleo con el *framebuffer*, debe referirse al párrafo de arriba correspondiente a *GRUB*, ahora la diferencia es que la opción está sola en una línea nueva:

```
vga=0x315
```

Entonces, así luce su `lilo.conf` luego de la modificación, decorado con algunos comentarios adicionales (todas las líneas que comienzan con `#`), que serán ignorados por *LILO*:

```
#
# Sección principal
#
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
# Al arrancar, queremos VGA normal. El framebuffer cambiará resoluciones por
# sí solo si lo usamos:
vga=normal
# Nuestro mensaje de arranque...
message=/boot/message
# Qué arrancar predeterminadamente. Pongamos nuestro núcleo nuevo aquí:
default=pepe
# Mostrar el prompt...
prompt
# ... esperar 5 segundos
timeout=50
#
# Nuestro núcleo nuevo: imagen predeterminada
#
image=/boot/vmlinux-2.4.4-foo
        label=pepe
        root=/dev/hda1
        read-only
# Si se usa el framebuffer VESA:
        vga=0x315
#
# El núcleo original
#
image=/boot/vmlinuz-2.4.2-19mdk
        label=linux
        root=/dev/hda1
        read-only
#
# Sección Windows
#
other=/dev/hda2
        label=dos
        table=/dev/hda
```

Esto bien podría ser como lucirá su archivo `lilo.conf`... pero recuerde, nuevamente, de adaptar el archivo de acuerdo con su configuración.

Ahora que se ha modificado adecuadamente el archivo, y a diferencia de *GRUB* que no lo necesita, debe indicarle a *LILO* que cambie el sector de arranque:

```
$ lilo
Added pepe *
Added linux
Added dos
$
```

De esta forma, Ud. puede compilar tantos núcleos como quiera, agregando tantas secciones como sea necesario. Ahora, sólo resta reiniciar la máquina para probar su núcleo nuevo.

Capítulo 16. Compilando e instalando software libre

Frecuentemente se me pregunta como instalar software libre desde los fuentes. Compilar software uno mismo es relativamente fácil debido a que la mayoría de los pasos a seguir son los mismos sin importar cual es el software a instalar. El propósito de este documento es guiar al principiante paso a paso y explicarle el significado de cada movimiento. Presumimos que el lector tiene un conocimiento mínimo de *Unix* (del tipo de `ls` o `mkdir` por ejemplo).

Esta guía no es más que una guía, no es un manual de referencia. Es por esto que al final se dan varios vínculos para poder responder las preguntas que queden. Probablemente se pueda mejorar esta guía, por lo que recibiremos con agradecimiento cualquier comentario o corrección sobre su contenido.

16.1. Introducción

Lo que hace la diferencia entre el software libre y el software propietario es el acceso al código fuente del software libre.[1]. Eso significa que el software libre se distribuye como archivados de archivos fuente. Esto puede resultarle poco familiar a los principiantes, porque los usuarios de software libre deben compilar ellos mismos los fuentes antes de poder usar el software.

Hay versiones compiladas de la mayoría del software libre existente. El usuario apurado no tiene más que instalar estos binarios pre-compilados. Sin embargo, algún software libre no se distribuye bajo esta forma, o las versiones más recientes todavía no se distribuyen en forma binaria. Más aun, si usa un sistema operativo exótico o una arquitectura exótica, un montón de software no va a estar ya compilado para Ud. Es más, compilar el software uno mismo permite conservar sólo las opciones interesantes o extender las funcionalidades del mismo agregando extensiones para obtener un software que responde perfectamente a sus necesidades.

16.1.1. Requisitos

Para compilar software, necesitará:

- una computadora con un sistema operativo funcionando,
- conocimiento general del sistema operativo que Ud. usa,
- algo de espacio en su disco rígido,
- un compilador (generalmente para el lenguaje *C*) y un archivador (`tar`),
- algo de comer (en los casos difíciles, puede durar un tiempo largo). Un verdadero hacker come pizza, no tornillos.
- algo de beber (por las mismas razones). Un verdadero hacker bebe cola con gas – por la cafeína.
- el número de teléfono de su “gurú” tecnológico, ese que recompila el núcleo todas las semanas,
- pero por sobre todo, paciencia, ¡y mucha!.

Compilar desde el código fuente generalmente no presenta muchos problemas, pero si Ud. no está acostumbrado, el menor tropiezo lo puede poner en una posición dificultosa o puede hacer que desista. El propósito de este documento es precisamente mostrarle como escapar de tal situación.

---

1. En realidad esto no es cierto, porque algunos software propietarios también ofrecen su código fuente. Pero, al contrario de lo que ocurre con el software libre, el usuario final no puede usar el código fuente de la manera que él desee.

16.1.2. Compilación

16.1.2.1. Los principios

Para poder traducir un código fuente a un archivo binario, es necesario efectuar una ***compilación***. Esta se hace generalmente sobre programas escritos en lenguaje *C* o *C++* (que son los lenguajes más usados en la comunidad de software libre, notablemente en el mundo *Unix*). Ciertos software libres están escritos en lenguajes que no necesitan compilación (por ejemplo *Perl* o el *shell*), pero aún así, estos necesitan algo de configuración.

Lógicamente, la compilación *C* está hecha por un compilador *C* que por lo general es `gcc`, el compilador libre escrito por el proyecto GNU (http://www.gnu.org/). La compilación de todo un paquete de software es una tarea compleja, que pasa por la compilación satisfactoria de archivos fuente diferentes (para el programador es más fácil poner las diferentes partes de su trabajo en archivos separados, por varios motivos). Para hacer más fácil esta tarea, estas operaciones repetitivas son efectuadas por un utilitario denominado `make`.

16.1.2.2. Las cuatro fases de la compilación

Para comprender bien como funciona la compilación (y por lo tanto, poder resolver posibles problemas), uno tiene que conocer sus cuatro fases. Su objetivo es convertir poco a poco un texto escrito en un lenguaje comprensible por un ser humano entrenado (por ejemplo, el lenguaje *C*), a un lenguaje comprensible por una máquina (o por un ser humano **muy** entrenado, aunque sólo en casos raros). `gcc` ejecuta cuatro programas uno tras otro, cada uno de los cuales se encarga de una etapa:

1. `cpp`: la primera etapa consiste en reemplazar las directivas (***pre-procesadores***) por instrucciones *C* puras. Típicamente, esto significa insertar un archivo de encabezado o *header* (`#include`), o definir una función macro (`#define`). Al final de esta fase, se genera código *C* puro.
2. *cc1*: esta fase consiste en convertir el *C* en ***lenguaje ensamblador***. El código generado depende de la arquitectura de destino.
3. `as`: esta fase consiste en generar el ***código objeto*** (o código binario) a partir del lenguaje ensamblador. Al final de esta fase, se generará un archivo con extensión `.o`.
4. *ld*: la última fase (la "edición de vínculos", en inglés *linkage*) ensambla (o ***vincula***) todos los archivos objeto (`.o`) y las bibliotecas asociadas, y produce un archivo ejecutable.

16.1.3. La estructura de una distribución

Una distribución de software libre correctamente estructurado siempre tiene la misma organización:

- Un archivo denominado `INSTALL`, que describe el proceso de instalación.
- Un archivo denominado `README`, que contiene información general relacionada con el programa (descripción breve, autor, la URL desde donde se puede bajar, documentación relacionada, vínculos útiles, etc). Si falta el archivo `INSTALL`, generalmente el archivo `README` contiene una descripción breve del procedimiento de instalación.
- Un archivo denominado `COPYING`, que contiene la licencia o describe las condiciones de distribución del software. A veces lo reemplaza un archivo denominado `LICENCE`.
- Un archivo denominado `CONTRIB` o `CREDITS` que contiene una lista de las personas relacionadas con el software (participación activa, comentarios pertinentes, programas de terceros, etc.).
- Un archivo denominado `CHANGES` (o, con menor frecuencia, `NEWS`, que contiene las mejoras y las correcciones de los *bugs* (errores en el software).

- Un archivo denominado `Makefile` (ver la sección *make*, página 147), que permite compilar el software (es un archivo que necesita `make`). Generalmente este archivo no existe al principio y se genera durante el proceso de configuración.
- Bastante seguido, un archivo `configure` o `Imakefile`, que permitirá generar un archivo `Makefile` nuevo.
- Un directorio que contendrá los fuentes, y donde generalmente se almacena el archivo binario al final de la compilación. Por lo general, este directorio se denomina `src`,
- Un directorio que contiene la documentación relacionada con el programa (generalmente en formato `man` o en formato *Texinfo*), cuyo nombre es, por lo general, `doc`,
- Eventualmente, un directorio que contiene los datos propios del programa (típicamente, los archivos de configuración, los ejemplos de los datos producidos, o archivos de recursos).

16.2. Descompresión

16.2.1. Archivado `tar.gz`

La norma[2] de compresión bajo los sistemas *Unix* es el formato `gzip`, desarrollado por el proyecto ***GNU***, y considerado como una de las mejores herramientas de compresión general.

Comúnmente se asocia `gzip` a un utilitario denominado `tar`. `tar` es un sobreviviente de los tiempos prehistóricos, cuando los informáticos almacenaban sus datos en cintas magnéticas. Hoy día, los disquetes y los CD-ROM han reemplazado a las cintas[3], pero todavía se usa `tar` para crear archivados. Por ejemplo, se pueden agregar todos los archivos de un directorio a un solo archivado. Luego, este archivado se puede comprimir fácilmente con `gzip`.

Esta es la razón por la cual muchos software libres están disponibles como archivados `tar`, comprimidos con `gzip`. Por lo tanto, sus extensiones son `.tar.gz` (o también, su forma abreviada `.tgz`).

16.2.2. Utilización GNU TAR

Para descomprimir este archivado, Ud. puede utilizar `gzip` y `tar`. Pero la versión GNU de `tar` (`gtar`) le permite utilizar `gzip` ***"al vuelo"***, y descomprimir un archivado de manera transparente (y sin necesitar el espacio extra en el disco).

La utilización de `tar` sigue este formato:

```
tar <archivo opciones> <.tar.gz archivo> [archivos]
```

La opción `<archivos>` no es obligatoria. Si se omite, se procesará todo el archivado. Si Ud. quiere extraer el contenido de un archivado `.tar.gz`, no necesita especificar este argumento.

Por ejemplo:

```
$ tar xvfz guile-1.3.tar.gz
-rw-r--r-- 442/1002      10555 1998-10-20 07:31 guile-1.3/Makefile.in
```

2. Cada vez más, se está usando un programa nuevo, denominado `bzip2`, más eficiente sobre los archivos de texto (aunque necesite más poder computacional y más memoria) Ver, más adelante, la sección *bzip2*, página 142 que trata específicamente con esto.

3. Aunque en algunos servidores con mucho volumen de información todavía se siguen usando las cintas magnéticas

```
-rw-rw-rw- 442/1002       6668 1998-10-20 06:59 guile-1.3/README
-rw-rw-rw- 442/1002       2283 1998-02-01 22:05 guile-1.3/AUTHORS
-rw-rw-rw- 442/1002      17989 1997-05-27 00:36 guile-1.3/COPYING
-rw-rw-rw- 442/1002      28545 1998-10-20 07:05 guile-1.3/ChangeLog
-rw-rw-rw- 442/1002       9364 1997-10-25 08:34 guile-1.3/INSTALL
-rw-rw-rw- 442/1002       1223 1998-10-20 06:34 guile-1.3/Makefile.am
-rw-rw-rw- 442/1002      98432 1998-10-20 07:30 guile-1.3/NEWS
-rw-rw-rw- 442/1002       1388 1998-10-20 06:19 guile-1.3/THANKS
-rw-rw-rw- 442/1002       1151 1998-08-16 21:45 guile-1.3/TODO
...
```

Entre las opciones de `tar` se encuentran las siguientes:

- `v` permite que `tar` sea “verboso”. Esto significa que mostrará en pantalla todos los archivos que encuentre en el archivado. Si se omite esta opción, el procesamiento será silencioso.
- `f` esta es una opción obligatoria. Sin ella, `tar` intenta usar una cinta magnética en vez de un archivado de archivo (es decir, el dispositivo `/dev/rmt0`).
- `z` permite manipular un archivado comprimido con `gzip` (con el nombre de archivo con sufijo `.gz`). Si Ud. olvida esta opción, `tar` producirá un error. A la inversa, no deberá utilizar esta opción si Ud. está frente a un archivado no comprimido.

`tar` permite efectuar varias acciones diferentes sobre un archivado (extracción, lectura, creación, adición ...). Una opción permite especificar la acción a usar:

- `x`: esta opción le permite extraer los archivos de un archivado.
- `t`: esta opción lista el contenido de un archivado.
- `c`: esta opción le permite crear un archivado, esto implica destruir su contenido actual. Por ejemplo, la puede usar para hacer una copia de seguridad de sus archivos personales.
- `r`: esta opción permite adicionar archivos al final del archivado. No se puede usar con un archivado comprimido.

16.2.3. bzip2

Un formato de compresión denominado `bzip2` ha comenzado a reemplazar a `gzip` en el uso general. `bzip2` produce archivos más cortos que los que produce `gzip`, pero todavía no es una norma de hecho. Sólo se pueden encontrar los archivados con la extensión `.tar.bz2` desde hace poco tiempo.

En lo que al comando `tar` respecta, `bzip2` se usa como `gzip`. La única cosa que hay que hacer es reemplazar la opción `z` por la opción `y`. Por ejemplo:

```
$ tar xvfy pepe.tar.bz2
```

Algunas distribuciones usan o usaban la opción `I` en su lugar:

```
$ tar xvfI pepe.tar.bz2
```

Otra posibilidad (que parece ser más portable, ¡pero es más larga de teclear!):

```
$ tar --use-compress-program=bzip2 -xvf pepe.tar.bz2
```

`bzip2` debe estar instalado e incluido en su variable de entorno `PATH` antes de poder correr `tar`.

16.2.4. ¡Simplemente hágalo!

16.2.4.1. La forma más fácil

Ahora está listo para descomprimir el archivado, no se olvide de hacerlo como administrador (`root`). Ud. tendrá que hacer cosas que un usuario no privilegiado no puede hacer, e incluso si puede hacer algunas como tal, es más fácil ser `root` durante toda la operación.

El primer paso es que Ud. se dirija al directorio `/usr/local/src`, y copie el archivado allí. De esta manera, siempre podrá encontrar el archivado en caso de perder el software instalado, ya que tendrá un lugar común donde buscar. Si no tiene mucho espacio en su disco rígido, haga una copia de seguridad del archivado en el medio habitual que Ud. usa para esto (disquetes, disquetes ZIP, CD-ROM, cinta magnética, etc.) una vez que instaló el software. También puede borrar el archivado pero antes de hacerlo, asegúrese de que lo pueda encontrar en la *web* cuando lo necesite.

Normalmente, la descompresión de un archivado `tar` debería crear un directorio nuevo (puede verificar esto antes de descomprimir con la opción `t`). Luego, cámbiese a ese directorio, ahora está listo para seguir adelante.

16.2.4.2. La manera más segura

Los sistemas *Unix* (por ejemplo, *GNU/Linux* y *FreeBSD*) suelen ser sistemas seguros. Esto significa que los usuarios no privilegiados no pueden hacer operaciones que puedan poner en peligro al sistema (por ejemplo, formatear el disco rígido) ni alterar los archivos de los demás usuarios. En la práctica y en particular, esto hace al sistema inmune frente a los virus.

Por otra parte, `root` puede hacer lo que se le antoje, incluso correr un programa malicioso (como, por ejemplo, un virus o un Troyano). El disponer del código fuente es una especie de garantía de seguridad frente a los virus[4]. Es decir, al tener el código fuente y si Ud. está lo suficientemente entrenado en el lenguaje de programación en el cual se programó el software, Ud. puede ver el código y deducir "qué, cómo y por qué" el programa hace lo que hace y determinar si el programa puede llegar a tener, o no, un comportamiento malicioso.

La idea consiste en crear un usuario dedicado a la administración (por ejemplo, `free` o `admin`) usando el comando `adduser`. Dicho usuario debe poder escribir en los directorios siguientes: `/usr/local/src`, `/usr/local/bin` y `/usr/local/lib`, así como también en todo el sub-árbol `/usr/man` (puede que también tenga que copiar archivos en otros lugares). Recomendamos por esto, hacer que este usuario sea el propietario de los directorios necesarios, o crear un grupo para él, y hacer que dicho grupo pueda escribir en esos directorios.

Una vez que se tomaron estas precauciones, Ud. puede seguir las instrucciones de la sección *La forma más fácil*, página 143.

16.3. Configuración

Un interés puramente técnico del hecho de disponer de los fuentes es la posibilidad de ***portar*** el software. Un software libre desarrollado para un sistema *Unix* se puede usar en todos los sistemas *Unix* existentes (sean libres o propietarios), con pocas modificaciones. Esto implica una configuración del software justo antes de la compilación.

Existen muchos sistemas de configuración, Ud. tiene que usar el que el autor del software quiera (a veces, se necesitan varios). Por lo general, Ud. puede:

4. Un proverbio del mundo de BSD dice: "never trust a software you don't have the sources for" (que significa: nunca confíe en un software del cual no tenga el código fuente).

- usar *Autoconf* (ver la sección *autoconf*, página 144) si existe un archivo denominado `configure` en el directorio padre de la distribución.
- usar *Imake* (ver la sección *Imake*, página 146) si existe un archivo denominado `Imakefile` en el directorio padre de la distribución.
- ejecutar un ***script*** del *shell*, (por ejemplo, `install.sh`) según lo que diga el archivo `INSTALL` (o el archivo `README`).

16.3.1. autoconf

16.3.1.1. Principio

Autoconf permite configurar el software correctamente. Crea los archivos necesarios para la compilación (por ejemplo, el archivo `Makefile`), y, a veces, cambia los fuentes directamente (como, por ejemplo, al usar un archivo `config.h.in`).

El principio de *Autoconf* es simple:

- el programador del software sabe qué pruebas son necesarias para configurar su software (ej.: "¿qué versión de esta o aquella ***biblioteca*** usa"?). Él las escribe, siguiendo una sintaxis precisa, en un archivo denominado `configure.in`.
- Él ejecuta *Autoconf*, el cual genera un script de configuración denominado `configure` a partir del archivo denominado `configure.in`. Este script efectuará las pruebas necesarias cuando se configure el programa.
- El usuario final ejecuta el script, y *Autoconf* se encarga de configurar todo lo que es necesario para la compilación.

