Linux

Conceptos Básicos, Administración y Servicios

Alberto Jaspe Villanueva
ajaspe@gmail.com
Octubre 2002

Índice general

Capítulo 1

Conceptos Básicos

1.1. Introducción al Sistema Operativo Linux

1.1.1. Un poco de historia

En 1991, Linus Benedict Torvalds, estudiante de la Universidad Helsinki, estrenó la primera versión pública de su sistema operativo Linux la 0.02. Desde entonces, millones de usuarios de todo el mundo poseen este sistema gratuito y miles de ellos contribuyen a su continuo desarrollo aportando ideas, programas, información sobre fallos del sistema ya sea en hardware/software (bugs), ayuda, tutoriales, etc.

Linux nació de la idea de crear un sistema clon de Unix basado en GNU (General Public License, Licencia General Pública) y el código fuente disponible gratuitamente. Se ha replicado una traducción al español de esta licencia en el apéndice C por el importante papel que juega en el SO Linux.

Esta idea nació en 1991 cuando Linus Torvalds estudiaba la carrera de Ciencias Informáticas. Torvalds se encontraba especialmente interesado en Minix, el único sistema Unix disponible en aquél entonces de fácil acceso para los estudiantes y profesores. Este sistema gratuito fue creado por Andrew Tanenbaum con el propósito de facilitar a los alumnos de la universidad el estudio y diseño de sistemas operativos. Minix era un Unix más, tanto en apariencia como en el kernel (núcleo del sistema operativo), pero distaba mucho de ser comparable a uno de los grandes. Es a partir de aquel momento que Torvalds decidió crear un sistema que excediera los éstandares de Minix, poniendo en marcha el proyecto personal Linux.

Torvalds tomó sus primeras clases de C y Unix en 1990 y en poco tiempo empezó a utilizar el sistema operativo Minix en su nuevo 386. Linux evolucionó desde el simple programa "Hola, Mundo" a una terminal. Durante mucho tiempo Torvalds trabajó en la soledad de sus ideas, hasta la mañana del 3 de julio de 1991 cuando pidió ayuda a través del Internet. Al principio fueron unos pocos los que le apoyaron, pero al poco tiempo muchos otros cibernautas se unieron al proyecto. En uno de los primeros emails enviados por Torvalds a la comunidad

del ciberespacio respecto a Linux, informaba sobre su proyecto como si fuera un hobby, nada tan grande ni comparable con GNU.

Durante el desarrollo Torvalds se encontró con muchos problemas a lo largo de la programación del kernel. Pero Linux empezó a disponer de controladores para los dispositivos internos del PC y un funcionamiento correcto del disco aproximadamente el 3 de julio, unas horas después de enviar su primer email informado sobre su proyecto. Dos meses más tarde Linux empezaba a funcionar y el código fuente de la primera versión 0.01 ya estaba disponible. La versión 0.01 incluía un bash shell 1.08 y el compilador gcc 1.40.

Muy pronto Linux se convirtió en un sistema mucho más fácil de instalar y configurar, y empezó a coger fama en todo el mundo. Al tener en muy poco tiempo miles de usuarios, las nuevas versiones de Linux salían casi semanalmente. En el presente hay millones de usuarios y gracias a ellos y a sus aportes, Linux crece sin respiro alguno. La última versión del kernel estable es Linux 2.4.19 de Septiembre de 2002.

Como todos los sistemas operativos, Linux también dispone de un logotipo. Torvalds decidió que la imagen que representaría a Linux sería la de un pingüino. En casi todas la páginas web relacionadas con Linux se puede hallar el logotipo.

Linux había nacido para ser un sistema operativo del tipo POSIX (sistema variante de UNIX), totalmente gratuito para el usuario y con libre acceso al código fuente. Estas tres ideas fueron las que lo han convertido en el sistema con mejor rendimiento, más fiable, veloz y con más desarrolladores del mundo. Pronto se ha colocado cerca de los grandes sistemas operativos como UNIX en el ámbito de servidores de comunicaciones, especialmente utilizado en empresas proveedoras de acceso a Internet.

Las versiones más recientes de Linux ofrecen la posibilidad de convertir nuestro ordenador personal en una potente estación de trabajo. Puede funcionar como estación de trabajo personal dándonos la posibilidad de acceder a las prestaciones que ofrece UNIX y cualquier otro sistema operativo. Además, gracias al aporte de muchas empresas hoy en día cuenta con potentes entornos gráficos que ayudan significativamente a elegir Linux. Puede adem´ás configurar para funcionar como estación de desarrollo y/o aprendizaje, proveer acceso a Intranets e Internet y muchas otras opciones.

Linux como estación de desarrollo y/o aprendizaje es uno de los mejores sistemas ya que dispone de muchos lenguajes de programación gratuitos como: GNU C, GNU C++, GNU Fortran 77, ADA, Pascal, TCL/Tk, etc. y ahora también las versiones conocidas de Delphi para Linux de Borland Inc. (Kylix) las cuales esperemos que también sean de fácil acceso por los usuarios o en todo caso a un costo razonable que permita contar con esta valiosa herramienta de programación. La mayoría de estos lenguajes vienen con extensas librerías de código fuente.

Linux como sistema operativo gratuito posee características que le hacen único. Las más importantes son: multitarea, memoria virtual, los drivers (controladores de dispositivos) TCP/IP más rápidos del mundo, librerías compartidas, multiusuario, modo de funcionamiento protegido (al contrario de otros Sistema Operativos) y la más fundamental soporta multitarea de 32 y 64 bits.

Posee además capacidades avanzadas para la interconexión de redes de PC´s ya que para desarrollar Linux hubo que utilizar Internet. El desarrollo del software y las características de interconexión de redes se empezó a desarrollar desde las primeras versiones de Linux y desde entonces ha ido evolucionando a gran velocidad y más aún con la gran aceptación de la red; en especial de Internet.

Hoy en día Linux es utilizado por millones de usuarios y miles de empresas. No hay duda pues que Linux es uno de los sistemas operativos con mas posibilidades y es el único que se actualiza día a día.

1.1.2. Características de Linux

- Multitarea: La palabra multitarea describe la habilidad de ejecutar varios programas al mismo tiempo. LINUX utiliza la llamada multitarea preventiva, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa.
- Multiusuario: Muchos usuarios usando la misma máquina al mismo tiempo.
- Multiplataforma: Las plataformas en las que en un principio se puede utilizar Linux son 386, 486. Pentium, Pentium Pro, Pentium II, AMD Kx, Amiga y Atari, también existen versiones para su utilización en otras plataformas, como Alpha, ARM, MIPS, PowerPC y SPARC.
- Multiprocesador: Soporte para sistemas con mas de un procesador esta disponible para Intel y SPARC (hasta 16 CPU's).
- Funciona en modo protegido 386.
- Protección de la memoria entre procesos, de manera que uno de ellos no pueda colgar el sistema.
- Carga de ejecutables por demanda: Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.
- Política de copia en escritura para la compartición de páginas entre ejecutables: esto significa que varios procesos pueden usar la misma zona de memoria para ejecutarse. Cuando alguno intenta escribir en esa memoria, la página (4Kb de memoria) se copia a otro lugar. Esta política de copia en escritura tiene dos beneficios: aumenta la velocidad y reduce el uso de memoria.

- Memoria virtual usando paginación (sin intercambio de procesos completos) a disco: A una partición o un archivo en el sistema de archivos, o ambos, con la posibilidad de añadir más áreas de intercambio sobre la marcha. Un total de 16 zonas de intercambio de 128Mb de tamaño máximo pueden ser usadas en un momento dado con un límite teórico de 2Gb para intercambio. Este limite se puede aumentar fácilmente con el cambio de unas cuantas lineas en el código fuente.

- La memoria se gestiona como un recurso unificado para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.

- Librerías compartidas de carga dinámica (DLL's) y librerías estáticas.