16.3.1.2. Ejemplo

Un ejemplo del uso de *Autoconf*:

```
$ ./configure
loading cache ./config.cache
checking for gcc... gcc
checking whether the C compiler (gcc  ) works... yes
checking whether the C compiler (gcc  ) is a cross-compiler... no
checking whether we are using GNU C... yes
checking whether gcc accepts -g... yes
checking for main in -lX11... yes
checking for main in -lXpm... yes
checking for main in -lguile... yes
checking for main in -lm... yes
checking for main in -lncurses... yes
checking how to run the C preprocessor... gcc -E
checking for X... libraries /usr/X11R6/lib, headers /usr/X11R6/include
checking for ANSI C header files... yes
checking for unistd.h... yes
checking for working const... yes
updating cache ./config.cache
creating ./config.status
creating lib/Makefile
creating src/Makefile
creating Makefile
```

Para tener un mayor control de lo que genera `configure`, se le pueden pasar algunas opciones por medio de la línea de comandos o variables de entorno. Por ejemplo:

```
$ ./configure --with-gcc --prefix=/opt/GNU
```

o (con *Bash*):

```
$ export CC=‘which gcc‘
$ export CFLAGS=-O2
$ ./configure --with-gcc
```

o:

```
$ CC=gcc CFLAGS=-O2 ./configure
```

16.3.1.3. Qué pasa si... ¿no funciona?

Un error típico del script `configure` es aquel del tipo `configure: error: Cannot find library guile` (configure: error: no se encuentra la biblioteca `guile`) (La mayoría de los errores del script `configure` lucen así).

Esto significa que el script `configure` no pudo encontrar una biblioteca (`guile` en nuestro ejemplo). El principio es que el script `configure` compila un pequeño programa de prueba que usa esta biblioteca. Si esta compilación no tiene éxito, no podrá compilar el software. Entonces ocurre un error.

- Busque la razón del error examinando al final del archivo `config.log`, que contiene una traza de todos los pasos de configuración. El compilador de lenguaje *C* es suficientemente claro con sus mensajes de error. Eso lo ayudará a resolver el problema.
- Verifique que la biblioteca en cuestión esté instalada correctamente. Si no es así, puede correr `/sbin/ldconfig`, borrar el archivo `config.cache` y volver a ejecutar el script `configure`. Si todavía sigue con problemas, intente volver a instalar la biblioteca (desde los fuentes o desde un archivo binario). Una forma eficiente de verificar la instalación es buscar el archivo que contiene los símbolos de la biblioteca, que siempre se denomina `lib<nombre>.so`. Por ejemplo,

  ```
  $ find / -name 'libguile*'
  ```

  o, si no:

  ```
  $ locate libguile
  ```

- Verifique que el compilador puede acceder a ella. Esto significa que se encuentra en algún directorio entre: `/usr/lib`, `/lib`, `/usr/X11R6/lib` (o entre aquellos especificados por la variable de entorno `LD_LIBRARY_PATH`, explicada en *¿Qué pasa si... no funciona?*, página 148 número 5.b.). Verifique que este archivo es una biblioteca ingresando `file libguile.so`.
- Verifique que los archivos de encabezado correspondientes a la biblioteca se encuentran en el lugar adecuado (generalmente, `/usr/include` o `/usr/local/include` o `/usr/X11R6/include`). Si Ud. no sabe cuales son los archivos de encabezado necesarios, verifique que instaló la versión de desarrollo de la biblioteca en cuestión (por ejemplo, `gtk+-devel` además de `libgtk`). La versión de desarrollo de la biblioteca proporciona los archivos `include` (incluir) necesarios para compilar un software usando esta biblioteca.
- Verifique que Ud. tiene espacio suficiente en el disco (el script `configure` necesita de algo de espacio para archivos temporales). Use el comando `df -k` para visualizar el espacio disponible en las particiones de su sistema, y ocúpese de las particiones llenas o casi llenas.

Si Ud. no comprende los mensajes de error almacenados en el archivo `config.log`, no dude en pedir ayuda a la comunidad de software libre (ver la sección *Soporte técnico*, página 153).

Es más, verifique si `configure` responde 100% de `No` o si responde `No` y Ud. está seguro que la biblioteca existe (por ejemplo, sería muy extraño que no exista la biblioteca `curses` en su sistema). Si ese es el caso, ¡probablemente esté mal configurada la variable `LD_LIBRARY_PATH`!

16.3.2. Imake

`Imake` le permite configurar un software libre creando un archivo `Makefile` a partir de reglas simples. Estas reglas determinan los archivos necesarios para compilar el archivo binario, y luego *`Imake`* genera el archivo `Imakefile` correspondiente. Estas reglas se especifican en un archivo denominado `Imakefile`.

Lo que tiene interesante *`Imake`* es que usa información dependiente de la arquitectura y del sitio. Esto es muy útil para los programas que usan *`X Window System`*. Pero *`Imake`* se usa para muchas otras aplicaciones.

La forma más fácil de usar *`Imake`*, es entrar en el directorio principal del archivado descomprimido, y luego correr el script `xmkmf`, que llama al programa *`Imake`*:

```
$ xmkmf -a
imake -DUseInstalled -I/usr/X11R6/lib/X11/config
make Makefiles
```

Si el sitio no está instalado correctamente, ¡debe recompilar e instalar *`X11R6`*!

16.3.3. Varios scripts del `shell`

Para más información lea los archivos `INSTALL` o `README`. Por lo general, Ud. tiene que ejecutar un archivo del tipo `install.sh` o `configure.sh`. Entonces, o el script de instalación será silencioso (y determinará lo que necesita por sí solo), o le preguntará información sobre su sistema (por ejemplo, las rutas).

Si Ud. no llega a determinar el archivo que tiene que ejecutar, puede ingresar `./` (bajo *`Bash`*), y luego presionar dos veces la tecla **TAB** (tecla del tabulador). *`Bash`* completará automáticamente el nombre de un archivo ejecutable en el directorio corriente (por lo tanto, un posible script de configuración). En caso de que varios archivos se puedan ejecutar, le dará una lista. Solo debe elegir el archivo correcto.

Un caso particular es la instalación de módulos *`Perl`* (aunque no solamente de estos). La instalación de tales módulos se hace mediante la ejecución de un script de configuración, el cual se encuentra escrito en *`Perl`*. Por lo general, el comando a ejecutar es:

```
$ perl Makefile.PL
```

16.3.4. Alternativas

Algunas distribuciones de software libre están mal organizadas, especialmente durante las primeras etapas de desarrollo (¡pero se previene al usuario!). En las mismas se necesita retocar “a mano” algunos archivos de configuración. Por lo general, estos archivos son un archivo `Makefile` (ver la sección *make*, página 147) y un archivo `config.h` (este nombre sólo es convencional). Como siempre, ¡lea los archivos `README` e `INSTALL`!

No recomendamos que estas manipulaciones sean hechas por usuarios que no sepan lo que están haciendo. Esto necesita de conocimientos reales y la motivación necesaria para tener éxito. Pero la práctica lleva a la perfección.

16.4. Compilación

Ahora que el software está configurado correctamente, sólo falta compilarlo. Generalmente esta parte es fácil, y no presenta problemas serios.

16.4.1. make

`make` es la herramienta favorita de la comunidad de software libre para compilar los fuentes. Tiene dos cosas interesantes:

- le permite ganar tiempo al desarrollador, porque le permite administrar la compilación de su proyecto eficientemente,
- permite que el usuario final compile e instale el software en pocas líneas de comando, incluso si él no tiene conocimientos preliminares de desarrollo.

Las acciones a ejecutar para obtener una versión compilada de los fuentes generalmente se almacenan en un archivo denominado `Makefile` o `GNUMakefile`. De hecho, cuando se invoca a `make`, este lee dicho archivo, si existe, en el directorio corriente. Si no es así, se puede especificar el archivo usando la opción `-f` con `make`.

16.4.2. Reglas

`make` funciona de acuerdo a un sistema de ***dependencias***, razón por la cual la compilación de un archivo binario ("***objetivo***") necesita pasar por varias etapas ("dependencias"). Por ejemplo, para crear el archivo binario (imaginario) `glloq`, se deben compilar y luego vincular los archivos objeto `main.o` e `init.o` (archivos intermedios de la compilación). Estos archivos objeto también son objetivos, cuyas dependencias son los archivos fuente.

Este texto sólo es una introducción mínima para sobrevivir en el mundo sin piedad de `make`. Si Ud. quiere aprender más, le aconsejamos dirigirse al sitio *`web`* de **APRIL** (http://www.april.org/groupes/doc), donde puede encontrar documentación más detallada sobre `make`. Para una documentación exhaustiva, debe referirse a **Managing Projects with Make** (**Administración de proyectos con Make**), segunda edición, de **O'Reilly**, por *Andrew Oram* y *Steve Talbott*.

16.4.3. Go, go, go!

Por lo general, el uso de `make` obedece a muchas convenciones. Por ejemplo:

- `make` sin argumentos simplemente ejecuta la compilación del programa, sin instalarlo.
- `make install` compila el programa (aunque no siempre), y asegura la instalación de los archivos necesarios en el lugar adecuado del sistema de archivos. Algunos archivos no siempre se instalan correctamente (`man`, `info`), ellos deben ser copiados por el usuario. Algunas veces, `make install` tiene que volver a ser ejecutado en los sub-directorios. Por lo general, esto pasa con los módulos desarrollados por terceros.
- `make clean` borra todos los archivos temporales creados por la compilación, y, en la mayoría de los casos, el archivo ejecutable.

La primera etapa para compilar un programa es ingresar (ejemplo imaginario):

```
$ make
gcc -c glloq.c -o glloq.o
gcc -c init.c -o init.o
gcc -c main.c -o main.o
gcc -lgtk -lgdk -lglib -lXext -lX11 -lm glloq.o init.o main.o -o glloq
```

Excelente, el archivo binario está compilado correctamente. Ahora estamos preparados para la etapa siguiente, que es la instalación de los archivos de la distribución (archivos binarios, archivos de datos, etc...). Ver la sección *Instalación*, página 152).

16.4.4. Explicaciones

Si Ud. es lo suficientemente curioso para mirar el archivo `Makefile`, encontrará comandos conocidos (`rm`, `mv`, `cp`, ...), aunque también encontrará algunas cadenas extrañas, de la forma `$(CFLAGS)`.

Estas son ***variables***, es decir las cadenas que generalmente se fijan al comienzo del archivo `Makefile`, y luego se reemplazan por el valor con el cual están asociadas. Estas son muy útiles cuando quiere usar las mismas opciones de compilación varias veces.

Por ejemplo, puede mostrar la cadena "`pepe`" en la pantalla usando `make all`:

```
TEST = pepe
all:
        echo $(TEST)
```

La mayor parte del tiempo, se definen las variables siguientes:

1. `CC`: este es el compilador que va a utilizar. Generalmente es *`cc1`*, que en la mayoría de los sistemas libres, es sinónimo de `gcc`. Cuando tenga dudas, ponga aquí `gcc`.
2. `LD`: este es el programa usado para asegurar la fase de la compilación final (ver la sección *Las cuatro fases de la compilación*, página 140). El valor predeterminado es `ld`.
3. `CFLAGS`: estos son los argumentos adicionales que se pasarán al compilador durante las primeras etapas de la compilación. Entre ellos:
   - `-I<ruta>`: le especifica al compilador donde buscar algunos archivos de encabezado adicionales (por ejemplo: `-I/usr/X11R6/include` permite incluir los archivos de encabezado que están en el directorio `/usr/X11R6/include`)
   - `-D<símbolo>`: define un símbolo adicional, útil para los programas cuya compilación depende de los símbolos definidos (ej: utilizar el archivo `string.h` si está definida `HAVE_STRING_H`).

   Generalmente hay líneas de compilación de la forma:

   ```
   $(CC) $(CFLAGS) -c pepe.c -o pepe.o
   ```

4. `LDFLAGS` (o `LFLAGS`): estos son los argumentos que se usan durante la última etapa de compilación. Entre ellos:
   - `-L<ruta>`: especifica una ruta adicional donde buscar bibliotecas (por ejemplo: `-L/usr/X11R6/lib`).
   - `-l<biblioteca>`: especifica una biblioteca adicional para usar durante la última etapa de compilación.

16.4.5. ¿Qué pasa si... no funciona?

No tenga pánico, le puede pasar a cualquiera. Entre las causas más comunes:

1. `glloq.c:16: decl.h: No such file or directory` (`glloq.c :16`: `decl.h`: no hay archivo o directorio con ese nombre)

   El compilador no pudo encontrar el archivo de encabezado correspondiente. Por lo tanto, la etapa de configuración del software debería haber anticipado este error. Como resolver este problema:

- verifique que verdaderamente exista el archivo de encabezado en uno de los directorios siguientes: `/usr/include`, `/usr/local/include`, `/usr/X11R6/include` o en alguno de sus sub-directorios. De no ser así, búsquelo por todo el disco (con `find` o `locate`), y, si todavía no lo encuentra, verifique que ha instalado la biblioteca de desarrollo correspondiente a este archivo de encabezado. Puede encontrar ejemplos de los comandos `find` y `locate` en las respectivas páginas Man.
- Verifique que el archivo de encabezado se pueda leer (para verificarlo, puede ingresar `less <ruta>/<archivo>.h`).
- Si es un directorio como `/usr/local/include` o como `/usr/X11R6/include`, Ud. tiene que agregar, a veces, un argumento nuevo al compilador. Abra el archivo `Makefile` correspondiente (tenga cuidado de abrir el archivo correcto, los que se encuentran en el directorio donde falla la compilación[5]) con su editor de texto favorito (*Emacs*, *VI*, etc). Busque la línea errónea, y agregue la cadena `-I<ruta>`, donde `<ruta>` es la ruta donde se puede encontrar el archivo de encabezado en cuestión, justo después de la llamada del compilador (`gcc`, o, a veces, `$(CC)`). Si no sabe donde agregar esta opción, agréguela al comienzo del archivo, después de `CFLAGS=<algo>` o después de `CC=<algo>`.
- ejecute `make` de nuevo, y si todavía sigue sin funcionar, verifique que esta opción (ver el punto anterior) se agrega durante la compilación en la línea errónea.
- si todavía sigue sin funcionar, pida ayuda al autor del software, a su gurú local, o a la comunidad de software libre para resolver su problema (ver la sección *Soporte técnico*, página 153).

2. `glloq.c:28: ‘struct pepe’ undeclared (first use this function)`

   (`glloq.c:28`: “`struct pepe`” no está declarada (esta es la primera utilización en esta función))

   Las estructuras son tipos de datos especiales, que usan todos los programas. El sistema define un montón de ellas en los archivos de encabezados. Eso significa que es muy probable que el problema sea la falta o el mal uso de un archivo de encabezado. El procedimiento correcto para resolver el problema es:

   - intente verificar si la estructura en cuestión es una estructura definida por el programa o por el sistema. Una solución es usar el comando *grep* para ver si la estructura está definida en alguno de los archivos de encabezado.

     Por ejemplo, cuando Ud. está en la raíz de la distribución:

     ```
     $ find . -name '*.h'| xargs grep 'struct pepe' | less
     ```

     Es posible que aparezcan muchas líneas en la pantalla (por ejemplo, cada vez que se define una función que use esta estructura). Elija, si existe, la línea donde se define la estructura mirando el archivo de encabezado obtenido por la utilización del comando *grep*.

     La definición de una estructura es:

     ```
     struct pepe {
           <contenido de la estructura pepe>
     };
     ```

     Verifique si ella corresponde a lo que Ud. tiene. De ser así, eso significa que no se incluye el encabezado en el archivo `.c` erróneo. Hay dos soluciones:

     - agregar la línea `#include "<nombre_de_archivo>.h"` al comienzo del archivo `.c` erróneo.

5. Ud. debe analizar el mensaje de error que devuelve `make`. Normalmente, las últimas líneas deberían contener un directorio (un mensaje como `make[1]: Leaving directory ‘/home/benj/Proyecto/pepe’`). Elija aquel que tiene el número más grande. Para verificar que es el bueno, vaya al directorio y ejecute `make` de nuevo para obtener el mismo error.

- o copiar y pegar la definición de la estructura al comienzo del archivo `.c` (esto no es de lo mejor, pero al menos por lo general, funciona).

- si este no es el caso, haga lo mismo con los archivos de encabezado del sistema (que, generalmente, se encuentran en los directorios siguientes: `/usr/include`, `/usr/X11R6/include` y `/usr/local/include`). Pero en este caso, use la línea `#include <<nombre_de_archivo>.h>`.
- si todavía no existe esta estructura, intente encontrar en que biblioteca (es decir, conjunto de funciones agrupadas en un solo paquete) debería estar definida (ver en el archivo `INSTALL` o `README` cuales son las bibliotecas que usa este programa y las versiones necesarias). Si la versión que necesita el programa no está instalada en el sistema, Ud. deberá actualizar esta biblioteca. Cabe destacar que debe tener sumo cuidado al manipular los archivos de encabezado del sistema, ya que estos son comunes a muchos programas.
- si todavía sigue sin funcionar, verifique que el programa funciona adecuadamente sobre su arquitectura (algunos programas todavía no han sido portados a todos los *Unix*). Verifique también que ha configurado el programa correctamente (por ejemplo, cuando ejecutó `configure`) para su arquitectura.

3. `parse error` (error de análisis sintáctico)

   Este es un problema que es relativamente complicado de resolver, porque generalmente es un error que aparece en cierta línea, pero después que el compilador lo encontró. A veces, es simplemente un tipo de datos que no está definido. Si Ud. encuentra un mensaje de error del tipo:

   ```
   main.c:1: parse error before ‘glloq_t
   main.c:1: warning: data definition has no type or storage class
   ```

   entonces el problema es que el tipo `glloq_t` no está definido. La solución al problema es más o menos la misma que en el problema anterior.

   > **Nota:** puede haber un error del tipo `parse error` en las bibliotecas `curses` antiguas si la memoria no nos falla.