- Se realizan volcados de estado (core dumps) para posibilitar los análisis post-mortem, permitiendo el uso de depuradores sobre los programas no sólo en ejecución sino también tras abortar éstos por cualquier motivo.

- Compatible con POSIX, System V y BSD a nivel fuente.

- Emulación de iBCS2, casi completamente compatible con SCO, SVR3 y SVR4 a nivel binario.

- Todo el código fuente está disponible, incluyendo el núcleo completo y todos los drivers, las herramientas de desarrollo y todos los programas de usuario; además todo ello se puede distribuir libremente. Hay algunos programas comerciales que están siendo ofrecidos para Linux actualmente sin código fuente, pero todo lo que ha sido gratuito sigue siendo gratuito.

- Control de tareas POSIX.

- Pseudo-terminales (pty's).

- Emulación de 387 en el núcleo, de tal forma que los programas no tengan que hacer su propia emulación matemática. Cualquier máquina que ejecute Linux parecerá dotada de coprocesador matemático. Por supuesto, si el ordenador ya tiene una FPU (unidad de coma flotante), esta será usada en lugar de la emulación, pudiendo incluso compilar tu propio kernel sin la emulación matemática y conseguir un pequeño ahorro de memoria.

- Soporte multilenguaje de páginas de código siendo bastante fácil añadir nuevos dinámicamente.

- Consolas virtuales múltiples: varias sesiones de login a través de la consola entre las que se puede cambiar con las combinaciones adecuadas de teclas (totalmente independiente del hardware de vídeo). Se crean dinámicamente y puedes tener hasta 256 en la nueva serie del kernel.

- Soporte para varios sistemas de archivo comunes, incluyendo minix-1, Xenix y todos los sistemas de archivo típicos de System V, y tiene un avanzado sistema de archivos propio con una capacidad de hasta 4 Tb y nombres de archivos de hasta 255 caracteres de longitud (Ext2).
- Acceso transparente a particiones MS-DOS (o a particiones OS/2 FAT) mediante un sistema de archivos especial: no es necesario ningún comando especial para usar la partición MS-DOS, esta parece un sistema de archivos normal de Unix (excepto por algunas restricciones en los nombres de archivo, permisos, y esas cosas). Las particiones comprimidas de MS-DOS 6 no son accesibles en este momento, y no se espera que lo sean en el futuro, pero si lo son mediante el DOSEMU.
- Soporte para VFAT (WNT, Windows 95y Windows 98), y también para FAT32.
- Un sistema de archivos especial llamado UMSDOS que permite que Linux sea instalado en un sistema de archivos DOS.
- Soporte en sólo lectura de HPFS-2 del OS/2 2.1
- Sistema de archivos de CD-ROM que lee todos los formatos estándar de CD-ROM, incluyendo Microsoft Joliet Nivel 3.
- TCP/IP, incluyendo ftp, telnet, NFS, etc.
- Appletalk.
- Software cliente y servidor Netware.
- Lan Manager / Windows Native (SMB), software cliente y servidor.
- Diversos protocolos de red incluidos en el kernel: TCP, IPv4, IPv6, AX.25, X.25, IPX, DDP, Netrom, etc. AX.25 es el protocolo de red de las transmisiones de Radio Paquete, habitualmente conocido como Packet.
- Diseño modular del kernel, abandonando el uso de un kernel monolítico.

1.1.3. Distribuciones

Una distribución no es otra cosa, que una recopilación de programas y ficheros, organizados y preparados para su instalación. Estas distribuciones se pueden obtener a través de Internet, o comprando los CDs de las mismas, los cuales contendrán todo lo necesario para instalar un sistema Linux bastante completo y en la mayoría de los casos un programa de instalación que nos ayudara en la tarea de una primera instalación. Casi todos los principales distribuidores de Linux, ofrecen la posibilidad de bajarse sus distribuciones, vía FTP (sin cargo alguno).

Existen varias distribuciones creadas por diferentes empresas a unos precios bastantes asequibles (si se compran los CDs, en vez de bajársela vía FTP), las cuales se deberían poder encontrar en tiendas de informática o librerías. En el peor de los casos siempre se puedes encargarlas directamente por Internet a las empresas que las crean. A veces, las revistas de informática sacan una edición bastante aceptable de alguna distribución.

Algunas de las distribuciones más importantes son:

- Redhat (www.redhat.com)

  Esta es una distribución que tiene muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que la distribuye. Fácil de instalar.

- Debian (www.debain.org)

  Distribución con muy buena calidad. El proceso de instalación es quizás un poco mas complicado que la anterior, pero sin mayores problemas. Gran estabilidad antes que últimos avances.

- Suse (www.suse.com)

  Otra de las grandes. Tiene un entorno muy amigable y muy sencillo, idóneo para usuarios noveles, pero esto le hace perder robustez.

- Caldera (www.calderasystems.com)

  Mayormente orientada al mundo empresarial.

- Slackware (www.slackware.com)

  Esta distribución es de las primeras que existió. Tuvo un periodo en el cual no se actualizaba muy a menudo, pero eso es historia. Es raro encontrar usuarios de los que empezaron en el mundo linux hace tiempo, que no hayan tenido esta distribución instalada en su ordenador en algún momento.

- Mandrake (www.linux-mandrake.com/es)

  Distribución basada en Redhat. Esta distribución viene con KDE totalmente integrado en el sistema. Fácil de instalar y configurar.

- Esware (www.esware.com)

  Esware es una distribución, basada en Redhat, creada por una empresa española. Es una distribución pensada especialmente para los hispano-parlantes. Tiene traducida al castellano, la instalación, todos los mensajes de consola, KDE en castellano así como numerosos programas y la documentación.

1.2. Nociones de UNIX básico

UNIX es un sistema operativo multitarea y multiusuario. Esto significa que puede haber mas de una persona usando un ordenador a la vez, cada uno de ellos ejecutando a su vez diferentes aplicaciones.

(Esto difiere de MS-DOS, donde solo una persona puede usar el sistema en un momento dado). Bajo UNIX, para que los usuarios puedan identificarse en el sistema, deben presentarse (log in), proceso que consta de dos pasos: Introducir el nombre de usuario (login) (el nombre con que será identificado por el sistema), y una palabra de paso (password), la cual es su llave personal secreta para entrar en la cuenta. Como solo usted conoce su palabra de paso, nadie mas podrá presentarse en el sistema con su nombre de usuario.

En los sistemas UNIX tradicionales, el administrador del sistema asignara el nombre de usuario y una palabra de paso inicial en el momento de crear la cuenta de usuario. Para el resto de los ejemplos del libro, usaremos el nombre de usuario "pepe".

Además, cada sistema UNIX tiene un nombre del sistema (hostname) asignado. Este "hostname" le da nombre a la máquina. El nombre del sistema es usado para identificar máquinas en una red, pero incluso aunque la máquina no este en red, debería tener su nombre. Más adelante veremos como inicializar el nombre de la máquina. En nuestros ejemplos, el nombre del sistema será "nino".

Linux es un operativo pensado tanto para funcionar en el hogar como en el trabajo. En casa, el usuario y el administrador suelen ser la misma persona (que deberá conocer comandos para operar como ambas entidades cuando sea necesario, a menos que el PC sea usado por varias personas de la familia, y uno de ellos sea el "administrador"). En esta sección se introducirán los términos básicos que tanto los usuarios como los administradores deben conocer para el uso "ordinario" del PC con Linux.

1.2.1. Algunos conceptos iniciales

Entrada en el sistema (login)

Antes de poder usar el sistema, deberá configurarse una cuenta de usuario. Esto es necesario, porque no es buena idea usar la cuenta de root para los usos normales. La cuenta de root debería reservarse para el uso de comandos privilegiados y para el mantenimiento del sistema, como se verá más adelante.

En el momento de presentarse en el sistema, vera una línea de comandos en la pantalla, invitándole a que introduzca su nombre de usuario. Después de teclearlo y pulsar la tecla "Intro" se le pedirá la palabra de paso o password. Si ésta es introducida incorrectamente, el sistema lanzará un mensaje de error "Login incorrect", y deberá intentarlo de nuevo.

```
nino login: pepe
Password:
```

Una vez que ha introducido correctamente el nombre de usuario y la palabra de paso, está oficialmente "presentado" en el sistema y libre para comenzar a trabajar.

Consolas virtuales (VCs)

La consola del sistema es el monitor y teclado conectado directamente al sistema. (Como UNIX es un sistema operativo multiusuario, puede tener otros terminales conectados a puertos serie del sistema, pero estos no serán la consola). Linux, como otras versiones de UNIX, proporciona acceso a consolas virtuales (o VC's), las cuales le permitirán tener mas de una sesión de trabajo activa desde la consola a la vez.

Para demostrar esto, entre en su sistema (como hemos visto antes). Ahora pulse alt-F2.Debería ver la pregunta login: de nuevo. Esta viendo la segunda consola virtual ha entrado en el sistema por la primera. Para volver a la primera VC, pulse alt-F1.

Un sistema Linux recién instalado probablemente le permita acceder a las primeras seis VC's, usando alt-F1 a alt-F6. Pero es posible habilitar hasta 12 VC's una por cada tecla de función del teclado. Como puede ver, el uso de VC's es muy potente puede estar trabajando en diferentes VC's a la vez.

Mientras que el uso de VC's es algo limitado (después de todo, solo puede mirar un VC cada vez), esto debería darle una idea de las capacidades multiusuario del sistema. Mientras esta trabajando en el VC 1, puede conmutar al VC 2 y comenzar a trabajar en otra cosa.

El interprete de comandos (shell): el BASH

En la mayoría de las exploraciones en el mundo de UNIX, estará hablando con el sistema a través del uso de un intérprete de comandos. Un intérprete de comandos es simplemente un programa que toma la entrada del usuario (p.e. las órdenes que teclea) y las traduce a instrucciones. Esto puede ser comparado con el COMMAND.COM de MS-DOS, el cual efectúa esencialmente la misma tarea. El intérprete de comandos es solo uno de los interfaces con UNIX. Hay muchos interfaces posibles, como el sistema X Windows, el cual le permite ejecutar comandos usando el ratón y el teclado. En la última sección de este capítulo veremos algo de este interface gráfico.

La shell, es una capa que protege al usuario de la máquina pura y dura y a él mismo. Gracias a la shell puedes introducir comandos, y te podrá hacer la vida más o menos fácil, dependiendo de la shell que se utilice.

Tan pronto como entra en el sistema, el sistema arranca un intérprete de comandos y Ud. ya puede teclear órdenes al sistema. Veamos un ejemplo rápido. Aquí, Pepe entra en el sistema y es situado en el intérprete de comandos

```
nino login: pepe
Password:

Welcome to nino!

pepe@nino:~$
```

"pepe@nino: $" es el "prompt" del intérprete de comandos, indicando que esta listo para recibir órdenes. El prompt puede variar de un sistema a otro y se puede cambiar fácilmente. En este caso, está compuesto del nomnbre de usuario + @ + nombre de la máquina + : + directorio actual + $. Como veremos más adelante, cada usuario tiene un directorio propio, donde guarda sus archivos y sus configuraciones, y que es llamado directorio "home". Este

directorio se representa, para mayor comodidad con el símbolo "∼" que podemos conseguir pulsando AltGR + 4.

Generalmente en Linux, la shell suele ser el "bash" (Bourne-Again SHell), que se caracteriza por no tener necesidad de teclear todos los comandos, o nombres de ficheros ya que cuando pulsas tabulador, terminará de escribir el resto. Por ejemplo, si escribe "mo" y pulsa dos veces la tecla "Tab" rápidamente, vemos que el bash ofrece una lista de los comandos que comienzan por "mo". Entre ellos, esta el comando "more". Si escribe una "r" a mayores, y volvemos a pulsar la tecla "Tab", completa la orden:

```
pepe@nino:~$ mo
more                            mozilla-1.0.0
mount                           mozilla-xremote-client
movemail                        mozilla-xremote-client-1.0.0
mozilla
pepe@nino:~$ more
```

Otra de las grandes ventajas de la shell BASH es que guarda un historial que recoje todos los comandos que hemos ido tecleando, de forma que si tenemos que repetir alguno o escribir uno parecido a algo que ya hayamos ejecutado, con los cursores podemos acceder fácilmente. Si comenzamos una nueva orden con un espacio (caracter en blanco), ese comando no será guardado en el historial del BASH. Ésto es sumamente útil cuando en el comando debemos escribir alguna contraseña o algo de lo que no queremos que el sistema, de ninguna manera, guarde constancia.

Tratemos de decirle ahora al sistema que haga algo:

```
pepe@nino:~$ make something
make: *** No hay ninguna regla para construir el objetivo 'something'.  Alto.
pepe@nino:~$
```

Bien, como resulta que make es el nombre de un programa ya existente en el sistema, el intérprete de comandos lo ejecuta. Esto nos lleva a una cuestión importante: ¿Que son órdenes? ¿Que ocurre cuando tecleamos "make something"? La primera palabra de la orden, "make", es el nombre de la orden a ejecutar. El resto de la orden es tomado como argumentos de la orden.

Cuando teclea una orden, el intérprete de comandos hace varias cosas. Primero de todo, busca el nombre de la orden y comprueba si es una orden interna. (Es decir, una orden que el propio intérprete de comandos sabe ejecutar por si mismo. Hay bastantes órdenes de ese tipo que veremos mas adelante). El intérprete de comandos también comprueba si la orden es un "alias" o nombre sustitutorio de otra orden. Si no se cumple ninguno de estos casos, el intérprete de comandos busca el programa y lo ejecuta pasándole los argumentos especificados en la línea de comandos.

```
pepe@nino:~$ echo hola
hola
pepe@nino:~$ dime hola
-bash: dime: command not found
pepe@nino:~$ alias dime=echo
pepe@nino:~$ dime hola
hola
pepe@nino:~$ unalias dime
pepe@nino:~$ dime hola
-bash: dime: command not found
```

Como se puede observar, el comando "echo" lleva como argumento lo que se quiera que repita por pantalla, haciendo eco de ello. Sin embargo el comando dime no existe, así que podemos hacer un alias. En ese momento, la orden "dime" actuará como el comando echo. Con "unalias", deshacemos el alias.

Por otra parte, existen en Linux un conjunto de variables denominadas variables de entorno. Dichas variables son creadas habitualmente por el propio shell en concordancia con el modo y las características dadas por e l administrador del sistema al dar de alta al usuario. Estas variables son las que definen el propio entorno de trabajo y son redefinibles por el usuario, lo que agrega flexibilidad al sistema y permite adaptar el entorno al gusto o necesidad de cada uno. La configuración del prompt, por ejemplo está definida por la variable de entorno "PS1". Estas variables se pueden ver y cambiar con el comando "set".

Cuando bash es invocado, como shell interactiva, lo primero que hace es leer y ejecutar los comandos localizados en el archivo /etc/profile luego de leer este archivo, busca los ficheros ∼/.bash_profile, ∼/.bash_login y ∼/.profile en este orden, leyendo y ejecutando solo lee el primero que encuentre. De esta manera podemos agregar las variables de entorno en ∼/.bash_profile por ejemplo para que de esta manera, cada vez que nos logueemos en el sistema e invoquemos bash, este leea nuestras variables localizadas en este archivo.

No nos extenderemos demasiado en el uso del shell, en este caso del bash. Pero debe quedar constancia de que es una de las herramientas más útiles en sistemas UNIX. En MSDOS existían procesos "batch", que eran pequeños programas conformados por una secuencia de comandos. Este concepto es mucho más amplio en las shells de UNIX, que nos ofrecen todo un lenguaje de programación llamado "scripting".