4. `no space left on device` (no queda espacio en el dispositivo)

   El problema es simple de resolver: no hay espacio suficiente en el disco para generar un archivo binario a partir del archivo fuente. La solución consiste en liberar espacio en la partición que contiene el directorio de instalación (borrar archivos innecesarios, archivos temporales, desinstalar programas que no use). Si descomprimió en `/tmp`, mejor hágalo en `/usr/local/src` que evita la saturación innecesaria de la partición `tmp`. Es más, verifique si hay archivos `core>` en su disco. De ser así, elimínelos o haga que el usuario al cual pertenezcan los elimine. En fin, trate de liberar espacio en disco.

5. `/usr/bin/ld: cannot open -lglloq: No such file or directory` (`/usr/bin/ld: no puedo abrir -lglloq: no hay archivo o directorio alguno con ese nombre`)

   Esto significa claramente que el programa *ld* (usado por `gcc` durante la última etapa de la compilación) no puede encontrar una biblioteca. Para incluir una biblioteca, *ld* busca un archivo cuyo nombre está en los argumentos del tipo `-l<biblioteca>`. Este archivo es `lib<biblioteca.so>`. Si *ld* no puede encontrarlo, produce un mensaje de error. Para resolver el problema, siga los pasos que se indican a continuación:

   a. verifique que el archivo existe en el disco rígido usando el comando `locate`. Por lo general, las bibliotecas gráficas se encuentran en `/usr/X11R6/lib`. Por ejemplo:

```
$ locate libglloq
```

Si la búsqueda no tiene resultado, Ud. puede buscar con el comando `find` (ej: `find /usr -name libglloq.so*`). Si Ud. no puede encontrar la biblioteca, entonces tendrá que instalarla.

b. una vez que está localizada la biblioteca, verifique que *ld* puede accederla: el archivo `/etc/ld.so.conf` especifica donde encontrar estas bibliotecas. Agregue el directorio en cuestión al final del mismo y ejecute `ldconfig`. También puede agregar este directorio a la variable de entorno `LD_LIBRARY_PATH`. Por ejemplo, si el directorio a agregar es `/usr/X11R6/lib`, ingrese:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/X11R6/lib
```

(si su *shell* es *Bash*).

c. si todavía no funciona, verifique que el formato de la biblioteca en cuestión es un archivo ejecutable (o ELF) (con el comando `file`). Si esta es un vínculo simbólico, verifique que el vínculo es correcto y no apunta a un archivo inexistente (por ejemplo, con `nm <lib>`). Los permisos pueden ser erróneos (por ejemplo, Ud. usa una cuenta que no es `root` y la biblioteca está protegida contra lectura).

6. `glloq.c(.text+0x34): undefined reference to 'glloq_init'` (`glloq.c(.text+0x34)`: referencia indefinida al símbolo '`glloq_init`')

Esto significa que no se resolvió un símbolo durante la última etapa de la compilación. Por lo general, este es un problema de biblioteca. Puede haber varias causas:

- la primera cosa a hacer es saber si se **supone** que el símbolo esté presente en una biblioteca. Por ejemplo, si es un símbolo que comienza por `gtk`, pertenece a la biblioteca `gtk`. Si el nombre de la biblioteca es difícil de identificar (como por ejemplo, `zorglub_gloubiboulga`), se pueden listar los símbolos de una biblioteca con el comando `nm`. Por ejemplo,

```
$ nm libglloq.so
0000000000109df0 d glloq_message_func
000000000010a984 b glloq_msg
0000000000008a58 t glloq_nearest_pow
0000000000109dd8 d glloq_free_list
0000000000109cf8 d glloq_mem_chunk
```

Agregar la opción `-o` a `nm` permite mostrar el nombre de la biblioteca en cada línea, lo cual facilita la búsqueda. Imaginemos que buscamos el símbolo `bulgroz_max`, una solución cruda es realizar una búsqueda del tipo:

```
$ nm /usr/lib/lib*.so | grep bulgroz_max
$ nm /usr/X11R6/lib/lib*.so | grep bulgroz_max
$ nm /usr/local/lib/lib*.so | grep bulgroz_max
/usr/local/lib/libfrobnicate.so:000000000004d848 T bulgroz_max
```

¡Formidable! El símbolo `bulgroz_max` está definido en la biblioteca `zorglub` (la letra mayúscula `T` se encuentra delante de su nombre). Entonces, Ud. sólo tiene que agregar la cadena `-lzorglub` en la línea de compilación editando el archivo `Makefile`: agréguela al final de la línea donde se define la variable `LDFLAGS` o `LFGLAGS` (o, en el peor de los casos, `CC`), o en la línea correspondiente a la creación del archivo binario final.

- la compilación está hecha con una versión de la biblioteca que no es la que permite el software. Lea el archivo `README` o `INSTALL` de la distribución para saber que versión de la biblioteca debe usar.

- no todos los archivos objeto de la distribución están vinculados correctamente. Falta, o no se menciona, el archivo donde está definida esta función. Ingrese `nm -o *.o`, para saber el nombre del archivo `.o` correspondiente y agréguelo en la línea de compilación si es que falta.
- la función o variable en cuestión puede ser fantasiosa. Intente eliminarla: edite el archivo fuente en cuestión (el nombre del mismo se especifica al comienzo del mensaje de error). Esta es una solución desesperada cuya consecuencia ciertamente será un funcionamiento anárquico del software (error de segmentación en el arranque, etc.)

7. `Segmentation fault (core dumped)` (Error de segmentación (se produjo un archivo `core`))

   A veces, el compilador se cuelga lamentablemente y produce este mensaje de error. No tengo consejo alguno, salvo pedirle que instale una versión más reciente de su compilador.

8. no queda espacio en `/tmp`

   La compilación necesita espacio temporal durante las diferentes etapas; si no tiene espacio suficiente, falla. Por lo tanto, debe limpiar la partición, pero debe tener cuidado ya que algunos programas en curso de ejecución (servidor *X*, tuberías...) se pueden colgar si se borran algunos archivos. ¡Ud. debe saber lo que está haciendo! Si `tmp` es parte de una partición que no la contiene solamente (por ejemplo, la raíz), busque y borre algunos archivos `core` eventuales.

9. `make`/`configure` en bucle infinito

   Generalmente, esto es un problema de la hora en su sistema. En efecto, `make` necesita saber la fecha y la hora de la computadora y de los archivos que verifica. Este compara las fechas de los archivos y usa el resultado para saber si el objetivo es más reciente que la dependencia.

   Algunos problemas en la fecha pueden inducir a `make` a reconstruirse a sí mismo indefinidamente (o a construir y reconstruir un sub-árbol en bucle). Si es así, el uso del comando `touch` (cuya consecuencia es poner en la hora corriente a los archivos pasados como argumento) resuelve el problema la mayoría de las veces.

   Por ejemplo:

   ```
   $ touch *
   ```

   O también, (más bruto, pero eficiente):

   ```
   $ find . | xargs touch
   ```

16.5. Instalación

16.5.1. Con make

Ahora que todo está compilado, Ud. tiene que copiar los archivos producidos a un lugar adecuado (por lo general, uno de los sub-directorios de `/usr/local`).

Generalmente, `make` puede hacer esto. Un objetivo especial es el objetivo `install`. Así que, lógicamente, `make install` permite instalar los archivos necesarios.

Por lo general, el procedimiento está descrito en los archivos `INSTALL` o `README`. Pero, algunas veces, el desarrollador se olvida de proporcionarlos. En ese caso, Ud. deberá instalar todo por su cuenta.

Entonces, copie:

- Los archivos ejecutables (los programas) en el directorio denominado `/usr/local/bin`.
- Las bibliotecas (archivos `lib*.so`) en el directorio denominado `/usr/local/lib`.
- Los archivos de encabezado (archivos `*.h`) en el directorio denominado `/usr/local/include` (tenga cuidado de no borrar los archivos originales).
- Los archivos de datos generalmente van en el directorio denominado `/usr/local/share`. Si Ud. no conoce el procedimiento de instalación, puede intentar correr los programas sin copiar los archivos de datos, y ponerlos en el lugar indicado cuando se le pida (con un mensaje de error del tipo `Cannotopen/usr/local/share/glloq/data.db`).
- la documentación es un poquito diferente:
  - Los archivos `man` generalmente se ponen en uno de los sub-directorios de `/usr/local/man`. Por lo general, estos archivos están en formato `troff` (o `groff`), y su extensión es un número. Su nombre es el nombre del comando (por ejemplo, `echo.1`). Si el número es `n`, copie el archivo en el sub-directorio `/usr/local/man/man<n>`.
  - los archivos `info` se ponen en el directorio `/usr/info` o `/usr/local/info`.

¡Y listo! ¡Enhorabuena! Ahora, ¡Ud. está listo para recompilar su sistema operativo por completo!

16.5.2. Problemas

Si recién termina de instalar un software libre, por ejemplo *GNU tar*, y si, cuando lo ejecuta, se inicia otro software o no funciona como lo probó directamente desde el directorio `src`, entonces tiene un problema con `PATH`. `PATH` es una variable de entorno que indica la ruta a seguir cuando se busca un ejecutable. Ud. puede verificarlo, ejecutando `type -a <programa>`.

La solución es poner el directorio de instalación más alto en la variable `PATH`, y/o borrar/renombrar los archivos que se ejecutan cuando no se desea, y/o renombrar sus programas nuevos (en este ejemplo, como `gtar`) para que no haya más confusión.

Ud. también puede hacer una alias si el *shell* se lo permite (por ejemplo, decir que `tar` significa `/usr/local/bin/gtar`).

16.6. Soporte

16.6.1. Documentación

Varias fuentes de documentación:

- los COMO son documentos cortos (mayormente, en inglés) sobre temas precisos (generalmente, mucho más de lo que necesitamos acá, pero útiles sin lugar a dudas). Busque en su disco rígido en el directorio `/usr/doc/HOWTO` (aunque a veces, deberá verificar esto usando el comando `locate HOWTO`). También están disponibles las traducciones al castellano de los COMO. Puede obtenerlas en la página de `LuCAS` (Proyecto de documentación de *GNU/Linux* en CAStellano) en la URL http://lucas.hispalinux.es/.
- Las páginas Man. Ingrese `man <comando>` para obtener información sobre el comando `<comando>`,
- la literatura especializada. Muchos editores grandes comenzaron a publicar libros acerca de los sistemas libres (especialmente sobre *GNU/Linux*). Esto es muy útil si Ud. es un principiante y no entiende todos los términos de la documentación presente.

16.6.2. Soporte técnico

Si Ud. compró una distribución "oficial" de **Linux-Mandrake**, puede dirigirse al personal de soporte técnico con preguntas sobre su sistema. Creemos que el soporte técnico tiene otras cosas que hacer que ayudar a todos los usuarios a instalar software adicional, pero algunos de ellos ofrecen una ayuda de instalación de `x` días. ¿Tal vez puedan pasar algo de tiempo con problemas de compilación?

Como mencionamos anteriormente, Ud. también puede dirigirse a la comunidad de software libre:

- Los ***foros de discusión*** (de *Usenet*) es.comp.os.linux (news:es.comp.os.linux) contestan todas las preguntas sobre *GNU/Linux*. Los foros de discusión que coinciden con `comp.os.bsd.*` tratan con los sistemas BSD. Pueden haber otros foros de discusión que traten con otros sistemas *Unix*. Recuerde leerlos por un tiempo antes de comenzar a escribirles.
- Varias asociaciones o grupos de entusiastas de la comunidad de software libre ofrecen un soporte voluntario. La mejor manera de encontrar los más cercanos al lugar donde vive es verificar las listas de los sitios *web* especializados o leer los foros de discusión por un rato. Por ejemplo, en Argentina (país donde vive el traductor **:-)**) existe el grupo de usuarios de Argentina ("LuGAr") cuya página principal es http://www.linux.org.ar donde puede encontrar muchísima (y muy buena) información[6].
- Muchos ***canales IRC*** ofrecen una asistencia en tiempo real (pero, a ciegas) por los gurús. Por ejemplo, Ud. puede ver el canal `#linux` en la mayor parte de la red IRC, o `#linuxhelp` en *IRCNET* (Aunque la mayoría, por no decir la totalidad, sean en inglés...)
- Como último recurso, pregúntele al autor del software (si es que menciona su nombre y su ***dirección electrónica*** en algún archivo de la distribución) si Ud. está seguro que encontró un *bug* (que puede ser debido a su arquitectura, pero después de todo, se supone que el software libre es portable)

16.6.3. Como encontrar software libre

Para encontrar software libre, pueden ser útiles un montón de vínculos:

- el sitio FTP enorme de Metalab (sunsite.unc.edu) o uno de sus sitios espejo
- los sitios *web* que siguen conforman un catálogo de mucho software libre que se puede usar sobre las plataformas *Unix* (aunque Ud. también puede encontrar software propietario en ellos):
  - http://www.freshmeat.net/ probablemente sea el sitio más completo,
  - http://rufus.w3.org/linux/RPM/ contiene miles de paquetes en formato RPM con un motor de búsqueda para encontrar el que Ud. necesita,
  - http://www.gnu.org/software/ para una lista exhaustiva de todo el software GNU. Por supuesto, todos ellos son libres y la mayoría está bajo la licencia GPL.
- también puede realizar una búsqueda usando los motores de búsqueda como http://www.google.com/linux. También puede dirigirse a los conocidos http://www.altavista.com/ y http://www.yahoo.com/ y efectuar una búsqueda del tipo: `+<software> +download` o `"download software"`.

6. Aunque no viva en Argentina, le recomiendo visitarla...

16.7. Agradecimientos

- Re-lecturas y comentarios críticos (y en orden alfabético): *Sébastien Blondeel*, *Mathieu Bois*, *Xavier Renaut*, *Kamel Sehil*.
- `Beta-testing`: *Laurent Bassaler*
- Traducción al castellano: *Fabian Mandelbaum*

Apéndice A. La Licencia Pública General GNU

El texto siguiente es la licencia GPL que se aplica a la mayoría de los programas que se encuentran en las distribuciones **Linux-Mandrake**.

Esta es una traducción al castellano no oficial de la Licencia Pública General GNU. No fue publicada por la Free Software Foundation, y legalmente no establece los términos de distribución de software que usa la GPL GNU– sólo el texto original en inglés de la GPL GNU hace eso. Sin embargo, esperamos que esta traducción ayudará a las personas que hablan castellano a comprender mejor la GPL GNU.

Traducido por Fabian (Fabito) Mandelbaum en Mayo de 2000. Buenos Aires. Argentina.

A.1. Preámbulo

Las licencias para la mayoría del software están diseñadas para quitarle su libertad de compartirlo y cambiarlo. En contraste, la Licencia Pública General GNU pretende garantizarle su libertad para compartir y cambiar el software libre – para asegurarse que el software es libre para todos sus usuarios. Esta Licencia Pública General se aplica a la mayoría del software de la “Free Software Foundation” y a cualquier otro programa cuyos autores se comprometan a usarla. (No obstante, algún otro software de la Free Software Foundation está cubierto por la Licencia Pública General de Biblioteca GNU LGPL). Ud. también puede aplicarla a sus programas.

Cuando hablamos de software libre, nos estamos refiriendo a libertad, no al precio. Nuestras Licencias Públicas Generales están diseñadas para asegurarse que Ud. tiene la libertad de distribuir copias de software libre (y, si Ud. lo desea, puede cobrar por este servicio), que Ud. recibe el código fuente o puede obtenerlo si así lo desea, que Ud. puede cambiar el software o usar piezas del mismo en programas libres nuevos; y que Ud. sabe que puede hacer estas cosas.

Para proteger sus derechos, necesitamos hacer restricciones que prohíban a cualquiera el negarle a Ud. estos derechos o pedirle a Ud. que renuncie a los derechos. Estas restricciones se traducen en ciertas responsabilidades para Ud. si es que distribuye copias del software, o si lo modifica.

Por ejemplo, si Ud. distribuye copias de tal programa, ya sea gratis o con un costo, Ud. debe darle a quienes lo reciban todos los derechos que Ud. posee. Ud. debe asegurarse que ellos, también, reciban o puedan obtener el código fuente. Y Ud. debe mostrarles estos términos para que ellos conozcan sus derechos.

Nosotros protegemos sus derechos con dos pasos:

1. el *copyright* (derecho de autor) del software, y
2. le ofrecemos esta licencia que le otorga permiso legal para copiar, distribuir y/o modificar el software.

También, para la protección de cada autor y la nuestra, nos queremos asegurar que todos entiendan que no hay garantía alguna para este software libre. Si un tercero modifica el software y lo distribuye, nosotros queremos que quienes lo reciban sepan que lo que ellos poseen no es el original, por lo que cualquier problema introducido por terceros no afectará la reputación del autor original.

Finalmente, cualquier programa libre está constantemente amenazado por las patentes de software. Nosotros deseamos evitar el peligro que los redistribuidores de un programa libre obtengan individualmente licencias de las patentes, haciendo al programa, en efecto, propietario. Para evitar esto, nosotros hemos aclarado que cualquier patente debe ser licenciada para el uso personal libre de cualquiera o no ser licenciada en absoluto.

Los términos y condiciones precisos para copiar, distribuir y modificar son los siguientes.

A.2. Términos y condiciones para la copia, distribución y modificación

- 0. Esta licencia se aplica a cualquier programa u otro trabajo que contiene una nota puesta por quien posee el copyright diciendo que puede ser distribuido bajo los términos de esta Licencia Pública General. En adelante, el "Programa" se refiere a cualquiera de tales programas o trabajos, y un "trabajo basado en el Programa" significa o el Programa o cualquier trabajo derivado bajo la ley del copyright: es decir, un trabajo conteniendo el Programa o una porción del mismo, ya sea textual o con modificaciones y/o traducciones a otro idioma. (En adelante, traducción se incluye sin limitación en el término "modificación"). Se dirige a cada titular de la licencia como "Ud.".

  Actividades que no sean la copia, distribución y modificación no están cubiertas por esta Licencia; están fuera del campo de la misma. El acto de ejecutar el Programa no está restringido, y la respuesta del Programa está cubierta sólo si su contenido constituye un trabajo basado en el Programa (independiente de haber sido el resultado de la ejecución del Programa). Que eso sea cierto depende de lo que haga el Programa.

- 1. Ud. puede copiar y distribuir copias textuales del código fuente del Programa como lo recibió, en cualquier medio, si Ud. publica en cada copia visible y adecuadamente una nota de copyright apropiada y la renuncia de garantía; mantiene intactas todas las notas que refieren a esta Licencia y a la ausencia de garantía alguna; y le da a cualquier otra persona que reciba el Programa una copia de esta Licencia junto con el Programa.

  Ud. puede cobrar un honorario por el acto físico de transferir una copia, y Ud. puede, a su elección, ofrecer la protección de la garantía a cambio de un honorario.

- 2. Ud. puede modificar su copia o sus copias del Programa o cualquier porción del mismo, conformando entonces un trabajo basado en el Programa, y copiar y distribuir tales modificaciones o trabajo bajo los términos de la Sección 1 arriba expuestos, si Ud. también cumple con todas estas condiciones:

  1. Ud. debe hacer que los archivos modificados tengan notas prominentes que digan que Ud. cambió los archivos y la fecha de cualquier cambio.
  2. Ud. debe hacer que cualquier trabajo que Ud. distribuya o publique, que en todo o en parte contiene o está derivado del Programa o cualquier parte del mismo, sea licenciado como un todo sin cargo a todas las terceras partes bajo los términos de esta Licencia.
  3. Si el Programa modificado normalmente lee comandos interactivamente cuando se ejecuta, Ud. puede hacer que, cuando el mismo inicie la corrida de tal uso interactivo de la manera más común, el Programa imprima o muestre un anuncio incluyendo la nota de copyright apropiada y una nota que indique que no hay garantía alguna (caso contrario, que diga que Ud. proporciona una garantía) y que los usuarios pueden redistribuir el programa bajo estas condiciones, y diciéndole al usuario como ver una copia de esta Licencia. (Excepción: si el Programa en sí mismo es interactivo pero normalmente no imprime tal anuncio, no es necesario que su trabajo basado en el Programa imprima un anuncio).

  Estos requisitos se aplican al trabajo modificado como un todo. Si secciones identificables de ese trabajo no están derivadas del Programa, y pueden considerarse razonablemente un trabajo separado e independiente por sí mismas, entonces esta Licencia, y sus términos, no se aplican a dichas secciones cuando Ud. las distribuye como trabajos separados. Pero cuando Ud. distribuye las mismas secciones como parte de un todo el cual es un trabajo basado en el Programa, la distribución del todo debe ser bajo los término de esta Licencia, cuyos permisos para otras licencias se extienden al todo, y por lo tanto, a todas y cada una de las partes sin importar quien las escribió.

Por lo tanto, la intención de esta sección no es reclamar derechos o competir por sus derechos sobre un trabajo escrito enteramente por Ud; sino, la intención es ejercitar el derecho de controlar la distribución de trabajos derivativos o colectivos basados en el Programa.

Además, el mero agregado de otro trabajo que no esté basado en el Programa junto con el Programa (o con un trabajo basado en el Programa) sobre un volumen de almacenamiento o medio de distribución no pone al otro trabajo bajo el marco de esta Licencia.

- 3. Ud. puede copiar y distribuir el Programa (o un trabajo basado en el mismo, bajo la Sección 2) en forma de código objeto o ejecutable bajo los términos de las Secciones 1 y 2 anteriores siempre y cuando Ud. también haga algo de lo siguiente:

  1. Lo acompaña con el código fuente legible por la máquina completo, el cual debe ser distribuido bajo los términos de las Secciones 1 y 2 anteriores sobre un medio comúnmente usado para el intercambio de software; o,
  2. Lo acompaña con una oferta escrita, válida por al menos tres años, de dar a cualquier tercero, por un cargo no mayor a su costo de realizar físicamente la distribución fuente, una copia completa legible por la máquina del código fuente correspondiente, a ser distribuido bajo los términos de las Secciones 1 y 2 anteriores sobre un medio comúnmente usado para el intercambio de software; o,
  3. Lo acompaña con la información que Ud. recibió como la oferta de distribuir el código fuente correspondiente. (Esta alternativa sólo está permitida para distribución no comercial y sólo si Ud. recibió el programa en forma de código objeto o ejecutable con tal oferta, de acuerdo con la Sub-sección b anterior).

  El código fuente para un trabajo significa la forma preferida del mismo para hacerle modificaciones. Para un trabajo ejecutable, el código fuente completo significa todo el código fuente para todos los módulos que contiene, más cualquier archivo asociado de definición de interfaces, más todos los scripts usados para controlar la compilación e instalación del ejecutable. Sin embargo, como una excepción especial, el código fuente distribuido no necesita incluir cosa alguna que normalmente se distribuya (ya sea en forma fuente o binaria) con los componentes mayores (compilador, núcleo, y así sucesivamente) del sistema operativo sobre el cual corre el ejecutable, a menos que dicho componente en sí mismo acompañe al ejecutable.

  Si la distribución del ejecutable o el código objeto se hace ofreciendo acceso a la copia desde un sitio designado, entonces el hecho de ofrecer la copia del código fuente desde el mismo sitio cuenta como distribución del código fuente, incluso si los terceros no están obligados a copiar los fuentes junto con el código objeto.

- 4. Ud. no puede copiar, modificar, sub-licenciar, o distribuir el Programa excepto como se provee expresamente bajo esta Licencia. Cualquier intento contrario de copiar, modificar, sub-licenciar o distribuir el Programa está prohibido, y anulará automáticamente sus derechos sobre esta Licencia. Sin embargo, a las partes que han recibido copias, o derechos, de Ud. bajo esta Licencia no se les anularán sus licencias siempre y cuando tales partes cumplan la misma por completo.
- 5. No es necesario que Ud. acepte esta Licencia, ya que Ud. no la firmó. Sin embargo, nada más le garantiza a Ud. el permiso para modificar o distribuir el Programa o sus trabajos derivativos. Estas acciones están prohibidas por ley si Ud. no acepta esta Licencia. Por lo tanto, al modificar o distribuir el Programa (o cualquier trabajo basado en el Programa), Ud. indica su aceptación de esta Licencia para hacerlo, y todos sus términos y condiciones para copiar, distribuir o modificar el Programa o los trabajos basados en el mismo.
- 6. Cada vez que Ud. redistribuye el Programa (o cualquier trabajo basado en el Programa), quien lo recibe automáticamente recibe una licencia del licenciatario original para copiar, distribuir o modificar el Programa sujeto a estos términos y condiciones. Ud. no puede imponer cualquier otra restricción sobre el ejercicio de los derechos aquí garantizados de quienes lo reciban. Ud. no es responsable de forzar el cumplimiento de esta Licencia por parte de terceros.

- 7. Si, como consecuencia de un veredicto de una corte o alegato de usurpación de una patente o cualquier otra razón (no limitada a cuestiones de patentes), se le imponen condiciones (ya sea por orden de la corte, convenio u otros) que contradicen las condiciones de esta Licencia, esto no lo libera a Ud. de las condiciones de esta Licencia. Si Ud. no puede hacer la distribución de manera de satisfacer simultáneamente sus obligaciones bajo esta Licencia y cualquier otra u otras obligaciones pertinentes, entonces como consecuencia, Ud. no puede distribuir el Programa en absoluto. Por ejemplo, si una licencia de patente no permite la distribución sin regalías del Programa por todos aquellos que reciban copias directamente o indirectamente a través de Ud., entonces la única forma en la cual Ud. puede satisfacer tanto esta Licencia como la otra sería contenerse en absoluto de distribuir el Programa.

  Si, bajo cualquier circunstancia particular, cualquier porción de esta sección se invalida o no se puede forzar, se pretende aplicar el balance de esta sección y la sección como un todo pretende aplicar en otras circunstancias.

  No es el propósito de esta sección inducir a Ud. a violar patente alguna o cualquier otro reclamo de derechos de propiedad o debatir la validez de cualquiera de tales reclamos; esta sección tiene el sólo propósito de proteger la integridad del sistema de distribución de software libre, que está implementado por prácticas de licencia pública. Mucha gente ha hecho contribuciones generosas al amplio rango de software distribuido por medio de ese sistema confiando en la aplicación consistente de dicho sistema; queda a criterio del autor/donor decidir si él o ella está dispuesto a distribuir software por medio de cualquier otro sistema y una licencia no puede imponer esa elección.

  El propósito de esta sección es dejar bien en claro lo que se cree es una consecuencia del resto de esta Licencia.

- 8. Si la distribución y/o el uso del Programa está restringido en ciertos países ya sea por patentes o por interfases con copyright, el dueño del copyright original que pone al Programa bajo esta Licencia puede agregar una limitación explícita a la distribución geográfica excluyendo dichos países, por lo cual la distribución sólo está permitida en, o entre, los países no así excluidos. En tal caso, esta Licencia incorpora la limitación como si estuviese escrita en el cuerpo de esta Licencia.
- 9. La Free Software Foundation puede publicar versiones revisadas y/o nuevas de la Licencia Pública General de vez en cuando. Tales versiones nuevas serán similares en espíritu a la versión presente, pero pueden diferir en detalle para tratar problemas o intereses nuevos.

  A cada versión se le da un número de versión distintiva. Si el Programa especifica un número de versión de esta Licencia que aplica al mismo y a “cualquier versión posterior”, Ud. tiene la opción de seguir los términos y condiciones de cualquiera de esas versiones o de cualquier versión posterior publicada por la Free Software Foundation. Si el Programa no especifica un número de versión de esta Licencia, Ud. puede elegir cualquier versión publicada alguna vez por la Free Software Foundation.

- 10. Si Ud. desea incorporar partes del Programa dentro de otros Programas libres cuyas condiciones de distribución son diferentes, escriba al autor para pedirle permiso. Para el software cuyo copyright posee la Free Software Foundation, escriba a la Free Software Foundation; a veces, nosotros hacemos excepciones a esto. Nuestra decisión estará guiada por los dos objetivos de preservar el estado libre de todos los derivativos de nuestro software libre y de promover el compartir y volver a usar el software en general.
-

SIN GARANTÍA

DEBIDO A QUE EL PROGRAMA SE LICENCIA SIN CARGO ALGUNO, NO HAY GARANTÍA PARA EL MISMO, A LA EXTENSIÓN PERMITIDA POR LA LEY APLICABLE. EXCEPTO CUANDO SE INDIQUE LO CONTRARIO POR ESCRITO LOS POSEEDORES DEL COPYRIGHT Y/O OTROS TERCEROS PROVEEN EL PROGRAMA “TAL CUAL ESTÁ” SIN GARANTÍAS DE TIPO ALGUNO,

YA SEAN EXPRESAS O IMPLÍCITAS, INCLUYENDO, PERO NO ESTANDO LIMITADO A, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN Y CONVENIENCIA PARA UN PROPÓSITO EN PARTICULAR. UD. ASUME TODOS LOS RIESGOS SOBRE LA CALIDAD Y RENDIMIENTO DEL PROGRAMA. SI EL PROGRAMA DEMUESTRA SER DEFECTUOSO, UD. ASUME EL COSTO DE CUALQUIER SERVICIO, REPARACIÓN O CORRECCIÓN NECESARIOS.

- EL POSEEDOR DEL COPYRIGHT, O CUALQUIER TERCERO QUE PUEDE MODIFICAR Y/O DISTRIBUIR EL PROGRAMA COMO SE PERMITE ARRIBA, NO ESTARÁ EXPUESTO DE MANERA ALGUNA A UD., A MENOS QUE SE REQUIERA POR LEY APLICABLE O SE ACUERDE POR ESCRITO, A DAÑOS INCLUYENDO CUALQUIER DAÑO GENERAL, ESPECIAL, INCIDENTE O CONSECUENTE DEBIDO AL USO O A LA IMPOSIBILIDAD DE HACER USO DEL PROGRAMA (INCLUYENDO, PERO NO LIMITADO A, LA PÉRDIDA DE DATOS O QUE LOS DATOS SE VUELVAN IMPRECISOS O PÉRDIDAS SOSTENIDAS POR UD. O TERCEROS O UNA FALLA DEL PROGRAMA PARA OPERAR CON CUALQUIER OTRO PROGRAMA), INCLUSO SI DICHO POSEEDOR U OTROS TERCEROS HAN SIDO AVISADOS DE LA POSIBILIDAD DE TALES DAÑOS.

FIN DE LOS TÉRMINOS Y CONDICIONES

A.3. Cómo aplicar estos Términos a sus programas nuevos

Si Ud. desarrolla un programa nuevo, y Ud. quiere que sea de la mayor utilidad posible al público, la mejor manera de hacer esto es hacerlo software libre que todos puedan redistribuir y cambiar bajo estos términos.

Para esto, agregue las notas siguientes al programa. Es más seguro agregarlas al comienzo de cada archivo fuente para hacer llegar la exclusión de la garantía de manera más efectiva; y cada archivo debe tener al menos la línea “copyright” y un puntero a donde se encuentra la nota completa.

<una línea para dar el nombre del programa y una idea breve de lo que hace.> Copyright (C) 20aa <nombre del autor>

Este programa es software libre; Ud. puede redistribuirlo y/o modificarlo bajo los términos de la Licencia Pública General GNU como fue publicada por la Free Software Foundation; ya sea la versión 2 de la Licencia, o (a su elección) cualquier versión posterior.

Este programa se distribuye con la esperanza de que será útil pero SIN GARANTÍA ALGUNA; incluso sin la garantía implícita de COMERCIALIZACIÓN o CONVENIENCIA PARA UN PROPÓSITO EN PARTICULAR. Vea la Licencia Pública General GNU para más detalles.

Ud. debería haber recibido una copia de la Licencia Pública General GNU junto con este programa; de no ser así, escriba a la Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

También agregue información sobre como ponerse en contacto con Ud. por medio de *correo-e* o correo postal.

Si el programa es interactivo, haga que el mismo muestre una pequeña nota como esta cuando inicia en un modo interactivo:

```
Gnomovision versión 69, Copyright (C) 20aa <nombre del autor>

Gnomovision viene ABSOLUTAMENTE SIN GARANTÍA;
para detalles ingrese ‘mostrar g’.

Este es software libre, y Ud. está alentado a redistribuirlo bajo ciertas
condiciones; ingrese ‘mostrar c’ para más detalles.
```

Los comandos hipotéticos “mostrar g” y “mostrar c” deberían mostrar las partes apropiadas de la Licencia Pública General. Por supuesto que los comandos que Ud. use pueden denominarse de otra forma en vez de

“mostrar g” y “mostrar c”; incluso pueden ser clicks con el ratón o elementos del menú – cualquier cosa que sea adecuada para su programa.

También debería hacer que su empleador (si Ud. trabaja como programador) o su escuela, si corresponde, firmen una “renuncia al copyright” para el programa, si es necesario. Aquí tiene un ejemplo; cambie los nombres adecuadamente:

Yoyodine, Inc., por la presente renuncia a todos los intereses del copyright del programa “Gnomovision” (que pasa a sus compiladores) escrito por Pedro Hacker.

<firma de Juan Perez>, 1 de Abril 2000 Juan Perez, Presidente de Compañía

Esta Licencia Pública General no permite incorporar a su programa dentro de programas propietarios. Si su programa es una biblioteca de subrutinas, Ud. puede considerar más útil el permitir enlazar aplicaciones propietarias con la biblioteca. Si esto es lo que Ud. desea hacer, use la Licencia Pública General de Biblioteca GNU en lugar de esta Licencia.

Apéndice B. Licencia de Documentación Libre GNU

Esta es una traducción al castellano de la Licencia de Documentación Libre GNU

This is an unofficial translation of the GNU Free Documentation Licens into spanish. It was not published by the Free Software Foundation, and does not legally state the distribution terms for documentation that uses the GNU FDL–only the original English text of the GNU FDL does that. However, we hope that this translation will help spanish speakers understand the GNU FDL better.

Esta es una traducción no oficial al castellano de la Licencia de Documentación Libre (FDL) GNU. No fue publicada por la Fundación de Software Libre (FSF), y legalmente no establece los términos de distribución de la documentación que usa la GNU FDL – sólo el texto original en inglés de la GNU FDL hace eso. Sin embargo, esperamos que esta traducción ayudará a las personas que hablan castellano a comprender mejor la FDL GNU.

Traducido por Fabian (Fabito) Mandelbaum en Marzo de 2001. París. Francia.

Versión 1.1, Marzo 2000

0. PREÁMBULO

El propósito de esta Licencia es hacer "libre"al manual, libro de texto, u otra documentación escrita, en el sentido de libertad: para asegurar a cualquiera la libertad efectiva de copiar o volver a distribuir el material, con o sin modificación, ya sea comercialmente o no comercialmente. En segundo término, esta Licencia preserva para el autor y para quien lo publica una forma de obtener crédito por su trabajo, a la vez que no se pueden considerar responsables por las modificaciones hechas por otros.

Esta Licencia es una especie de "copyleft", lo cual significa que los trabajos derivados del documento deben ser, en sí mismos, libres en el mismo sentido. La misma complementa a la Licencia Pública General GNU, la cual es una licencia "copyleft"diseñada para el software libre.

Hemos diseñado esta licencia para poder usarla para los manuales del software libre, debido a que el software libre necesita documentación libre: un programa libre debería estar acompañado de manuales que proporcionen las mismas libertades que proporciona dicho programa. Pero esta Licencia no está limitada a los manuales de software; la misma se puede usar para cualquier trabajo de textos, independientemente del tema o si se publica como libro impreso. Recomendamos esta Licencia principalmente para trabajos cuyo propósito sea instructivo o de referencia.

1. APLICABILIDAD y DEFINICIONES

Esta Licencia aplica a cualquier manual o cualquier otro trabajo que contiene una nota del propietario del copyright que diga que se puede distribuir bajo los términos de esta Licencia. En adelante, el "Documento"refiere a cualquiera de dichos manuales o trabajos. Cualquier miembro del público disfruta de la licencia, y se denominará al mismo como "Usted".

Una "Versión Modificada"del Documento significa cualquier trabajo que contiene al Documento o a una porción del mismo, ya sea copiada textualmente, o con modificaciones y/o traducida a otro idioma.

Una "Sección Secundaria"es un apéndice nombrado o una sección de carácter central del Documento que trata exclusivamente con la relación de los publicadores o autores del Documento con el tema general del Documento (o con temas relacionados) y no contiene cosa alguna que pueda caer directamente dentro de ese tema general. (Por ejemplo, si el Documento es una parte de un libro de texto sobre matemáticas, una Sección Secundaria puede no explicar matemática alguna.) La relación puede ser sólo una cuestión de conexión histórica con el tema o con temas relacionados, o de posición legal, comercial, filosófica, ética, o política respecto de dichos temas

Las "Secciones Invariantes"son ciertas Secciones Secundarias cuyos títulos se designan, como parte de esas Secciones Invariantes, en la nota que dice que el Documento se libera bajo esta Licencia.

Los "Textos de Cubierta"son ciertos pasajes de texto pequeños que se listan, tales como Textos de Tapa o Textos de Contratapa, en la nota que dice que el Documento se libera bajo esta Licencia.

Una copia "Transparente"del Documento significa una copia legible por una máquina, representada en un formato cuya especificación esté disponible al público general, cuyo contenido se pueda ver y editar directa e inmediatamente con editores de texto genéricos o (para las imágenes compuestas de pixels) programas genéricos de pintado o (para los dibujos) algún editor de dibujos disponible ampliamente, y que es adecuado como entrada de formateadores de texto o para la traducción automática a una variedad de formatos aptos para usar como entrada de formateadores de texto. Una copia hecha en un formato que de otra forma sería Transparente cuya anotación ha sido diseñada para frustrar o desalentar la modificación subsecuente hecha por sus lectores no es Transparente. Una copia que no es "Transparente"se denomina "Opaca".

Ejemplos de formatos adecuados para copias Transparentes incluyen ASCII plano sin anotación, formato de entrada Texinfo, formato de entrada LaTeX, SGML o XML usando un DTD disponible públicamente, y HTML simple de acuerdo a las normas diseñado para que las personas lo puedan modificar. Los formatos Opacos incluyen PostScript, PDF, formatos propietarios que sólo se pueden leer y editar con procesadores de texto propietarios, SGML o XML para el cual el DTD y/o las herramientas de procesado no están disponibles para el público en general, y el HTML generado por la máquina que producen algunos procesadores de texto sólo con propósitos de presentación.

La “Página del Título” significa, para un libro impreso, la página del título propiamente dicha, más tales páginas siguientes, necesarias para contener, legiblemente, el material que esta Licencia necesita que aparezca en la página del título. Para trabajos en formatos que no tienen página de título alguna, la “Página del Título” significa el texto que está cerca de la aparición más prominente del título del trabajo, que precede al comienzo del cuerpo del texto.

2. COPIADO TEXTUAL

Usted puede copiar y distribuir el Documento en cualquier medio, ya sea comercialmente o no comercialmente, siempre y cuando esta Licencia, las notas acerca del copyright, y la nota de la licencia que dice que esta Licencia aplica al Documento se reproduzcan en todas las copias, y que Ud. no agregue otras condiciones cualesquiera que sean a aquellas de esta Licencia. Usted no puede usar medidas técnicas para obstruir o controlar la lectura o la copia posterior de las copias que hace o distribuye. Sin embargo, puede aceptar una compensación a cambio de copias. Si distribuye una cantidad suficientemente grande de copias también debe seguir las condiciones de la sección 3.

También puede prestar copias, bajo las mismas condiciones que se indican arriba, y puede mostrar copias públicamente.

3. COPIADO EN CANTIDAD

Si publica copias impresas del Documento en cantidad mayor a 100, y la nota de la licencia del Documento necesita de Textos de Cubierta, debe incluir las copias en cubiertas que lleven, clara y legiblemente, todos esos Textos de Cubierta: los Textos de Tapa en la tapa, y los Textos de Contratapa en la contratapa. Ambas cubiertas también deben identificarlo clara y legiblemente como quien publica estas copias. La cubierta frontal debe presentar el título completo con todas las palabras o el título con igual prominencia y visibilidad. Adicionalmente, Usted puede agregar otro material sobre las cubiertas. El copiado con cambios limitados a las cubiertas, siempre y cuando las mismas preserven el título del Documento y satisfagan estas condiciones, se puede tratar como copiado textual en otros sentidos.

Si los textos que necesita cualquier cubierta son muy voluminosos para caber de forma legible, debería poner los primeros que se listan (tantos como quepan razonablemente) sobre la cubierta real, y continuar con el resto sobre las páginas adyacentes.

Si publica o distribuye copias Opacas del Documento en cantidad mayor a 100, debe incluir o bien una copia Transparente legible por una máquina junto con cada copia Opaca, o bien mencionar en o con cada copia Opaca una ubicación en una red de computadoras accesible públicamente que contenga una copia Transparente completa del Documento, libre de material agregado, que el público general que usa la red tenga acceso a transferir anónimamente sin cargo alguno usando protocolos de red públicos y normalizados. Si usa la última opción, debe tomar pasos razonablemente prudentes cuando comience la distribución de copias Opacas en cantidad, para asegurar que esta copia Transparente permanecerá accesible de esta forma en la ubicación mencionada por lo menos durante un año después de la última vez que distribuyó una copia Opaca (directamente o por medio de sus agentes o distribuidores) de esa edición al público.

Se pide, aunque no es necesario, que contacte a los autores del Documento con anterioridad suficiente antes de comenzar a redistribuir cualquier cantidad grande de copias, de forma de darles una oportunidad para proporcionarle a Ud. una versión actualizada del Documento.

4. MODIFICACIONES

Puede copiar y distribuir una Versión Modificada del Documento bajo las condiciones de las secciones 2 y 3 de arriba, siempre y cuando libere a la Versión Modificada precisamente bajo esta Licencia, con la Versión Modificada cumpliendo el rol del Documento, de forma tal que se licencia la distribución y modificación de la Versión Modificada a quienquiera posea una copia de la misma. Adicionalmente, debe hacer lo siguiente en la Versión Modificada:

A. Usar en la Página del Título (y sobre las cubiertas, si es que hay alguna) un título distinto del título del Documento, y de aquellos de las versiones previas (las cuales deberían, en caso que existan, estar listadas en la sección Historia del Documento). Usted puede usar el mismo título que una versión previa si el publicador original de esa versión da permiso.

B. Listar en la Página del Título, como autores, a una o más personas o entidades responsables por la autoría de las modificaciones en la Versión Modificada, junto con al menos cinco de los autores principales del Documento (todos sus autores principales, si el mismo tiene menos de cinco).

C. Mencionar en la Página del Título el nombre del editor de la Versión Modificada, como el editor.

D. Preservar todas las notas de copyright del Documento.

E. Agregar una nota de copyright apropiada para las modificaciones hechas por Ud. adyacente a las otras notas de copyright.

F. Incluir, inmediatamente después de las notas de copyright, una nota de licencia que da permiso al público general para usar la Versión Modificada bajo los términos de esta Licencia, en la forma que se muestra en el Apéndice más adelante.

G. Preservar en dicha nota de licencia las listas completas de las Secciones Invariantes y los Textos de Cubierta necesarios que se dan en la nota de licencia del Documento.

H. Incluir una copia sin alterar de esta Licencia.

I. Preservar la sección titulada "Historia" y el título de la misma, y agregar a la misma un ítem que mencione al menos el título, año, autores nuevos, y publicador de la Versión Modificada como se da en la Página del Título. Si en el Documento no hay sección alguna titulada "Historia", crear una que mencione el título, año, autores, y publicador del Documento como se da en la Página del Título, luego agregar un ítem que describa la Versión Modificada como se mencionó en la oración anterior.

J. Preservar la ubicación de red, si hay alguna, dada en el Documento para el acceso público a una copia Transparente del Documento, y de la misma manera las ubicaciones de red dadas en el Documento para las versiones previas en la que el mismo se basa. Estas pueden colocarse en la sección "Historia". Usted puede omitir una ubicación de red para un trabajo que fue publicado al menos cuatro años antes que el Documento propiamente dicho, o si el publicador original de la versión a la cual se refiere da permiso.

K. En cualquier sección titulada "Reconocimientos" o "Dedicatorias", preservar el título de la sección, y preservar en la sección toda la sustancia y el tono de cada uno de los reconocimientos a los contribuyentes y/o dedicaciones aquí mencionados.

L. Preservar todas las Secciones Invariantes del Documento, inalteradas en su texto y en sus títulos. Los números de sección o el equivalente no se consideran parte de los títulos de sección.

M. Borrar cualquier sección titulada "Aprobaciones". Tal sección no puede incluirse en la Versión Modificada.

N. No cambiar el título de sección alguna por "Aprobaciones" o de forma tal que genere un conflicto con cualquier Sección Invariante.

Si la Versión Modificada incluye nuevas secciones o apéndices de carácter central que califican como Secciones Secundarias y no contienen material copiado del Documento, Ud. puede a su elección designar a alguna o todas estas secciones como invariantes. Para hacer esto, agregue los títulos de las mismas a la lista de Secciones Invariantes en la nota de licencia de la Versión Modificada. Estos títulos deben ser distintos de los títulos de cualquier otra sección.

Usted puede agregar una sección titulada "Aprobaciones", siempre y cuando no contenga otra cosa que aprobaciones de terceros de su Versión Modificada–por ejemplo, menciones de revisiones hechas por sus pares o que el texto ha sido aprobado por una organización como la definición fidedigna de una normativa.

Puede agregar un pasaje de hasta cinco palabras como un Texto de Tapa, y un pasaje de hasta veinticinco palabras como un Texto de Contratapa, al final de la lista de Textos de Cubierta en la Versión Modificada. Sólo puede agregarse un pasaje del Texto de Tapa y uno del Texto de Contratapa por medio de una entidad (o por medio de contratos hechos con una). Si el Documento ya incluye un texto de cubierta para la misma cubierta, agregado con anterioridad por Ud. o por un contrato hecho por la misma entidad en nombre de la cual Ud. está actuando, no puede agregar otro; pero puede reemplazar el anterior, sobre permiso explícito del publicador previo que agregó el anterior.

El(Los) autor(es) y publicador(es) del Documento no dan por medio de esta Licencia permiso para usar sus nombres para publicidad para o para imponer o implicar atribución de cualquier Versión Modificada.

5. COMBINANDO DOCUMENTOS

Usted puede combinar el Documento con otros documentos liberados bajo esta Licencia, bajo los términos definidos en la sección 4 de arriba para las versiones modificadas, siempre y cuando incluya en la combinación todas las Secciones Invariantes de todos los documentos originales, sin modificaciones, y las liste a todas como Secciones Invariantes de su trabajo combinado en la nota de licencia del mismo.

El trabajo combinado sólo debe contener una copia de esta Licencia, y múltiples Secciones Invariantes idénticas se pueden reemplazar con una copia única. Si hay múltiples Secciones Invariantes con el mismo nombre

pero contenido diferente, haga que el título de cada una de dichas secciones sea único, agregando al final del mismo, entre paréntesis, el nombre del autor o editor original de esa sección si se conoce, o de lo contrario un número único. Haga el mismo ajuste a los títulos de sección en la lista de Secciones Invariantes en la nota de licencia del trabajo combinado.

En la combinación, Ud. debe combinar cualquier sección titulada "Historia"en los distintos documentos originales, formando una sección titulada "Historia"; de la misma forma, combine cualquier sección titulada "Agradecimientos", y cualquier sección titulada "Dedicatorias". Usted debe borrar todas las secciones tituladas "Aprobaciones."

6. COLECCIONES DE DOCUMENTOS

Usted puede hacer una colección que consista del Documento y otros documentos liberados bajo esta Licencia, y reemplazar las copias individuales de esta Licencia en los distintos documentos con una única copia que se incluya en la colección, siempre y cuando siga las reglas de esta Licencia para copiado textual de cada uno de los documentos en todos los otros sentidos.

Puede extraer un único documento de tal colección, y distribuirlo individualmente bajo esta Licencia, siempre y cuando inserte una copia de esta Licencia dentro del documento extraído, y siga esta Licencia en todos los demás sentidos que traten con el copiado literal de ese documento.

7. AGREGACIÓN CON TRABAJOS INDEPENDIENTES

Una compilación del Documento o sus derivados con otros documentos o trabajos separados e independientes, en o sobre un volumen de un medio de almacenamiento o distribución, no cuenta como un todo como una Versión Modificada del Documento, siempre y cuando no se reclame copyright alguno sobre la compilación. Una compilación tal, se denomina una "agregación", y esta Licencia no aplica a los demás trabajos autocontenidos compilados por lo tanto con el Documento, o a cuenta de haber sido compilados de esta manera, si ellos mismos no son trabajos derivados del Documento.

Si el requisito de Texto de Cubierta de la sección 3 se aplica a estas copias del Documento, entonces si el Documento es menos que un cuarto de toda la agregación, los Textos de Cubierta del Documento pueden colocarse en cubiertas que rodeen sólo al Documento dentro de la agregación. De no ser así los mismos deben aparecer en las cubiertas alrededor de la agregación completa.

8. TRADUCCIÓN

La traducción se considera una especie de modificación, por lo tanto puede distribuir traducciones del Documento bajo los términos de la sección 4. El reemplazo de Secciones Invariantes con traducciones requiere permiso especial de los propietarios del copyright, pero Ud. puede incluir traducciones de alguna o de todas las Secciones Invariantes además de las versiones originales de estas Secciones Invariantes. Puede incluir una traducción de esta Licencia siempre y cuando también incluya la versión original en Inglés de esta Licencia. En caso de desacuerdo entre la traducción y la versión original en Inglés de esta Licencia, prevalecerá la versión original en Inglés.

9. TERMINACIÓN

Usted no puede copiar, modificar, sub-licenciar, o distribuir el Documento excepto como se ha previsto para tales fines bajo esta Licencia. Cualquier otro intento de copiar, modificar, sub-licenciar o distribuir el Documento es nulo, y terminará automáticamente sus derechos bajo esta Licencia. Sin embargo, las partes que han recibido copias, o derechos, de parte de Ud. bajo esta Licencia no harán que terminen sus respectivas licencias mientras que dichas partes permanezcan en completo cumplimiento.

10. REVISIONES FUTURAS DE ESTA LICENCIA

De vez en cuando, la Fundación del Software Libre (Free Software Foundation) puede publicar versiones nuevas, revisadas de la Licencia de Documentación Libre GNU. Tales versiones nuevas serán similares en espíritu a la presente versión, pero pueden diferir en detalle para solucionar problemas o preocupaciones nuevas. Vea copyleft (http://www.gnu.org/copyleft/).