Cambiar la password

También debe asegurarse de la forma de cambiar su palabra de paso. La orden passwd le pedirá su palabra de paso vieja y la nueva. Volverá a pedir una segunda vez la nueva para validarla. Tenga cuidado de no olvidar su palabra de paso, si eso ocurre, deberá pedirle al administrador del sistema que la modifique por usted.

```
pepe@nino:~$ passwd
Changing password for pepe
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
Password changed
pepe@nino:~$
```

Salida del sistema

Desde la línea de comandos, para avandonar el sistema, en cualquier momento usaremos la orden "exit", que cerrara la sesión y volverá a mostrar la pantalla de loging.

```
pepe@nino:~$ exit

nino login:
```

1.2.2. La ayuda "on-line"

Prácticamente cada sistema UNIX, incluido Linux, proporciona una utilidad conocida como "paginas de manual". Estas paginas contienen documentación en línea para todas las órdenes del sistema, recursos, ficheros de configuración, etc.

La orden usada para acceder a las paginas de manual es man. Por ejemplo, si esta interesado en conocer otras opciones de la orden ls, puede escribir

```
pepe@nino:~$ man ls
```

y le será mostrada la pagina de manual para ls.

Desafortunadamente la mayoría de las paginas de manual han sido escritas por gente que ya conocía lo que la orden o recurso hacia, por esto, las paginas de manual usualmente solo contienen detalles técnicos de la orden sin ningún tipo de tutorial de uso. Pese a esto, estas paginas son una gran fuente de información que permiten refrescar la memoria si olvidamos la sintaxis de un comando. Igualmente, estas paginas le darán mucha información sobre órdenes que no trataremos en este libro.

Se sugiere que pruebe man con los comandos que ya hemos tratado y con los que vayamos introduciendo. Notara que alguno de los comandos no tiene pagina de manual. Esto puede ser debido a diferentes motivos. En primer lugar, las paginas no han sido escritas aun (el Proyecto de Documentación de Linux es también el responsable de las paginas de manual). En segundo lugar, la orden puede ser interna del intérprete de comandos, o un alias, en cuyo caso no tendrán una pagina propia. Un ejemplo es la orden cd la cual es interna del intérprete de comandos. El propio intérprete de comandos es quien procesa cd, no hay un programa separado.

Además de las páginas de manual, la mayoría de comandos y programas de Linux aceptan el parámetro –help o -h, que proporciona una información rápida de su funcionamiento y parámetros por pantalla.

```
pepe@nino:~$ cat --help
Modo de empleo: cat [OPCION] [FICHERO]\ldots
Concatenate FILE(s), or standard input, to standard output.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonblank output lines
  -e                       equivalent to -vE
  -E, --show-ends          display $ at end of each line
  -n, --number             number all output lines
  -r, --reversible         use \ to make the output reversible, implies -v
  -s, --squeeze-blank      never more than one single blank line
  -t                      equivalente a -vT
  -T, --show-tabs         muestra los caracteres de tabulacion como ^I
  -u                      (sin efecto)
  -v, --show-nonprinting  utiliza la notación ^ y M-, salvo para LFD y TAB
      --help     muestra esta ayuda y finaliza
      --version  informa de la version y finaliza

Sin FICHERO, o cuando FICHERO es -, lee la entrada estandar.

Comunicar bichos a <bug-coreutils@gnu.org>.
pepe@nino:~$
```

Por último, en los sistemas Linux existe un directorio /usr/share/doc con documentación de todo lo que tiene instalado el sistema operativo, y de donde se puede conseguir mucha documentación y muy útil.

Ejercicio: *modificar el formato del prompt, buscando la ayuda necesaria en el manual de bash.*

1.2.3. Navegación y gestión de ficheros

Bajo la mayoría de los sistemas operativos (UNIX incluido), existe el concepto de fichero, el cual es un conjunto de información al que se le ha asignado un nombre (llamado nombre del fichero).

Ejemplos de fichero son un mensaje de correo, o un programa que puede ser ejecutado. Esencialmente, cualquier cosa salvada en el disco es guardada en un fichero individual.

Los ficheros son identificados por sus nombres. Por ejemplo, el fichero que contiene su historial podría ser salvado con el nombre history-paper. Estos nombres usualmente identifican el fichero y su contenido de alguna forma significativa para usted. No hay un formato

estándar para los nombres de los ficheros como lo hay en MS-DOS y en otros sistemas operativos; en general estos pueden contener cualquier carácter (excepto /), y están limitados a 256 caracteres de longitud.

Con el concepto de fichero aparece el concepto de directorio. Un directorio es simplemente una colección de ficheros. Puede ser considerado como una "carpeta" que contiene muchos ficheros diferentes. Los directorios también tienen nombre con el que los podemos identificar. Además, los directorios mantienen una estructura de árbol; es decir, directorios pueden contener otros directorios.

Un fichero puede ser referenciado por su nombre con camino (mas conocido por "path"), el cual esta constituido por su nombre, antecedido por el nombre del directorio que lo contiene. Por ejemplo, supongamos que Pepe tiene un directorio de nombre papers que contiene tres ficheros: history-final, english-lit y masters-thesis. (Cada uno de los tres ficheros contiene información sobre tres de los proyectos en los que Pepe esta trabajando). Para referirse al fichero english-lit, Pepe puede especificar su camino:

```
papers/english-lit
```

Como puede ver, el directorio y el nombre del fichero van separados por un carácter /. Por esta razón, los nombres de fichero no pueden contener este carácter. Los usuarios de MS-DOS encontraran esta convención familiar, aunque en el mundo MS-DOS se usa el carácter "\").

Como hemos mencionado, los directorios pueden anidarse uno dentro de otro. Por ejemplo, supongamos que Pepe tiene otro directorio dentro de papers llamado cheat-sheet. El path de este fichero seria

```
papers/notes/cheat-sheet
```

Por lo tanto, el path realmente es la "ruta" que se debe tomar para localizar a un fichero. El directorio sobre un subdirectorio dado es conocido como el directorio padre. Aquí, el directorio papers es el padre del directorio notes.

Por último, decir que en el Apéndice B hay una tabla de comparación de comandos de MSDOS con comandos de UNIX, útil para aquellos usuarios familiarizados con el entorno DOS.

Estructura de directorios

La mayoría de los sistemas UNIX tienen una distribución de ficheros estándar, de forma que recursos y ficheros puedan ser fácilmente localizados. Esta distribución forma el árbol de directorios, el cual comienza en el directorio "/", también conocido como "directorio raíz". Directamente por debajo de / hay algunos subdirectorios importantes: /bin, /etc, /dev y /usr, entre otros. Estos a su vez contienen otros directorios con ficheros de configuración del sistema, programas, etc.

En particular, cada usuario tiene un directorio "home". Este es el directorio en el que el usuario guardara sus ficheros. En los ejemplos anteriores, todos los ficheros de Pepe (como cheat-sheer y history-final) estaban contenidos en el directorio home de Pepe. Usualmente, los directorios home de los usuarios cuelgan de /home y son nombrados con el nombre del usuario al que pertenecen.

Por lo tanto, el directorio "home" de Pepe es /home/pepe.

Como ejemplo, las librerías se encuentran en /lib, /usr/lib y /usr/local/bin, la configuración en /etc y los ficheros de log y estadísticas en /var/spool, /var/log y /var/adm. Las páginas de manual están en /usr/man, los archivos de código fuente en /usr/src y los del entorno gráfico en /usr/X11. Los archivos temporales estarán en /tmp, y la comunicación con parámetros del sistema se hace desde el directorio /proc.

```
pepe@nino:/$ ls
bin/   cdrom@  etc/    home/  initrd/  lost+found/  opt@   root/
boot/  dev/    floppy/ Inet@  lib/    mnt/         proc/  sbin/
```

Uno de estos directorios merece una atención especial, el directorio /dev. Los "ficheros" en /dev son conocidos como controladores de dispositivo (device drivers), son usados para acceder a los dispositivos del sistema y recursos, como discos duros, módems, memoria, etc. Por ejemplo, de la misma forma que puede leer datos de un fichero, puede leerla desde la entrada del ratón leyendo /dev/mouse.

Los ficheros que comienzan su nombre con fd son controladores de disqueteras. fd0 es la primera disquetera, fd1 la segunda. Ahora, alguien astuto se dará cuenta de que hay mas controladores de dispositivo para disqueteras de los que hemos mencionado. Estos representan tipos específicos de discos. Por ejemplo, fd1H1440 accederá a discos de 3.5' de alta densidad en la disquetera 1.

Aquí tenemos una lista de algunos de los controladores de dispositivo mas usados. Nótese que incluso aunque puede que no tenga alguno de los dispositivos listados, tendrá entradas en dev de cualquier forma.

- /dev/console

  Hace referencia a la consola del sistema, es decir, al monitor conectado directamente a su sistema.

- /dev/ttyS_ y /dev/cua_

  Usados para acceder a los puertos serie. Por ejemplo, /dev/ttyS0 hace referencia a "COM1" bajo MS-DOS. Los dispositivos /dev/cua son "callout", los cuales son usados en conjunción con un módem.

- /dev/hd_

  Los nombres de dispositivo que comienzan por hd acceden a discos duros. /dev/hda hace referencia a la totalidad del primer disco duro, mientras que /dev/hda1 hace referencia a la primera partición en /dev/hda.

- /dev/sd_

  Los nombres de dispositivo que comienzan con sd son dispositivos SCSI. Si tiene un disco duro SCSI, en lugar de acceder a él mediante /dev/hda, deberá acceder a /dev/sda. Las cintas SCSI son accedidas vía dispositivos st y los CD-ROM SCSI vía sr.

- /dev/lp_

  Los nombres que comienzan por lp acceden a los puertos paralelo. /dev/lp0 hace referencia a "LPT1" en el mundo MS-DOS.

- /dev/null

  Usado como "agujero negro", cualquier dato enviado a este dispositivo desaparece. ¿Para que puede ser útil esto?. Bien, si desea suprimir la salida por pantalla de una orden, podría enviar la salida a /dev/null.

- /dev/tty_

  Los nombres que comienzan por /dev/tty hacen referencia a "consolas virtuales" de su sistema (accesibles mediante las Alt+F1, Alt+F2, etc).

- /dev/pty_

  Los nombres de dispositivo que comienzan con /dev/pty son "pseudoterminales". Estos son usados para proporcionar un "terminal" a sesiones remotas. Por ejemplo, si su maquina esta en una red, telnet de entrada usara uno de los dispositivos /dev/pty.

Estos directorios no son más que ejemplos de la jerarquía de directorios de linux y de su organización. Para una referencia más detallada, acúdase al Apéndice A.

Moverse por el entorno

Ahora que ya podemos presentarnos como usuarios, y sabemos como indicar ficheros con su camino completo, ¿como podemos cambiar nuestro directorio de trabajo?.

Para empezar, debemos conocer el lugar en el árbol de directorios donde nos encontramos actualmete. Ésto se puede llevar a cabo gracias al comando "pwd".

```
pepe@nino:~$ pwd
/home/pepe
pepe@nino:~$
```

La orden para movernos por la estructura de directorios es cd, abreviación de "cambio de directorio". Hay que destacar, que la mayoría de las órdenes Unix mas usadas son de dos o tres letras. La forma de uso de la orden cd es:

```
cd <directorio>
```

donde ¡directorio¿es el nombre del directorio al que queremos ir.

Como dijimos, al entrar al sistema comenzamos en el directorio "home". Si Pepe quiere ir al subdirectorio papers...

```
pepe@nino:~$ cd papers
pepe@nino:~/papers$
```

Cada directorio tiene una entrada de nombre ".." la cual se refiere al directorio padre. De igual forma, existe en cada directorio la entrada "." la cual se refiere a si mismo. De esta forma, la siguiente secuencia hace que Pepe salga del directorio papers ("cd ..") y a continuación se queda en el mismo directorio ("cd .").

```
pepe@nino:~/papers$ cd ..
pepe@nino:~$ cd .
pepe@nino:~$
```

También pueden usarse nombres con el camino absoluto en la orden cd. Para ir al directorio /usr/lib con cd, introduciremos la siguiente orden...

```
pepe@nino:~$ cd /usr/lib
pepe@nino:/usr/lib$
```

Por último, usando cd sin argumentos nos llevara a nuestro directorio de origen (home)...

```
pepe@nino:/usr/lib$ cd
pepe@nino:~$
```

Mirando el contenido de los directorios

Ahora que ya sabe como moverse por los directorios probablemente pensara: ¿Y bien? El simple movimiento por el árbol de directorios es poco útil, necesitamos un nuevo comando, ls. ls muestra por el terminal la lista de ficheros y directorios, por defecto, los del directorio activo. Por ejemplo...

```
pepe@nino:~$ ls
letters  Mail  papers
```

Aquí podemos ver que Pepe tiene tres entradas en su directorio actual: Mail, letters y papers. Esto no nos dice demasiado, ¿son ficheros o directorios?. Podemos usar la opción -F de la orden ls para obtener mas información...

```
pepe@nino:~$ ls -F
letters/  Mail/  papers/
```

Por el carácter / añadido a cada nombre sabemos que las tres entradas son subdirectorios. La orden ls -F puede también añadir al final "*", esto indica que es un fichero ejecutable. Si ls -F no añade nada, entonces es un fichero normal, es decir no es ni un directorio ni un ejecutable.

Por lo general cada orden UNIX puede tomar una serie de opciones definidas en forma de argumentos. Estos usualmente comienzan con el carácter "-", como vimos antes con ls -F. La opción -F le dice a ls que de mas información sobre el tipo de ficheros, en este caso añadiendo un / detrás de cada nombre de un directorio.

Si a ls le pasamos un nombre de directorio, mostrará el contenido de ese directorio...

```
pepe@nino:~$ ls -F papers
english-lit  history-final  master-tesis  notes/
```

Se puede, por ejemplo, listar los archivos que hay en /usr/bin, en donde se guardan los ejecutables del sistema:

```
pepe@nino:~$ ls -F /usr/bin
```

Otros dos parámetros interesantes del comando ls son -l, -R y -a que nos devolverán un listado detallado de los archivos, un listado recursivo y todos los archivos ocultos que haya, respectivamente. El listado detallado, se puede probar en el siguiente ejemplo; nos da información sobre el total de archivos y carpetas, y una información muy detallada de cada uno: los permisos (de los que hablaremos más adelante), el usuario propietario y el grupo, el tamaño, la fecha y hora de creación y el nombre del archivo.

```
pepe@nino:~$ ls -l
pepe@nino:~/papers$ ls -l
total 4
-rw-r--r--    1 pepe     pepe            0 2002-12-03 20:44 english-lit
-rw-r--r--    1 pepe     pepe            0 2002-12-03 20:44 history-final
-rw-r--r--    1 pepe     pepe            0 2002-12-03 20:44 master-tesis
drwxr-xr-x    2 pepe     pepe         4096 2002-12-03 20:44 notes
pepe@nino:~/papers$
```

El listado recursivo (-R) hace que el comando ls entre de forma recursiva en todas los subdirectorios del directorio que listamos. El resultado se observa en el siguiente ejemplo.

```
pepe@nino:~$ ls -R
.:
letters  Mail  papers

./letters:

./Mail:

./papers:
english-lit  history-final  master-tesis  notes

./papers/notes:
pepe@nino:~$
```

Todos estos argumentos, se pueden convinar, de forma que podemos pedir un listado de todos los archivos del directorio home, pero detallado y mostrando los archivos ocultos (en UNIX todo archivo que comience por el caracter '.' es considerado oculto).

```
pepe@nino:~$ ls -l -a
total 40
drwxr-xr-x    7 pepe     pepe         4096 2002-12-03 20:50 .
drwxrwsr-x   23 root     staff        4096 2002-12-03 12:29 ..
-rw-------    1 pepe     pepe          744 2002-12-04 16:47 .bash_history
-rw-r--r--    1 pepe     pepe          556 2002-12-03 12:29 .bash_profile
-rw-r--r--    1 pepe     pepe         1123 2002-12-03 12:29 .bashrc
drwxr-xr-x    2 pepe     pepe         4096 2002-12-03 20:39 letters
drwx------    2 pepe     pepe         4096 2002-12-03 20:40 Mail
drwxr-xr-x    3 pepe     pepe         4096 2002-12-03 20:44 papers
drwx------    2 pepe     pepe         4096 2002-12-03 17:17 .ssh
pepe@nino:~$
```

Normalmente en el directorio home se guardan archivos ocultos con configuraciones de programas que usamos. En este caso, se pueden observar varios del interprete de comandos (BASH).