A cada versión de la Licencia se la da un número de versión distintivo. Si el Documento especifica que un número de versión de esta Licencia en particular "o cualquier otra versión posterior"aplica al mismo, Ud. tiene la opción de seguir los términos y condiciones de cualquier aquella versión especificada o de cualquier versión posterior que ha sido publicada (no como borrador) por la Free Software Foundation. Si el Documento no especifica un número de versión de esta Licencia, puede elegir cualquier versión que haya sido publicada (no como borrador) por la Free Software Foundation.

Cómo usar esta Licencia para sus documentos

Para usar esta Licencia en un documento que Ud. ha escrito, incluya una copia de la Licencia en el documento y ponga las notas del copyright y la licencia siguientes justo después de la página del título:

> Copyright (c) AÑO SU NOMBRE. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

Si no tiene Sección Invariante alguna, escriba "sin Secciones Invariantes"en vez de decir cuales secciones son invariantes. Si no tiene Textos de Tapa, escriba "sin Textos de Tapa"en vez de "Con los Textos de Tapa LISTA"; de la misma forma para los Textos de Contratapa.

Si su documento contiene ejemplos no triviales de código de programa, recomendamos liberar estos ejemplos en paralelo bajo su elección de licencia de software libre, tal como la Licencia Pública General GNU, para permitir el uso de los ejemplos en software libre.

Apéndice C. GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free"in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version"of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section"is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections"are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts"are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent"copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent"is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple

HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page"means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page"means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History"in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/ copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version"applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections"instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts"instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Glosario

APM

Advanced Power Management (Administración avanzada de energía). Característica usada por algunos *BIOS* para hacer que la máquina entre en modo de reposo luego de un período de inactividad determinado. En las portátiles, APM también es el responsable de reportar el estado de la batería y, si esta lo soporta, el tiempo estimado de vida.

ASCII

American Standard Code for Information Interchange (Código Estándar Americano para el Intercambio de Información). El código estándar que se usa para almacenar caracteres, incluyendo a los caracteres de control, en una computadora. Muchos códigos de 8 bits (tales como el ISO 8859-1, el conjunto de caracteres predeterminado de *GNU/Linux*) contienen al ASCII como su mitad inferior (Ver *ISO 8859*).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

Figura 1. Tabla ASCII

BSD

Berkeley Software Distribution (Distribución de software de Berkeley). Es una variante de *Unix* desarrollada en el departamento de computación de la Universidad de Berkeley. Esta versión siempre ha sido considerada técnicamente más avanzada que las otras, y ha contribuido muchas innovaciones al mundo de la computación en general y al de *Unix* en particular.

CHAP

Challenge-Handshake Authentication Protocol (Protocolo de Autenticación de Desafío-Apretón de manos). Protocolo usado por los ISPs para autenticar a sus clientes. En este esquema, se envía un valor al cliente (la máquina que se conecta), el cliente calcula un *hash* a partir de este valor y se lo envía al servidor, y el servidor compara el *hash* con el que él mismo calculó.

Ver también: PAP.

CIFS

Common Internet FileSystem (Sistema de Archivos Común de *Internet*) El predecesor del sistema de archivos SMB, usado en los sistemas *DOS*.

DHCP

Dynamic Host Configuration Protocol (Protocolo de Configuración Dinámica del Host). Un protocolo diseñado para que las máquinas de una red local obtengan una dirección IP dinámicamente.

DMA

Direct Memory Access (Acceso Directo a Memoria). Característica usada por la arquitectura de *PC*; que permite que un periférico lea o escriba de la memoria principal sin intervención de la CPU. Los dispositivos PCI usan "bus mastering" (apropiación del bus) y no necesitan DMA.

DNS

Domain Name System (Sistema de Nombres de Dominio). El mecanismo de direcciones/nombres distribuido que se usa en *Internet*. Este mecanismo le permite mapear un nombre de dominio a una dirección IP, que es lo que le deja buscar un sitio por el nombre de dominio sin conocer la dirección IP de dicho sitio.

DPMS

Display Power Management System (Sistema de Administración de Energía del Monitor). Protocolo usado por todos los monitores modernos para manipular las características de administración de energía. Los monitores que soportan estas características generalmente se denominan "ecológicos".

ELF

Executable and Linking Format (Formato de Vinculado y de Ejecutables). Hoy día, este es el formato binario usado por la mayoría de las distribuciones *GNU/Linux*.

ext2

Es una abreviatura para el "segundo sistema de archivos extendido." ext2 es el sistema de archivos nativo de *GNU/Linux*. El beneficio de utilizar ext2 en lugar de los sistemas de archivos más antiguos, tales como FAT, o incluso FAT32, es que este ofrece alto rendimiento, nombres de archivo largos, permisos sobre los archivos, y una tolerancia mayor frente a los errores.

FAQ

Frequently Asked Questions (Preguntas Formuladas Frecuentemente): documento que contiene una serie de preguntas/respuestas acerca de un tema específico. Históricamente aparecieron en los foros de discusión, pero ahora este tipo de documento aparece en varios sitios *web*, e incluso hay productos comerciales que tienen su FAQ. Generalmente, son fuentes de información muy buenas.

FAT

File Allocation Table (Tabla de Ubicación de Archivos). Sistema de archivos usado por *DOS* y las primeras versiones de *Windows*. Las versiones más modernas de *Windows* usan una variante de FAT denominada FAT32.

FDDI

Fiber Distributed Digital Interface (Interfaz Digital Distribuída de Fibra). Una capa física de red de alta velocidad, que usa fibra óptica para las comunicaciones. Sólo se usa en redes grandes, principalmente debido a su costo.

FHS

Filesystem Hierarchy Standard (Normativa para la Jerarquía de un Sistema de Archivos). Un documento que contiene guías y consejos para una organización coherente del árbol de archivos en sistemas *Unix*. **Linux-Mandrake** cumple con esta normativa en la mayoría de sus aspectos.

FIFO

First In, First Out (Primero en Llegar, Primero en Salir). Una estructura de datos o un buffer de hardware donde los elementos se quitan en el orden en el que fueron puestos. Las tuberías de *Unix* son

el ejemplo más común de FIFO. En la programación estas estructuras también se conocen con el nombre de “cola”.

FTP

File Transfer Protocol (Protocolo de Transferencia de Archivos). Este es el protocolo típico de *Internet* usado para transferir archivos desde una máquina a otra.

GIF

Graphics Interchange Format (Formato de Intercambio de Gráficos). Un formato de archivos de imagen, ampliamente usado en la *web*. Las imágenes GIF pueden estar comprimidas o animadas. Debido a problemas con el copyright no es una buena idea usarlas, reemplácelas tanto como sea posible con el formato PNG que es mucho más avanzado.

GNU

GNU's Not Unix (GNU No es Unix). El proyecto GNU ha sido iniciado por Richard Stallman al comienzo de los años ’80 y tiene como objetivo el desarrollo de un sistema operativo libre (“libre” como en libertad de opinión). Corrientemente, todas las herramientas están allí, excepto... el núcleo. El núcleo del proyecto GNU, *Hurd*, todavía no es “duro como una roca”. *Linux* toma prestadas, entre otras, dos cosas de GNU: su compilador *C*, `gcc`, y su licencia, la GPL.

Ver también: GPL.

GPL

General Public License (Licencia Pública General). La licencia del núcleo de *GNU/Linux*, va en la dirección contraria a todas las licencias propietarias en el sentido de que no pone restricción alguna a la copia, modificación y redistribución del software, con la condición de que el código fuente esté disponible. La única restricción, si es que se la puede denominar así, es que las personas a las cuales Ud. redistribuye el software también se deben beneficiar con los mismos derechos.

GUI

Graphical User Interface (Interfaz Gráfica de Usuario). Un programa que usa menús, botones, colores, y fuentes diferentes para parecer más fácil de usar a primera vista. Necesita un servidor *X*.

HTML

HyperText Markup Language (Lenguaje de Marcado de HiperTexto). El lenguaje que se usa para crear documentos *web*.

HTTP

HyperText Transfer Protocol (Protocolo de Transferencia de HiperTexto). El protocolo que se usa para conectarse a sitios *web* y recuperar documentos HTML.

IDE

Integrated Drive Electronics (Electrónica de Disco Integrada). En las *PC* de hoy día es el bus de disco más usado. Un bus IDE puede contener hasta dos dispositivos, y la velocidad del bus está limitada por el dispositivo conectado que tiene la cola de comandos más lenta (¡y no la velocidad de transferencia menor!).

Ver también: ATAPI.

IP, dirección

Es una dirección numérica que consiste de cuatro partes que identifica a su computadora en *Internet*, o cualquier otra red basada en TCP/IP. Las direcciones IP están estructuradas de forma jerárquica, con los dominios de nivel superior y los dominios nacionales, los dominios, los sub-dominios y la dirección

personal de cada máquina. Una dirección IP luciría como *192.168.0.1*. La dirección personal de una máquina puede ser o bien estática o bien dinámica. Las direcciones IP estáticas son direcciones que nunca cambian, sino que son más bien permanentes. Las direcciones IP dinámicas son aquellas que pueden cambiar. Los usuarios de acceso telefónico y cable-módem tienen direcciones IP típicamente dinámicas mientras que algunas conexiones DSL y otras conexiones de velocidad mayor proporcionan direcciones IP estáticas.

IP, enmascarado de

Es cuando Ud. usa un cortafuegos para ocultar del exterior la dirección IP verdadera de su computadora. Típicamente, cualquier conexión de red externa que Ud. realice más allá del cortafuegos heredará la dirección IP del cortafuegos. Esto es útil en situaciones donde Ud. debe tener una conexión con *Internet* rápida con una dirección IP única pero desea utilizar más de una computadora que tienen asignadas direcciones IP de la red interna.

IRC

Internet Relay Chat (Charla Interactiva en *Internet*). Una de los pocas normas de *Internet* para charlas en vivo. Permite la creación de canales, las charlas privadas, y también el intercambio de archivos. También está diseñada para poder hacer que los servidores se conecten unos con otros, que es la razón por la cual hoy día existen varias redes IRC: **Undernet**, **DALnet**, **EFnet** para nombrar algunas.

ISA

Industry Standard Architecture (Arquitectura Estándar de la Industria). El primer bus de todos los usados en las *PC*, está siendo abandonado lentamente en favor del bus PCI. Sin embargo, algunos fabricantes de hardware siguen usándolo. Todavía es muy común que las placas SCSI que se proveen con los rastreadores, las grabadoras de CD... sean ISA. ¡Qué lastima!

ISO

International Standards Organisation (Organización de Normas Internacionales). Grupo de compañías, consultores, universidades y otras fuentes que elaboran normativas sobre varios temas, incluyendo a la computación. Las normas están numeradas. Por ejemplo, la norma número 9660, describe al sistema de archivos que usan los CD-ROM.

ISP

Internet Service Provider (Proveedor de Servicios de *Internet*). Compañía que vende accesos a *Internet* a sus clientes, ya sea por línea telefónica o líneas dedicadas.

JPEG

Join Photographic Experts Group (Grupo de Expertos en Fotografía). Otro formato de archivo de imagen muy común. JPEG está optimizado para comprimir imágenes realísticas (paisajes, gente, etc.), y no funciona muy bien con imágenes no-realísticas.

LAN

Local Area Network (Red de Área Local). Nombre genérico dado a una red de máquinas conectadas al mismo cable físico.

LDP

Linux Documentation Project (Proyecto de Documentación de *GNU/Linux*). Una organización sin fines de lucro que mantiene la documentación de *GNU/Linux*. Sus documentos más conocidos son los COMOs, pero también mantiene las FAQ, e incluso algunos libros.

MBR

Master Boot Record (Registro de Arranque Maestro). Nombre dado al primer sector de un disco rígido del cual se puede arrancar. El MBR contiene el código usado para cargar el sistema operativo en memoria o un cargador de arranque (como *LILO*), y la tabla de particiones de este disco rígido.

MIME

Multipurpose Internet Mail Extensions (Extensiones de Correo de *Internet* de propósitos Múltiples). Una cadena de la forma `tipo/sub-tipo` que describe el contenido de un archivo adjuntado a un correo electrónico. Esto permite a los clientes que reconozcan MIME definir acciones en función del tipo de archivo.

MPEG

Moving Pictures Experts Group (Grupo de Expertos de Imágenes en Movimiento). Un comité de la ISO que genera normas para la compresión de audio y vídeo. MPEG también es el nombre de los algoritmos para efectuar dicha compresión. Desafortunadamente, este formato es muy restrictivo, y como consecuencia todavía no hay reproductores MPEG de código abierto...

NCP

NetWare Core Protocol (Protocolo de Base de NetWare). Protocolo definido por **Novell** para acceder a los servicios de archivos e impresión de Novell NetWare.

NFS

Network FileSystem (Sistema de Archivos de Red). Un sistema de archivos de red creado por **Sun Microsystems** para poder compartir archivos en una red de forma transparente.

NIC

Network Interface Card (Tarjeta Interfaz de Red). Adaptador instalado en una computadora que provee una conexión física a la red, tal como una tarjeta *Ethernet*.

NIS

Network Information System (Sistema de Información de Red). También conocido como “Yellow Pages” (Páginas amarillas), pero **British Telecom** tiene un copyright de ese nombre. NIS es un protocolo diseñado por **Sun Microsystems** para poder compartir información común a lo largo de un **dominio** NIS, que puede agrupar toda una red LAN, parte de una red LAN o varias LAN. Puede exportar bases de datos de contraseñas, bases de datos de servicios, información de grupos y más.

PAP

Password Authentication Protocol (Protocolo de Autenticación de Contraseña): protocolo usado por los ISPs para autenticar a sus clientes. En este esquema, el cliente (Ud.) envía un par identificador/contraseña al servidor, que no está cifrado.

Ver también: CHAP.

PCI

Peripheral Components Interconnect (Interconexión de Componentes Periféricos). Un bus creado por **Intel** que hoy día es el bus típico de la arquitectura *PC*, aunque también lo usan otras arquitecturas. Es el sucesor del bus ISA, y ofrece numerosos servicios: identificación del dispositivo, información de la configuración, compartir IRQ, apropiación del bus (bus mastering) y más.

PCMCIA

Personal Computer Memory Card International Association (Asociación Internacional de Tarjetas de Memoria de Computadoras Personales): más y más comunmente denominadas “PC Card” por razones de simplicidad, esta es la norma para tarjetas externas que se insertan en las portátiles: módems,

discos rígidos, tarjetas de memoria, tarjetas *Ethernet* y más. A veces el acrónimo en inglés se expande, en broma a *People Cannot Memorize Computer Industry Acronyms* (La Gente No Puede Memorizar los Acrónimos de la Industria de Computadoras)...

PNG

Portable Network Graphics (Gráficos de Red Portables). Formato de archivo de imagen creado principalmente para su uso en la *web*, ha sido diseñado como un reemplazo de GIF libre de patentes y también tiene algunas características adicionales.

PnP

Plug'N'Play (Enchufar Y Usar). Al principio era un agregado al bus ISA para poder agregar información de configuración para los dispositivos. Se ha vuelto un término de uso más amplio que agrupa a todos los dispositivos capaces de reportar sus parámetros de configuración. Como tales, todos los dispositivos PCI son Plug'N'Play.

POP

Post Office Protocol (Protocolo de Oficina de Correos). Es el protocolo común utilizado para transferir el correo desde un ISP.

PPP

Point to Point Protocol (Protocolo de Punto a Punto). Este es el protocolo que se usa para enviar datos a través de las líneas serie. Es común su uso para enviar paquetes IP a *Internet*, pero también se puede usar con otros protocolos tales como el protocolo IPX de **Novell**.

RAID

Redundant Array of Independent Disks (Matriz Redundante de Discos Independientes). Proyecto iniciado por el departamento de ciencias de la computación de la Universidad de Berkeley, en el cual el almacenamiento de datos se "reparte" en una matriz de discos.

RAM

Random Access Memory (Memoria de Acceso Aleatorio). Término usado para identificar a la memoria principal de una computadora.

RFC

Request For Comments (Pedido De Comentarios). Los RFC son los documentos oficiales normativos de *Internet*. Describen todos los protocolos, su uso, sus requisitos, y así sucesivamente. Cuando Ud. quiera aprender como funciona un protocolo, debe leer el RFC correspondiente.

RPM

Redhat Package Manager (Administrador de Paquetes de **Red Hat**). Un formato de empaquetado desarrollado por **Red Hat** para crear paquetes de software, que se usa en muchas distribuciones de *GNU/Linux*, incluida **Linux-Mandrake**.

SCSI

Small Computers System Interface (Interfaz de Sistema para Computadoras Pequeñas). Un bus de alto rendimiento diseñado para permitir varios tipos de periféricos. A diferencia de IDE, un bus SCSI no está limitado por la velocidad a la cual los periféricos pueden aceptar comandos. Sólo las máquinas de alto nivel integran un bus SCSI directamente en la placa madre. Las *PC* necesitan agregar una tarjeta.

SMB

Server Message Block (Bloque de Mensaje del Servidor). Protocolo usado por las máquinas *Windows* (*9x* o *NT*) para compartir archivos e impresoras en una red. Ver también **CIFS**.

SMTP

Simple Mail Transfer Protocol (Protocolo Simple de Transferencia de Correo). Este es el protocolo más común para transferir *correo-e*. Los Agentes de Transmisión de Correo (MTAs) tales como *sendmail* o *postfix* usan SMTP. A veces también se los denomina servidores SMTP.

SVGA

Super Video Graphics Array (SuperMatriz Gráfica de Vídeo). Norma de modo de vídeo definida por VESA para la arquitectura *PC*. La resolución es 800×600 puntos con 16 colores.

TCP

Transmission Control Protocol (Protocolo de Control de la Transmisión). Este es el protocolo confiable más común que usa a IP para transferir paquetes de la red. TCP agrega las verificaciones necesarias encima de IP para asegurarse que los paquetes se entregan.

URL

Uniform Resource Locator (Ubicador de Recursos Uniforme). Una cadena de caracteres con un formato especial usado para identificar unívocamente un recurso en *Internet*. Dicho recurso puede ser un archivo, un servidor, u otros. La sintaxis de una URL es

`protocolo://servidor.nombre[:puerto]/ruta/al/recurso`.

Cuando sólo se especifica el nombre de una máquina y el protocolo es `http://`, predeterminadamente se recupera el archivo `index.html` del servidor.

VESA

Video Electronics Standards Association (Asociación de Normas Electrónicas de Vídeo). Una asociación normativa de la industria que apunta a la arquitectura de *PC*. Por ejemplo, es la autora de la norma SVGA.

WAN

Wide Area Network (Red de Área Extensa). Esta red, si bien es similar a una red LAN, conecta a computadoras sobre una red que no está físicamente conectada a los mismos cables y están separadas por una distancia mayor.

alias