Trate de moverse por varios directorios usando ls y cd. En algunos casos podrá encontrarse el desagradable mensaje de error "Permission denied". Esto simplemente es debido a cuestiones de seguridad del UNIX. Para poder moverse o listar un directorio debe de tener permisos para poder hacerlo. Hablaremos sobre ello más adelante.

Ejercicio: *probar los siguientes parámetros del comando "ls": -h, -R, -r, -S, -k. Poner algunos ejemplos de cada uno de ellos y de su convinación.*

Crear nuevos directorios

Es el momento de aprender a crear directorios. Para ello se usa la orden mkdir. Pruebe lo siguiente:

```
pepe@nino:~$ mkdir foo
pepe@nino:~$ ls -F
foo/  letters/  Mail/  papers/
pepe@nino:~$ cd foo
pepe@nino:~/foo$ ls
pepe@nino:~/foo$
```

Acaba de crear un nuevo directorio, pero dentro no hay ningún fichero.

Ejercicio: *intente crear un subdirectorio de otro directorio que no exista. Por ejemplo, en su home cree "hola/quetal". Este ejercicio se deberá efectuar utilizando un sólo comando. Luego busque en el manual alguna opción para saber qué es exactamente lo que mkdir está haciendo*

Copia de ficheros

La copia de ficheros es efectuada por la orden cp. Ésta utiliza dos argumentos: el primero, el archivo de origen, y el segundo, la carpeta o archivo de destino. Si el segundo es una carpeta, copiará el archivo dentro de esa carpeta. Si es un archivo, lo copiará con ese nombre de archivo.

```
pepe@nino:~/foo$ cp /etc/fstab .
pepe@nino:~/foo$ cp /usr/lib/libc.so .
pepe@nino:~/foo$ ls -F
fstab  libc.so
pepe@nino:~/foo$ cp fstab otra-cosa
pepe@nino:~/foo$ ls
fstab  libc.so  otra-cosa
pepe@nino:~/foo$
```

Nótese como se usa el directorio "." para referirnos al directorio actual. Existe de nuevo el parámetro -R para que la copia sea recursiva, es decir, se copien todos los archivos y subdirectorios (y sus respectivos contenidos).

Moviendo ficheros

La orden mv mueve ficheros en lugar de copiarlos. La sintaxis es muy parecida a la de cp...

```
pepe@nino:~/foo$ ls
fstab  libc.so  otra-cosa
pepe@nino:~/foo$ mv fstab tablas
pepe@nino:~/foo$ ls -F
libc.so  otra-cosa  tablas
pepe@nino:~/foo$
```

Nótese como fstab ya no existe, en su lugar esta el fichero tablas. Esta orden puede usarse para renombrar ficheros, como acabamos de hacer, pero también para mover ficheros a directorios diferentes. También soporta el parámetro -R, para movimientos recursivos.

Téngase en cuenta que tanto mv como cp sobreescribirán los ficheros destino (si ya existen) sin consultar. Sea cuidadoso cuando mueva un fichero a otro directorio, porque puede haber ya un fichero con el mismo nombre que será sobrescrito.

Borrando ficheros y directorios

Para borrar un fichero, use la orden rm. ("rm" viene de "remove"). Los argumentos son los ficheros a ser borrados.

```
pepe@nino:~/foo$ ls
libc.so  otra-cosa  tablas
pepe@nino:~/foo$ rm libc.so otra-cosa
pepe@nino:~/foo$ ls
tablas
pepe@nino:~/foo$
```

Para borrar directorios, éstos deben estar vacíos. Si se pretende borrar un directorio y todos sus contenidos, se puede utilizar el parámetro -r.

```
pepe@nino:~$
pepe@nino:~$ cd foo
pepe@nino:~/foo$ ls
tablas
pepe@nino:~/foo$ cd ..
pepe@nino:~$ rm foo
rm: no se puede borrar 'foo': Es un directorio
pepe@nino:~$ rm -r foo
pepe@nino:~$ ls
letters  Mail  papers
pepe@nino:~$
```

Reconocer y ver ficheros

El comando "file" es capaz de reconocer el tipo del fichero que se le pase como parámetro. Veamos algunos ejemplos:

```
pepe@nino:~/foo$ file hello.c
hello.c: ASCII C program text
pepe@nino:~/foo$ file cap1.tex
cap1.tex: LaTeX document text
```

Las órdenes "more" y "cat" son usadas para ver el contenido de ficheros. more muestra el fichero pantalla a pantalla mientras que cat lo muestra entero de una vez. Como ejemplo se pueden ejecutar los siguientes comandos...

```
pepe@nino:$ cat tablas
pepe@nino:$ more tablas
```

Durante la ejecución de more pulse "Space" para avanzar a la pagina siguiente y "b" para volver a la pagina anterior. Hay otros comandos disponibles, los citados son solo los mas básicos. "q" finalizará la ejecución de more.

Ejercicio: *utilizando los comandos vistos en esta sección, trate de identificar al menos 5 tipos distintos de fichero, e intente ver su contenido.*

Redireccionar la salida de un comando

Normalmente la salida de los comandos se direcciona a la denominada "Salida estándar". Pero se puede variar el flujo de la salida hacia, por ejemplo, ficheros. Para esto, se utiliza el símbolo "mayor que" (>). En el siguiente ejemplo, se hace un listado del contenido de la carpeta home, y se redirecciona al nuevo archivo "lista". Después se puede observar mediante un "cat" que el fichero tiene, efectivamente, el listado...

```
pepe@nino:~$ ls -l > lista
pepe@nino:~$ ls
letters  lista  Mail  papers
pepe@nino:~$ cat lista
total 12
drwxr-xr-x    2 pepe     pepe         4096 2002-12-03 20:39 letters
-rw-r--r--    1 pepe     pepe            0 2002-12-05 12:28 lista
drwx------    2 pepe     pepe         4096 2002-12-03 20:40 Mail
drwxr-xr-x    3 pepe     pepe         4096 2002-12-03 20:44 papers
pepe@nino:~$
```

Si lo que se busca es añadir ma texto a un fichero con la redirección, en vez de el símbolo ">", que sobrescribiría el fichero de destino en caso de que existiera, se utiliza ">>", que, por contrario, concatena. Es decir, continuando el ejemplo anterior, si utilizamos ahora ">>" con un comando "echo" lo concatenaría con el anterior...

```
pepe@nino:~$ echo Ahora estoy en el final del fichero >> lista
pepe@nino:~$ cat lista
total 12
drwxr-xr-x    2 pepe     pepe         4096 2002-12-03 20:39 letters
-rw-r--r--    1 pepe     pepe            0 2002-12-05 12:28 lista
drwx------    2 pepe     pepe         4096 2002-12-03 20:40 Mail
drwxr-xr-x    3 pepe     pepe         4096 2002-12-03 20:44 papers
Ahora estoy en el final del fichero
pepe@nino:~$
```

Los comodines

Una característica importante de la mayoría de los interpretes de comandos en Unix es la capacidad para referirse a mas de un fichero usando caracteres especiales. Estos llamados *comodines* le permiten referirse a, por ejemplo, todos los ficheros que contienen el carácter "ñ".

El comodín "*" hace referencia cualquier carácter o cadena de caracteres en el fichero. Por ejemplo, cuando usa el carácter "*" en el nombre de un fichero, el intérprete de comandos lo sustituye por todas las combinaciones posibles provenientes de los ficheros en el directorio al cual nos estamos refiriendo.

Veamos un ejemplo: supongamos que Pepe quiere listar todos los archivos y directorios que contengan una "a" en su nombre.

```
pepe@nino:~/papers$ ls
english-lit   history-final   master-tesis   notes
pepe@nino:~/papers$ ls *a*
history-final   master-tesis
pepe@nino:~/papers$
```

Como puede ver, el comodín "*" ha sido sustituido con todas las combinaciones posibles que coincidían de entre los ficheros del directorio actual.

El uso de "*" solo, simplemente se refiere a todos los ficheros, puesto que todos los caracteres coinciden con el comodín.

El uso de tuberías (pipes)

El uso de pipes es otra caracteristica del intérprete de comandos, que nos permite conectar una cadena de comandos en un "pipe", donde la salida estandar (stdout) del primero es enviada directamente a la entrada estandar (stdout) del segundo y así sucesivamente.

En el siguiente ejemplo, sacamos un listado del contenido de la carpeta /usr/bin. Esta carpeta tiene un gran cantidad de archivos, que veremos parar rápidamente por pantalla. Para poder verlos todos, queremos que se nos vayan mostrando paso a paso por pantalla (comando more)...

```
pepe@nino:~/papers$ ls /usr/bin | more
[
411toppm
a2p
aaaa
access
addr2line
afm2tfm
allcm
allec
allneeded
amstex
anytopnm
appletviewer
apropos
apt-cache
apt-cdrom
apt-config
apt-extracttemplates
apt-ftparchive
apt-get
aptitude
apt-sortpkgs
--Mas--
```

Los enlaces (*links*)

Los enlaces le permiten dar a un único fichero múltiples nombres. Los ficheros son identificados por el sistema por su numero de inodo, el cual es el único identificador del fichero para el sistema de ficheros . Un directorio es una lista de números de inodo con sus correspondientes nombres de fichero. Cada nombre de fichero en un directorio es un enlace a un inodo particular.

Existen dos tipos de enlace, los duros y los blandos. Hablaremos sólo de los blandos, puesto que son los más utilizados a nivel de usuario. Éstos, también conocidos como *enlaces simbólicos*, permiten crear un fichero que apunte a otro. A efectos prácticos, estos enlaces funcionan igual que trabajando directamente sobre el fichero: si se tiene un fichero de texto

con una carta, por ejemplo, y además un enlace a esta carta, podemos modificar la carta modificando su enlace.

Para crear enlaces se utiliza el comando "ln", y utilizando la opción "-s" se consiguen los enlaces simbólicos, de forma que el formato sería...

```
ln -s <fichero origen> <nombre del enlace destino>
```

Veamos entonces un ejemplo práctico: creamos un link (enlace) simbólico al fichero "notes", llamado "notes_link", y probamos a modificarlo, viendo posteriormente como se ha moficado también el archivo base, "notes":

```
pepe@nino:~/papers$ ls
english-lit  history-final  master-tesis  notes
pepe@nino:~/papers$ ln -s notes notes_link
pepe@nino:~/papers$ ls
english-lit  history-final  master-tesis  notes notes_link
pepe@nino:~/papers$ cat notes
Hola, en este archivo hay notas.
pepe@nino:~/papers$ echo Y ahora hay mas notas >> notes_link
pepe@nino:~/papers$ cat notes
Hola, en este archivo hay notas.
Y ahora hay mas notas
pepe@nino:~/papers$
```

Ejercicio: *trate de crear un enlace a otro enlace previamente creado, comprobando así la recursividad del comando.*

1.2.4. Control de tareas

El control de tareas es una utilidad incluida en muchos shells (incluido el BASH), que permite el control de multitud de comandos o tareas al momento, mediante otros comandos propios del shell. Antes de seguir, deberemos hablar un poco sobre los procesos.

Procesos

Cada vez que usted ejecuta un programa, usted lanza lo que se conoce como proceso, que es simplemente el nombre que se le da a un programa cuando se esta ejecutando. El comando ps visualiza la lista de procesos que se están ejecutando actualmente, por ejemplo:

```
pepe@nino:~$ ps
  PID TTY          TIME CMD
13594 tty3     00:00:00 bash
18519 tty3     00:00:00 ps
pepe@nino:~$
```

La columna PID representa el identificador de proceso. La última columna COMMAND, es el nombre del proceso que se esta ejecutando. Ahora solo estamos viendo los procesos que esta ejecutando pepe. Hay muchos mas procesos aparte de estos corriendo en el sistema, para verlos todos, teclearemos el comando "ps -aux".

Vemos que hay dos procesos, bash (que es la shell o intérprete de comandos que usa Pepe), y el propio comando ps. Como puede observar, la bash se ejecuta concurrentemente con el comando ps. La bash ejecuto ps cuando Pepe tecleo el comando. Cuando ps termina de ejecutarse (después de mostrar la tabla de procesos), el control retorna al proceso bash, que muestra el prompt, indicando que esta listo para recibir otro comando.

Un proceso que esta corriendo se denomina tarea para el shell. Los términos proceso y tarea, son intercambiables. Sin embargo, se suele denominar "tarea" a un proceso, cuando es usado en conjunción con control de tareas, que es un rasgo del shell que permite cambiar entre distintas tareas.

En muchos casos, los usuarios solo ejecutan un trabajo cada vez, que es el último comando que ellos teclearon desde el shell. Sin embargo, usando el control de tareas, usted podrá ejecutar diferentes tareas al mismo tiempo, cambiando entre cada uno de ellos conforme lo necesite. ¿Cuan beneficioso puede llegar a ser esto?. Supongamos que esta usted con su procesador de textos, y de repente necesita parar y realizar otra tarea, con el control de tareas, usted podrá suspender temporalmente el editor, y volver al shell para realizar cualquier otra tarea, y luego regresar al editor como si no lo hubiese dejado nunca.

Ejercicio: *Basándose en las páginas de manual, pruebe las siguientes opciones del comando ps por separado y conjuntamente: -f, -a, -e, -u, -x, -g*

Planos de ejecución

Un proceso puede estar en Primer plano o en Segundo plano. Solo puede haber un proceso en primer plano al mismo tiempo, el proceso que esta en primer plano, es el que interactúa con usted recibe entradas de teclado, y envía las salidas al monitor. (Salvo, por supuesto, que haya redirigido la entrada o la salida, como ya se ha explicado anteriormente). El proceso en segundo plano, no recibe ninguna señal desde el teclado, por lo general, se ejecutan en silencio sin necesidad de interacción.

Algunos programas necesitan mucho tiempo para terminar, y no hacen nada interesante mientras tanto. Compilar programas es una de estas tareas, así como comprimir un fichero grande. No tiene sentido que se siente y se aburra mientras estos procesos terminan. En estos casos es mejor lanzarlos en segundo plano, para dejar el ordenador en condiciones de ejecutar otro programa.

Los procesos pueden ser suspendidos. Un proceso suspendido es aquel que no se esta ejecutando actualmente, sino que esta temporalmente parado. Después de suspender una tarea, puede indicar a la misma que continúe, en primer plano o en segundo, según necesite. Retomar una tarea suspendida no cambia en nada el estado de la misma, la tarea continuara ejecutándose justo donde se dejó.