Mecanismo usado en un *shell* para hacer que este substituya una cadena por otra antes de ejecutar un comando. Ud. puede ver todos los alias definidos en la sesión corriente ingresando `alias` en el *prompt*.

archivo oculto

Es un archivo que no se puede "ver" cuando se ejecuta un comando `ls` sin opciones. Los nombres de los archivos ocultos comienzan con un `.` y casi siempre se los utiliza para almacenar las preferencias y configuraciones personales del usuario para los distrintos programas que usa. Por ejemplo, la historia de comandos de *Bash* se guarda en `.bash_history`, que es un archivo oculto.

archivos, sistema de

También conocido como *filesystem*. Es el esquema usado para poder almacenar archivos en un medio físico (disco rígido, disquete) en una manera consistente. Son ejemplos de sistemas de archivos: FAT, el ext2fs de *GNU/Linux*, ISO-9660 (usado por los CD-ROM) y así sucesivamente.

ARP

Address Resolution Protocol (Protocolo de Resolución de Direcciones). El Protocolo de *Internet* que se usa para hacer corresponder dinámicamente las direcciones *Internet* a direcciones físicas (hardware) sobre redes de área local. Esto está limitado a redes que soportan la difusión por hardware.

arranque

También conocido como *boot*. Es el procedimiento que toma lugar cuando se enciende una computadora, donde se reconocen los periféricos uno tras otro, y donde se carga en memoria el sistema operativo.

arranque, cargador de

También conocido como *bootloader*. Es un programa que inicia el sistema operativo. Muchos cargadores de arranque le brindan la oportunidad de cargar más de un sistema operativo permitiéndole elegir entre los mismos dentro de un menú de arranque. Los cargadores de arranque como *GRUB* son populares gracias a esta característica y son muy útiles en sistemas de arranque dual o múltiple.

arranque, disquete de

También conocido como *bootdisk*, es un disquete que puede arrancar y contiene el código necesario para cargar el sistema operativo desde el disco rígido (a veces es auto-suficiente – es decir, no carga el sistema operativo desde el disco, sino desde sí mismo).

ATAPI

("AT Attachment Packet Interface"). Es una extensión de la especificación ATA ("Advanced Technology Attachment") conocida comunmente con el nombre de IDE ("Integrated Drive Electronics") que proporciona comandos adicionales para controlar las unidades de CD-ROM y las unidades de cinta. Los controladores IDE que poseen estas características se denominan EIDE ("Enhanced IDE").

ATM

Es un acrónimo de *Asynchronous Transfer Mode* (Modo de Transferencia Asincrónico). Una red ATM empaqueta los datos en bloques de tamaño normalizado (53 bytes: 48 de datos y 5 de cabecera) que puede transportar eficientemente de un punto a otro. ATM es una tecnología de red de paquetes de circuitos conmutados orientada a las redes ópticas de alta velocidad (multi-megabits).

atómico

Se dice que un conjunto de operaciones es atómico cuando se ejecuta todo de una vez, y no se puede interrumpir.

beta testing

Es el nombre que se da al proceso de probar la versión beta de un programa. Usualmente los programas se sacan en etapas alfa y beta para la prueba del mismo antes de sacar la versión "final".

biblioteca

Es una colección de procedimientos y funciones en formato binario para que los programadores usen en sus programas (siempre y cuando la licencia de la biblioteca en cuestión se los permita). El programa encargado de cargar las bibliotecas compartidas en tiempo de ejecución se denomina "vinculador dinámico".

bip

es el pequeño ruido que hace el parlante de su computadora para avisarle acerca de alguna situación ambigua cuando Ud. está utilizando el completado de la línea de comandos y, por ejemplo, hay más de una elección posbile para completar. Puede haber otros programas que hagan bips para hacerle saber de alguna situación en particular.

bit

Del inglés *BInary digiT* (Dígito binario). Un solo dígito que puede tomar los valores 0 o 1, dado que el cálculo se hace en base dos. Unidad elemental de información binaria.

buffer

Una porción de memoria pequeña de tamaño fijo, que puede ser asociada a un archivo de modo de bloques, a una tabla del sistema, a un proceso, y así sucesivamente. El buffer cache mantiene la coherencia de todos los buffers.

Ver también: buffer cache.

buffer cache

Una parte crucial del núcleo de un sistema operativo. Tiene a su cargo mantener todos los buffers actualizados, compactando el cache cuando sea necesario, borrando los buffers innecesarios y más.

Ver también: buffer.

bug

Comportamiento ilógico o incoherente de un programa en un caso especial, o comportamiento que no sigue la documentación entregada con el programa. Generalmente, las características nuevas en los programas introducen bugs nuevos. Error de programación.

byte

Octeto. Paquete de ocho bits consecutivos, interpretados en base dos como un número entre 0 y 255.

Ver también: bit.

capitalización

Cuando se toma en el contexto de las cadenas de caracteres, es la distinción entre mayúsculas (o letras capitales) y minúsculas.

cliente

Programa o computadora que esporádicamente, y por un tiempo dado, se conecta a otro programa u otra computadora para darle órdenes o pedirle información. En el caso de un sistema **de igual a igual** (*peer to peer*) tales como PPP o SLIP el cliente se toma como el extremo de la conexión que inicia la llamada y el otro extremo se toma como servidor. Es uno de los componentes de un **sistema cliente/servidor**.

cliente/servidor, sistema

Sistema o protocolo que consiste de un **servidor** y de uno o varios **clientes**.

código objeto

Es el código generado por el proceso de compilación para ser vinculado con otros códigos objeto y bibliotecas para formar un archivo ejecutable. El código objeto es legible por la máquina.

comando, modo de

Bajo *VI* o uno de sus clones, es el estado del programa en el cual la presión de una tecla (esto, por sobre todo se refiere a las letras) no resultará en la inserción de la letra correspondiente en el archivo editado, sino que efectuará una acción específica a la tecla en cuestión (a menos que el clon tenga comandos que se puedan cambiar y Ud. haya personalizado su configuración). Ud. puede salir de este modo ingresando uno de los comandos que lo llevarán de vuelta al modo de inserción: **i**, **I**, **a**, **A**, **s**, **S**, **o**, **O**, **c**, **C**, ...

comandos, línea de

Lo que proporciona el *shell* y le permite al usuario ingresar comandos directamente. También es el sujeto de una “flame war” eterna entre sus adeptos y sus detractores **:-)**

comodín

Los caracteres '*' y '?' se utilizan como caracteres comodín y pueden representar cualquier cosa. El '*' representa cualquier cantidad de caracteres incluyendo a ningún caracter. El '?' representa exactamente un caracter. A menudo los comodines se usan en las expresiones regulares.

compilación

Es el proceso de traducir código fuente que una persona puede leer (bueno, con un poco de práctica) y que está escrito en algún lenguaje de programación (por ejemplo, *C*) en un archivo binario que puede leer la máquina.

completado

Capacidad de un *shell* para expandir automáticamente una sub-cadena a un nombre de archivo, un nombre de usuario u otros, siempre y cuando la sub-cadena no sea ambigua.

compresión

Es una forma de encoger archivos o disminuir la cantidad de caracteres que se envían por un vínculo de comunicaciones. *compress*, zip, gzip, y bzip2 se cuentan entre algunos programas de compresión.

consola

Es el nombre que se da a lo que generalmente se denominaban terminales. En los sistemas *GNU/Linux*, Ud. tiene lo que se denominan consolas virtuales que le permiten usar una pantalla o monitor para múltiples sesiones independientes. Predeterminadamente, tiene seis consolas virtuales a las que se acceden presionando **ALT-F1** hasta **ALT-F6**. También hay una séptima consola virtual, **ALT-F7**, que le permitirá usar el X Window System. En X, puede pasarse a la consola de texto presionando **CTRL-ALT-F1** hasta **CTRL-ALT-F6**.

consola virtual

Es el nombre que se le da a lo que se solían denominar terminales. En los sistemas *GNU/Linux*, Ud. tiene lo que se llaman consolas virtuales que le permiten usar una pantalla o monitor para muchas sesiones que corren independientes unas de otras. De manera predeterminada, Ud. tiene seis consolas virtuales a las que puede acceder presionando **ALT-F1** hasta **ALT-F6**. De manera predeterminada, hay una séptima consola virtual, **ALT-F7**, que le permite acceder al Sistema X Window que está ejecutando. En X, puede acceder a la consola de texto presionando **CTRL-ALT-F1** hasta **CTRL-ALT-F6**.

Ver también: consola.

contraseña

Es una palabra, o una combinación de palabras y letras, secreta que se usa para asegurar alguna cosa. Las contraseñas se usan en conjunto con las conexiones de usuario (login) en los sistemas operativos multiusuario, sitios *web*, sitios FTP, y así sucesivamente. Las contraseñas deberían ser frases o combinaciones alfanuméricas difíciles de adivinar y nunca deberían basarse en palabras comunes del diccionario. Las contraseñas aseguran que otras personas no se pueden conectar a una computadora o a un sitio usando la cuenta de Ud.

cookies

Archivos temporales que un servidor *web* remoto escribe en el disco rígido local. Los "cookies" le permiten al servidor estar informado de las preferencias del usuario cuando este se vuelva a conectar.

copia de respaldo (*backup*)

Es una forma de guardar sus datos importantes en un medio y ubicación seguros. Las copias de respaldo deberían realizarse regularmente, especialmente con los archivos de configuración y la información más crítica (los directorios principales de los cuales se debe hacer copia de seguridad son /etc, /home, y

`/usr/local`). Tradicionalmente, mucha gente usa `tar` con `gzip` o `bzip2` para hacer copia de respaldo de los directorios y los archivos. Ud. puede utilizar estas herramientas o programas como `dump` y `restore`, junto con muchas otras soluciones libres o comerciales de copia de respaldo.

correo-e

Significa Correo Electrónico. Esta es la forma de enviar mensajes electrónicamente entre personas sobre la misma red. Al igual que con el correo común (también conocido como correo postal), el *correo-e* necesita un destino y la dirección del remitente para ser enviado adecuadamente. El remitente debe tener una dirección de la forma "remitente@dominio.del.remitente" y el destinatario debe tener una dirección de la forma "destinatario@dominio.del.destinatario" El *correo-e* es un método muy rápido de comunicación y típicamente sólo toma unos pocos minutos en llegar a cualquiera, sin importar en que lugar del mundo se encuentre dicho destinatario. Para poder escribir un *correo-e*, Ud. necesita de un cliente de *correo-e* como *Pine* o `mutt` los cuales son clientes de modo texto, o clientes GUI como *KMail*.

cortafuegos

Máquina que, en la topología de una red local, es el único punto de conexión con la red externa, y que filtra o controla la actividad sobre algunos puertos, o se asegura que sólo algunas interfaces IP específicas puedan tener acceso a ellos.

cuenta

En un sistema *Unix*, un nombre de conexión, un directorio personal, una contraseña y un *shell* que le permiten a una persona conectarse a este sistema.

cuota

Es un método para restringir el uso y límite del disco para los usuarios. Los administradores pueden restringir el tamaño de los directorios personales de los usuarios configurando los límites de la cuota sobre sistemas de archivos específicos.

datagrama

Un datagrama es un paquete discreto de datos y encabezados que contienen direcciones, que es la unidad básica de transmisión a través de un red IP. También puede ser que lo haya oído nombrar como un "paquete".

dependencias

Son las etapas de la compilación que es necesario satisfacer antes de continuar con las siguientes para poder compilar un programa satisfactoriamente.

dirección física (hardware)

Es un número que identifíca unívocamente a un host en una red física en la capa de acceso al medio. Son ejemplos las **Direcciones Ethernet** y las **Direcciones AX.25**.

directorio

Parte de la estructura de un sistema de archivos. Dentro de un directorio se almacenan archivos u otros directorios. Algunas veces hay sub-directorios (o ramas) dentro de un directorio. Generalmente se denomina a esto un árbol de directorios. Si desea ver lo que hay dentro de otro directorio, o bien tendrá que listarlo o bien tendrá que cambiarse al mismo. A los archivos dentro de un directorio se los denomina hojas mientras que a los sub-directorios se los denomina ramas. Los directorios siguen las mismas restricciones que los archivos aunque los permisos significan cosas diferentes. Los directorios especiales '.' y '..' se refieren, respectivamente al directorio en sí mismo y a su directorio padre.

directorio personal

Generalmente se abrevia "home" (casa). Este es el nombre del directorio personal de un usuario dado.

Ver también: cuenta.

directorio raíz

Este es el directorio tope de un sistema de archivos. Este directorio no tiene padre, por lo tanto '..' para el directorio raíz apunta a sí mismo. El directorio raíz se escribe como '/'.

discretos, valores

Los valores discretos son aquellos que no son contínuos. Es decir, existe algún tipo de "separación" entre dos valores consecutivos.

distribución

Es un término que se usa para distinguir a un producto de un vendedor de *GNU/Linux* de otro. Una distribución está compuesta del núcleo y utilitarios de *GNU/Linux* centrales, así como también de programas de instalación, programas de terceros, y algunas veces software propietario.

DLCI

(*Data Link Connection Identifier*). Es el identificador de la conexión de datos y se usa para identificar una conexión virtual punto a punto única en una red de Relevo de Tramas (*Frame Relay*). Normalmente el proveedor de red de relevo de tramas asigna a los DLCIs.

dueño

En el contexto de los usuarios y sus archivos, el dueño de un archivo es el usuario que creó a ese archivo.

dueño, grupo

En el contexto de los grupos y sus archivos, el grupo dueño de un archivo es el grupo al cual pertenece el usuario que creó a ese archivo.

eco

Es cuando Ud. puede ver los caracteres que tecleea, por ejemplo, en el campo donde ingresa su nombre de usuario y/o contraseña. Los caracteres se muestran "tal cual" y no como un "*".

editor

Es un término usado típicamente para los programas que editan archivos de textos. También se denominan editores de texto. Los editores de *GNU/Linux* más conocidos son el editor GNU Emacs (*Emacs*) y el editor de *Unix*, *VI*.

ejecución, nivel de

Es una configuración de software del sistema que permite que existan sólo un grupo de procesos seleccionados. En el archivo `/etc/inittab` se definen cuales son los procesos ejecutados en cada uno de los niveles de ejecución. Hay ocho niveles de ejecución definidos: 0, 1, 2, 3, 4, 5, 6, S y el cambio entre niveles de ejecución lo puede realizar sólo un usuario privilegiado con los comandos `init` y `telinit`.

englobamiento

En el *shell*, la capacidad de agrupar cierto conjunto de nombres de archivo con un patrón de englobamiento.

Ver también: englobamiento, patrón de.

englobamiento, patrón de

Es una cadena de caracteres conformada por caracteres normales y especiales. El *shell* interpreta y expande los caracteres especiales.

entorno

Es el contexto de ejecución de un proceso. Esto incluye a toda la información que necesita el sistema operativo para administrar el proceso y lo que necesita el procesador para ejecutar el proceso adecuadamente.

Ver también: proceso.

entorno, variables de

Una parte del entorno del proceso. Las variables de entorno se pueden ver desde el *shell* directamente.

Ver también: proceso.

escapar

En el contexto del *shell*, es la acción de poner alguna cadena de caracteres entre comillas para evitar que el *shell* la interprete. Por ejemplo, cuando Ud. quiere, o debe, usar espacios en alguna línea de comandos y enviar el resultado a otro comando por una tubería, tiene que poner al primer comando entre comillas ("escapar" el comando) de no ser así, el *shell* no lo interpretará bien y no funcionará como se espera.

escritorio

Si está utilizando *X*, el escritorio es el lugar de la pantalla dentro del cual Ud. trabaja y sobre el cual se muestran los iconos y las ventanas. También se denomina "fondo", y por lo general se llena con un color simple, un color en degradé o incluso una imagen.

Ver también: escritorios virtuales.

escritorios virtuales

En *X*, el *ventanas, administrador de* puede proporcionarle varios *escritorio*s. Esta característica útil le permite organizar sus ventanas, evitando el problema de tener docenas de ellas apiladas una encima de otra. Esto funciona como si Ud. tuviera muchas pantallas. Se puede pasar de un escritorio virtual a otro en una manera que depende del administrador de ventanas que Ud. está utilizando.

expresión regular

Potente herramienta teórica que se usa para buscar y hacer corresponder cadenas de texto. Le permite especificar patrones que deben obedecer dichas cadenas. Muchos utilitarios *Unix* la usan: `sed`, *awk*, *grep* y *Perl* entre otros.

flag

Es un indicador (usualmente un bit) que se usa para señalar alguna condición a un programa. Por ejemplo, un sistema de archivos tiene, entre otros, a un *flag* para indicar si tiene que ser volcado en una copia de respaldo, de forma tal que cuando este está activo se hace una copia de respaldo del sistema de archivos, y cuando no lo está no.

foco

Para una ventana, acción de recibir eventos de teclado (tales como pulsado y soltado de teclas) y clics del ratón, a menos que sean "atrapados" por el administrador de ventanas.

framebuffer

Proyección de la memoria RAM de una placa de vídeo en la memoria principal. Esto permite que las aplicaciones accedan a la RAM de vídeo sin necesidad de comunicarse con la placa. Por ejemplo, todas las estaciones de trabajo gráficas de alto nivel usan *framebuffers*.

gateway

Vínculo que conecta dos redes IP. También denominado pasarela.

GFDL

La *GNU Free Documentation License* (Licencia de Documentación Libre GNU). Es la licencia que se aplica a toda la documentación de la distribución **Linux-Mandrake**.

host

Se refiere a una computadora y normalmente se usa cuando se habla de computadoras conectadas sobre una red.

icono

Es un dibujo pequeño (normalmente de 16×16, 32×32, 48×48, y a veces 64×64 pixels de tamaño) que representa, bajo un entorno gráfico, a un documento o a un programa.

i-nodo

Punto de entrada que conduce al contenido de un archivo en un sistema de archivos de tipo *Unix*. Un i-nodo está identificado de manera única con un número, y contiene meta-información acerca del archivo al cual se refiere, tal como sus tiempos de acceso, su tipo, su tamaño, ¡pero no su nombre!

inserción, modo de

Bajo *VI* o uno de sus clones, es el estado del programa en el cual al presionar una tecla, esta se insertará en el archivo que se está editando (excepto casos patológicos como el completado y la abreviación, justificación a la derecha al final de la línea, ...). Uno sale del modo de inserción al presionar **Esc** (o **Ctrl-[**).

internet

Es una red enorme que conecta a las computadoras alrededor del mundo.

ISO 8859

La norma ISO 8859 incluye varias extensiones de 8 bits al conjunto de caracteres ASCII (ver *ASCII*). La ISO 8859-1, el “Alfabeto Latino No. 1”, es especialmente importante. El mismo se ha vuelto ampliamente implementado y ya se puede ver como el reemplazo de-facto estándar de ASCII.

ISO 8859-1 (Figura 2) soporta los idiomas siguientes: Afrikaans, Alemán, Catalán, Danés, Escocés, Español, Faeroés, Finlandés, Francés, Gallego, Holandés, Inglés, Islandés, Irlandés, Italiano, Noruego, Portugés, Sueco, y Vasco.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8																
9																
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	–	®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figura 2. Tabla ISO-8859-1

Note que los caracteres ISO 8859-1 también son los primeros 256 caracteres de ISO 10646 (Unicode). Sin embargo, le falta el símbolo del EURO y no cubre al Finlandés y al Francés por completo. ISO 8859-15 (Figura 3) es una modificación de ISO 8859-1 que cubre estas necesidades.

El conjunto completo de alfabetos ISO 8859 incluye:

Nombre	**Idioma(s)**
ISO 8859-1	idiomas de Europa Occidental (Latin-1)
ISO 8859-2	idiomas de Europa Oriental (Latin-2)
ISO 8859-3	sudeste de Europa y misceláneos (Latin-3)
ISO 8859-4	idiomas Escandinavos/Bálticos (Latin-4)
ISO 8859-5	Latino/Cirílico
ISO 8859-6	Latino/Arábigo
ISO 8859-7	Latino/Griego
ISO 8859-8	Latino/Hebreo
ISO 8859-9	modificación de Latin-1 para Turco (Latin-5)
ISO 8859-10	idiomas Laponés/Nórdico/Esquimal (Latin-6)
ISO 8859-11	Tailandés
ISO 8859-13	idiomas de la cuenca Báltica (Latin-7)
ISO 8859-14	Celta (Latin-8)
ISO 8859-15	idiomas de Europa Occidental con el símbolo del Euro (Latin-9)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8																
9																
A		¡	¢	£	€	¥	Š	§	š	©	ª	«	¬	–	®	¯
B	°	±	²	³	Ž	µ	¶	·	ž	¹	º	»	Œ	œ	Ÿ	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Figura 3. Tabla ISO-8859-15

job

En el contexto del *shell*, un job es un proceso que está corriendo en segundo plano. Ud. puede tener varios jobs en un mismo *shell* y controlarlos.

Ver también: primer plano, segundo plano.

kernel

Ver **núcleo**

kill ring

Bajo *Emacs*, es el conjunto de zonas de texto cortadas o copiadas desde que se inició el editor, que pueden ser llamadas para volver a insertarlas, y que está organizado como un anillo.

lanzar

Es la acción de invocar, o iniciar, un programa.

lenguaje ensamblador

Es un lenguaje de programación que está más cerca de la computadora, por lo tanto se denomina un lenguaje de programación de "bajo nivel". El lenguaje ensamblador tiene la ventaja de la velocidad debido a que estos programas se escriben en términos de instrucciones de procesador por lo que se necesita poca o ninguna traducción cuando se generan los ejecutables. Su principal desventaja es que depende del procesador (o arquitectura). También la escritura de programas complejos es una tarea ardua. Entonces, el lenguaje ensamblador es el lenguaje de programación más rápido, pero no es portable entre las distintas arquitecturas.

linkage (vincular código objeto)

Última etapa del proceso de compilación, que consiste en vincular juntos a todos los archivos objetos para producir un archivo ejecutable, y hacer coincidir los símbolos que no se pudieron resolver con las bibliotecas dinámicas (a menos que se haya pedido una vinculación estática, en cuyo caso el código de estos símbolos se incluirá en el ejecutable).

Linux

Es un sistema operativo tipo *Unix* que corre en una variedad de computadoras diferentes, y cualquiera es libre de usarlo y modificarlo. Linus Torvalds escribió a Linux (el núcleo).

login

Nombre de conexión para un usuario en un sistema *Unix*. También se denomina así al hecho de conectarse.

loopback

Interfaz de red virtual de una máquina consigo misma, que permite que los programas en ejecución no tengan en cuenta el caso especial donde dos entidades de red son, de hecho, la misma máquina.

mayor

Número específico a la clase de dispositivo.

menor

Número que define con precisión al dispositivo del cual estamos hablando.

menú desplegable

Es un menú que está "enrollado" con un botón en alguna de sus esquinas. Cuando Ud. presiona sobre dicho botón se " desenrolla", o despliega, el menú completo.

modo bloque, archivos de

Archivos cuyo contenido se almacena en una memoria temporal. Todas las operaciones para tales archivos pasan por estas zonas de memoria, lo que permite la escritura asincrónica sobre el hardware, y para las lecturas, no volver a leer lo que ya está almacenado en memoria.

Ver también: buffer, buffer cache, modo caracter, archivos de.

modo caracter, archivos de

Archivos cuyo contenido no se almacena en una memoria temporal (buffer). Toda la entrada/salida se realiza físicamente en el momento. Estos archivos corresponden a los flujos de datos.

modo de lectura-escritura

Para un archivo significa que se puede escribir en el mismo. Se puede leer su contenido y también modificarlo.

Ver también: modo de solo lectura.

modo de solo lectura

Para un archivo significa que no se puede escribir en el mismo. Se puede leer su contenido pero no se puede modificar.

Ver también: modo de lectura-escritura.

monousuario

Se usa para describir al estado de un sistema operativo, o incluso a un sistema operativo en sí mismo, que sólo permite conectarse y usar el sistema a un único usuario a la vez.

montaje, punto de

Es el directorio donde se una partición u otro dispositivo se anexa al sistema de archivos de *GNU/Linux*. Por ejemplo, su CD-ROM está montado en el directorio `/mnt/cdrom`, desde donde Ud. puede explorar el contenido de cualquier CD montado.

MSS

(*Maximum Segment Size*) Tamaño máximo de segmento es la mayor cantidad de datos que se pueden transmitir a la vez. Si quiere evitar la fragmentación local, el MSS debería ser igual al encabezado MTU de IP.

MTU

(*Maximum Transmission Unit*) Es un parámetro que determina el tamaño mayor de datagrama que se puede transmitir por una interfaz IP sin necesidad de descomponerlo en unidades más pequeñas. El MTU debería ser mayor que el datagrama de mayor tamaño que Ud. desee transmitir sin fragmentación. Note que esto sólo evita la fragmentación local, en la ruta puede haber otro vínculo que tenga un MTU menor y el datagrama se fragmentará allí. Los valores típicos son 1500 bytes para una interfaz *Ethernet*, o 576 bytes para una interfaz SLIP.

multitarea (`multitasking`)

Es cuando un sistema operativo puede correr más de un programa a la vez. Hay dos tipos de multitarea: la multitarea por prioridad es cuando el sistema operativo es el responsable de distribuir el tiempo de CPU entre los procesos, mientras que multitarea cooperativa es cuando los procesos son los que devuelven el tiempo de CPU. La primera variante es, obviamente, la mejor opción debido a que ningún programa puede monopolizar el tiempo de CPU bloqueando así a los otros procesos. *GNU/Linux* es un sistema operativo que usa multitarea por prioridad real.

multiusuario

Se usa para describir a un sistema operativo que permite que múltiples usuarios se conecten y usen al sistema exactamente a la vez, pudiendo cada uno hacer sus propias tareas independientemente de los demás usuarios. Es necesario que un sistema operativo multitarea proporcione soporte para el modo multiusuario. *GNU/Linux* es un sistema operativo multiusuario y también multitarea.

nombrado

Una palabra usada comúnmente en computación para un método que identifica objetos. Ud. escuchará seguido acerca de “convenciones de nombrado” para los archivos, funciones, en un programa y así sucesivamente.

noticias, grupos de (`newsgroups`)

Foros de discusión y áreas de noticias a las que se puede acceder usando un cliente de noticias o USENET para leer y escribir mensajes específicos al tema de dichos foros. Por ejemplo, el grupo de noticias `alt.os.linux.mandrake` es un grupo de noticias alternativo (alt) que trata con los sistemas operativos (os) *GNU/Linux* (linux), y específicamente con **Linux-Mandrake** (mandrake). Los grupos de noticias se dividen de esta manera para facilitar la búsqueda de un tema en particular.

nulo, caracter

El caracter o byte número 0, se usa para marcar el final de una cadena de caracteres o *string*. Su nombre técnico es `NULL`.

objetivo

Es el objeto de la compilación, es decir el archivo binario que generará el compilador.

on the fly ("al vuelo")

Se dice que algo se hace "al vuelo" cuando se realiza junto con alguna otra cosa, sin que Ud. lo note o lo haya pedido explícitamente.

open source (código abierto)

Es el nombre que se le da al código fuente de un programa libre que se pone a disposición del público y de la comunidad en general para su desarrollo. La teoría detrás de esta filosofía es que el hecho de permitir que el código fuente sea usado y modificado por un grupo de programadores más amplio, a la larga producirá un producto más útil para todos. Entre algunos programas populares de código abierto se encuentran *Apache*, *sendmail* y *GNU/Linux*.

página `man`

Es un documento que contiene la definición y el uso de un comando. Este documento se consulta con el comando `man`. La primer cosa que uno debería (aprender a) leer cuando se entera de un comando que no conoce **:-)**

paginador

Programa que muestra un archivo de texto una pantalla a la vez, y que facilita el desplazamiento y la búsqueda de cadenas en dicho archivo. Le aconsejamos usar `less` como paginador.

pantalla completa

Este término se usa para referirse a las aplicaciones que ocupan todo el área visible de su pantalla.

patch, "patchear"

Archivo que contiene una lista de correcciones a hacer sobre un código fuente para agregar características nuevas, eliminar errores, o modificarlo de acuerdo a los deseos y necesidades de uno. La acción consistente en aplicar estas correcciones al archivado de código fuente. También conocido como "parche".

path (ruta)

Es una asignación para los archivos y los directorios al sistema de archivos. Las diferentes capas de la ruta están separadas por la "barra" o "/". Hay dos tipos de rutas en los sistemas *GNU/Linux*. La ruta **relativa** es la posición de un archivo o directorio en relación al directorio corriente. La ruta **absoluta** es la posición de un archivo o directorio en relación al directorio raíz.

pixmap

Es un acrónimo para *pixel map* (Mapa de pixeles). Es otra forma de referirse a una imagen de mapa de bits.

plugin

Programa “adicionable” que se usa para mostrar o reproducir algunos contenidos multimedios que se encuentran en un documento *web*. Por lo general, se puede transferir desde *Internet* fácilmente si su navegador todavía no puede mostrar o reproducir esa clase de información.

por lotes

Es un modo de procesamiento en el cual se envían trabajos al procesador, y luego el procesador los ejecuta uno tras otro hasta que ejecuta el último y queda disponible para recibir otra lote de procesos.

portar

Portar un programa es traducir dicho programa de forma tal que se pueda usar en un sistema para el cual, originalmente, no se tenía intención de usar, o que se pueda usar en sistemas “similares”. Por ejemplo, para poder correr un programa de *Windows* nativo bajo *GNU/Linux* (en modo nativo), primero se debe portar dicho programa a *GNU/Linux*.

precedencia

Dicta el orden de evaluación de los operandos en una expresión. Por ejemplo: Si Ud. tiene `4 + 3 * 2` el resultado que obtiene es 14, ya que la suma tiene mayor precedencia que el producto. Si Ud. quiere evaluar primero el producto, tiene que agregar paréntesis para obtener algo así `4 + (3 * 2)`, y entonces obtiene 10 como resultado debido a que los paréntesis tienen mayor precedencia que la suma y el producto y por lo tanto se los evalúa primero.

preprocesadores

Son directivas de compilación que instruyen al compilador para que reemplace dichas directivas por código en el lenguaje de programación usado en el archivo fuente. Son ejemplos de preprocesadores del lenguaje *C*: `#include`, `#define`, etc.

primer plano

En el contexto del *shell*, el proceso que está en primer plano es aquel que está corriendo actualmente. Ud. tiene que esperar que tal proceso termine para poder volver a ingresar comandos.

Ver también: job, segundo plano.

proceso

En un contexto *Unix*, un proceso es una instancia de un programa en ejecución junto con su entorno.

prompt

En un *shell*, es la cadena que aparece antes del cursor. Cuando lo vea, Ud. puede ingresar sus comandos.

protocolo

Los protocolos organizan la comunicación entre máquinas diferentes a través de una red, ya sea usando hardware o software o ambos. Estos definen el formato de los datos transferidos, si una máquina controla a otra, etc. Algunos protocolos bien conocidos incluyen a HTTP, FTP, TCP, y UDP.

proxy

Una máquina que se coloca entre su red local e *Internet*, cuyo rol es acelerar la transferencia de datos para los protocolos usados más ampliamente (por ejemplo, HTTP y FTP). Mantiene un cache de los pedidos anteriores, lo que evita el costo de tener que volver a pedir un archivo cuando alguna máquina pida lo mismo. Son muy útiles para redes de ancho de banda reducido (entiéndase: conexiones por módem). A veces, también es la única máquina que puede acceder al exterior de la red.

RDSI

Red Digital de Servicios Integrados. Conjunto de normas de comunicaciones para permitir que un solo cable o una fibra óptica transporte voz, servicios de red digital y vídeo. Ha sido diseñado para reemplazar eventualmente a los sistemas de teléfono actuales. Técnicamente es una red de datos de conmutación de circuitos.

recorrer

Para un directorio en un sistema *Unix*, esto significa que el usuario tiene permitido atravesar este directorio, y posiblemente los directorios debajo de este. Para esto, es necesario que el usuario tenga derecho de ejecución sobre este directorio.

Relevo de Tramas

(*Frame Relay*) Es una tecnología de redes idealmente adecuada para transportar tráfico que se presenta en ráfagas o es de naturaleza esporádica. Los costos de red se reducen teniendo a varios clientes de Relevo de Tramas compartiendo la misma capacidad de red y confiando que los mismos deseen hacer uso de la red en momentos ligeramente distintos.

root

Es el super-usuario de cualquier sistema *Unix*. Típicamente root (conocido también como administrador) es la persona responsable de mantener y supervisar al sistema *Unix*. Esta persona también tiene acceso completo a cualquier cosa en el sistema.

ruta

Es el camino que toman los datagramas a través de la red para llegar a su destino. Camino entre una máquina y otra en una red.

script

Los scripts del *shell* son secuencias de comandos a ejecutar como si hubiesen sido ingresadas en la consola una tras otra. Los scripts del *shell* son el equivalente *Unix* (aproximado) de los archivos por lotes (batch) de *DOS*.

segundo plano

En el contexto del *shell*, un proceso está corriendo en segundo plano si Ud. puede ingresar comandos en la consola mientras el mismo está corriendo. Ver también `job`, **primer plano**.

Ver también: job, primer plano.

seguridad, niveles de

Característica única de **Linux-Mandrake** que le permite configurar niveles de restricciones diferentes de acuerdo a cuán seguro quiera hacer su sistema. Hay 6 niveles predefinidos desde 0 hasta 5, donde 5 es el nivel más restrictivo. Ud. también puede definir su nivel de seguridad propio.

servidor

Programa o computadora que propone una característica o presta un servicio y espera las conexiones de los **clientes** para ejecutar las órdenes de estos o darles la información que estos pidan. Ejemplos típicos son los servidores FTP, HTTP, NFS, servidores de *correo-e*, etc. En el caso de sistemas **de igual a igual** (*peer to peer*) tales como PPP o SLIP el servidor se toma como el extremo de la conexión que recibe la llamada y el otro extermo se toma como cliente. Es uno de los componentes de un **sistema cliente/servidor**.

shadow passwords

Un conjunto de administración de contraseñas en los sistemas *Unix* en el cual el archivo que contiene las contraseñas cifradas ya no es legible por todo el mundo, como lo es cuando se usa el sistema normal de contraseñas.

shell

El *shell* es la interfaz básica al núcleo del sistema operativo y es quien proporciona la línea de comandos donde el usuario ingresa comandos para ejecutar programas y comandos del sistema. La mayoría de los shells proporcionan un lenguaje de script que se puede utilizar para automatizar tareas o simplificar tareas complejas usadas con frecuencia. Estos scripts del *shell* son similares a los archivos batch del sistema operativo *DOS*, pero son mucho más potentes. Algunos ejemplos de shells son *Bash*, *sh*, y *tcsh*.

sistema de archivos raíz

Este es el sistema de archivos que está en el nivel superior. En este sistema de archivos *GNU/Linux* monta la raíz de su árbol de directorios. Este sistema de archivos debe residir en una partición propia, ya que es la base para todo el sistema. El mismo contiene al directorio raíz.

sistema operativo

Es un proceso que corre permanentemente en segundo plano que permite la operación básica de la computadora. La tarea primaria para cualquier sistema operativo es la administración de todos los recursos específicos de la máquina. En un sistema *GNU/Linux*, es el núcleo y los módulos cargables los que llevan a cabo estas tareas. Algunos sistemas operativos bien conocidos incluyen a *GNU/Linux*, *AmigaOS*, *MacOS*, *FreeBSD*, *OS/2*, *Unix*, *Windows NT*, y *Windows 9x*.

socket

Tipo de archivo correspondiente a cualquier conexión de red.

standard error

Error estándar. Es el descriptor de archivo número 2, abierto por cada proceso, usado por convención para imprimir mensajes de error. Predeterminadamente es la pantalla de la terminal.

Ver también: standard input, standard output.

standard input

Entrada estándar. Es el descriptor de archivo número 0, abierto por cada proceso, usado por convención como el descriptor desde el cual el proceso recibe los datos. Predeterminadamente, es el teclado.

Ver también: standard error, standard output.

standard output

Salida estándar. Es el descriptor de archivo número 1, abierto por cada proceso, usado por convención como el descriptor en el cual el proceso imprime su salida. Predeterminadamente, es la pantalla de la terminal.

Ver también: standard error, standard input.

streamer

Es un dispositivo que toma *streams* (flujos) de caracteres como su entrada. Un *streamer* típico es una unidad de cinta.

telnet

Crea una conexión a un host remoto y le permite conectarse a la máquina siempre y cuando Ud. posea una cuenta. Telnet es el método de conexión remota más utilizado, sin embargo hay alternativas mejores y más seguras como `ssh`.

temas, soporte de

Una aplicación gráfica soporta temas si se puede cambiar su apariencia en tiempo real. También muchos administradores de ventanas soportan temas.

tubería

Un tipo especial de archivo *Unix*. Un programa escribe datos en la tubería, y otro programa lee los datos del otro lado de la tubería. Las tuberías *Unix* son FIFO, por lo que los datos se leen en el mismo orden en el que fueron enviados. De uso amplio con el *shell*. Ver también **tubería nombrada**.

Ver también: tubería nombrada.

tubería nombrada

Una tubería *Unix* que está vinculada, al contrario de las tuberías usadas en el *shell*. Ver también **tubería**, **vínculo**.

Ver también: tubería.

usuario, nombre de

Es un nombre (o más genéricamente, una palabra) que identifica a un usuario en un sistema. Cada usuario se asocia a un único UID (Identificador de usuario)

Ver también: login.

variables

Son cadenas que se usan en los archivos `Makefile` para reemplazarlas por su valor cada vez que aparecen. Usualmente su valor se declara al comienzo del archivo `Makefile`. Se usan para simplificar la administración de los `Makefile`s y el árbol de archivos de código fuente.

Más ampliamente, en la programación las variables son palabras que se refieren a otras entidades (números, cadenas de caracteres, tablas, etc.) que tienden a variar mientras el programa se está ejecutando.

ventana

En el contexto de las redes, la **ventana** es la mayor cantidad de datos que el extremo receptor puede aceptar en un punto dado en el tiempo.

ventanas, administrador de

Es el programa responsable del “look and feel” de un entorno gráfico que trata con los distintos elementos de una ventana como por ejemplo: las barras, los marcos, los botones, los menús, y algunos atajos de teclado. Sin ellos sería muy difícil o imposible tener escritorios virtuales, cambiar el tamaño de las ventanas al vuelo, moverlas, etc.

verboso

Para los comandos, el modo verboso significa que el comando reporta en la salida estándar todas las acciones que lleva a cabo y los resultados de dichas acciones. A veces, los comandos tienen una forma de definir el “nivel de verbosidad”, lo cual significa que se puede controlar la cantidad de información que reportará el comando.

vínculo

Referencia a un i-nodo en un directorio, por lo tanto le da un nombre (de archivo) al i-nodo.

vínculos de software

Ver “vínculos simbólicos”.

vínculos simbólicos

Archivos especiales, que sólo contienen una cadena de caracteres, y donde cualquier acceso a ellos es equivalente a un acceso al archivo cuyo nombre es dicha cadena, el cual puede existir o no, y la ruta de la misma se puede dar de forma relativa o absoluta.