Tenga en cuenta que suspender un trabajo no es lo mismo que interrumpirlo. Cuando usted interrumpe un proceso (generalmente con la pulsación de Ctrl+C, el proceso muere, y deja de estar en memoria y utilizar recursos del ordenador. Una vez eliminado, el proceso no puede continuar ejecutándose, y deberá ser lanzado otra vez para volver a realizar sus tareas. También se puede dar el caso de que algunos programas capturan la interrupción, de modo que pulsando Ctrl+C no se para inmediatamente. Esto se hace para permitir al programa realizar operaciones necesarias de limpieza antes de terminar (tiempo necesario para guardar algunos registros, etc.). De hecho, algunos programas simplemente no se dejan matar por ninguna interrupción.

Por último, comentar que existe una utilidad llamada "top" que muestra los procesos activos, la CPU y memoria usada, por cada proceso, etc. Para invocarlo, simplemente teclee "top". Dentro del programa, la tecla "h" le mostrará toda la ayuda necesaria.

Ejercicio: *poner un par de ejercicios*

Gestión de procesos

Empecemos con un ejemplo sencillo. El comando yes es un comando aparentemente inútil que envía una serie interminable de y-es a la salida estándar. (Realmente es muy útil. Si se utiliza una tubería (o "pipe") para unir la salida de yes con otro comando que haga preguntas del tipo si/no, la serie de y-es confirmara todas las preguntas.)

```
pepe@nino:~$ yes
y
y
y
y
y
...
```

La serie de y-es continuara hasta el infinito, a no ser que usted la elimine, pulsando la tecla de interrupción, generalmente Ctrl+C. También puede deshacerse de esta serie de y-es redigiriendo la salida estándar de yes hacia /dev/null, que como recordara es una especie de "agujero negro" o papelera para los datos. Todo lo que usted envíe allí, desaparecerá.

```
pepe@nino:~$ yes > /dev/null
```

Ahora va mucho mejor, el terminal no se ensucia, pero el prompt de la shell no retorna. Esto es porque yes sigue ejecutándose y enviando esos inútiles y-es a /dev/null. Para recuperarlo, pulse la tecla de interrupción.

Supongamos ahora que queremos dejar que el comando yes siga ejecutándose, y volver al mismo tiempo a la shell para trabajar en otras cosas. Para ello nos enviaremos a yes a segundo plano, lo que nos permitirá ejecutarlo, pero sin necesidad de interacción.

Una forma de mandar procesos a segundo plano es añadiendo un carácter "&" al final de cada comando.

```
pepe@nino:~$ yes > /dev/null &
[1] 19616
pepe@nino:~$
```

Como podrá ver, ha regresado a la shell. ¿Pero qué es eso de "[1] 19616"?, ¿se está ejecutando realmente el comando yes?

"[1]" representa el número de tarea del proceso yes. La shell asigna un número a cada tarea que se este ejecutando. Como yes es el único comando que se esta ejecutando, se le asigna el número de tarea 1. El número "164" es el número de identificación del proceso, o PID, que es el número que el sistema le asigna al proceso. Ambos números pueden usarse para referirse a la tarea como veremos después.

Ahora usted tiene el proceso yes corriendo en segundo plano, y enviando constantemente la señal y hacia el dispositivo /dev/null. Para chequear el estado del proceso, utilice el comando interno de la shell jobs...

```
pepe@nino:~$ jobs
[1]+  Running                 yes >/dev/null &
pepe@nino:~$
```

¡Ahí está!. También puede usar el comando ps, como mostramos antes, para comprobar el estado de la tarea.

Para eliminar una tarea, utilice el comando kill. Este comando toma como argumento un número de tarea o un número de ID de un proceso. Cuando se identifica la tarea con el número de tarea, se debe preceder el número con el carácter de porcentaje ("%").

```
pepe@nino:~$ kill %1
pepe@nino:~$ jobs
[1]+  Terminado               yes >/dev/null
pepe@nino:~$
```

La tarea esta, en efecto, muerta, y si usa el comando jobs de nuevo, no mostrará nada. También podrá matar la tarea usando el número de ID de proceso (PID), el cual se muestra conjuntamente con el ID de tarea cuando arranca la misma. En nuestro ejemplo el ID de proceso es 164, así que el comando

```
pepe@nino:~$ kill 164
```

es equivalente a

```
pepe@nino:~$ kill %1
```

Hay otra manera de poner una tarea en segundo plano. Usted puede lanzarlo como un proceso normal (en primer plano), pararlo, y después relanzarlo en segundo plano. Primero, lance el proceso yes en primer plano como lo haría normalmente. De nuevo, dado que yes corre en primer plano, no debe retornar el prompt de la shell. Ahora, en vez de interrumpir la tarea con Ctrl+C, suspenderemos la tarea. El suspender una tarea no la mata: sólamente la detiene temporalmente hasta que Ud. la retoma. Para hacer esto usted debe pulsar la tecla de suspender, que suele ser Ctrl+Z.

```
pepe@nino:~$ yes > /dev/null

[Ctrl+Z]

[1]+  Stopped                 yes >/dev/null
pepe@nino:~$
```

Mientras el proceso esta suspendido, simplemente no se esta ejecutando. No gasta tiempo de CPU en la tarea. Sin embargo, usted puede retomar el proceso de nuevo como si nada hubiera pasado. Continuara ejecutándose donde se dejó.

Para relanzar la tarea en primer plano, use el comando fg (del ingles “foreground”).

```
pepe@nino:~$ fg
yes >/dev/null
```

La shell muestra el nombre del comando de nuevo, de forma que tenga conocimiento de que tarea es la que ha puesto en primer plano. Pare la tarea de nuevo, con Ctrl+Z. Esta vez utilice el comando bg para poner la tarea en segundo plano. Esto hará que el comando siga ejecutándose igual que si lo hubiese hecho desde el principio con “&” como en la sección anterior.

```
pepe@nino:~$ fg
yes >/dev/null

[Ctrl+Z]

[1]+  Stopped                 yes >/dev/null
pepe@nino:~$ bg
[1]+ yes >/dev/null &
pepe@nino:~$
```

Y tenemos de nuevo el prompt. El comando jobs debería decirnos que yes se esta ejecutando, y podemos matar la tarea con kill tal y como lo hicimos antes.

¿Cómo podemos parar la tarea de nuevo? Si pulsa Ctrl+Z no funcionará, ya que el proceso esta en segundo plano. La respuesta es poner el proceso en primer plano de nuevo, con el comando fg, y entonces pararlo. Como puede observar podrá usar fg tanto con tareas detenidas, como con las que estén segundo plano.

Hay una gran diferencia entre una tarea que se encuentra en segundo plano, y una que se encuentra detenida. Una tarea detenida es una tarea que no se esta ejecutando, es decir, que no usa tiempo de CPU, y que no esta haciendo ningún trabajo (la tarea aun ocupa un lugar en memoria, aunque puede ser volcada a disco). Una tarea en segundo plano, se esta ejecutando, y usando memoria, a la vez que completando alguna acción mientras usted hace otro trabajo. Sin embargo, una tarea en segundo plano puede intentar mostrar texto en su terminal, lo que puede resultar molesto si esta intentando hacer otra cosa. Por ejemplo, si usted uso el comando

```
pepe@nino:~$ yes &
```

sin redirigir stdout a /dev/null, una cadena de y-es se mostraran en su monitor, sin modo alguno de interrumpirlo (no puede hacer uso de Ctrl+C para interrumpir tareas en segundo plano). Para poder parar esas interminables y-es, tendría que usar el comando fg para pasar la tarea a primer plano, y entonces usar Ctrl+C para matarla.

Solo recordarle que el uso de control de tareas es una utilidad de la shell. Los comandos fg, bg y jobs son internos de la shell. Si por algún motivo usted utiliza una shell que no soporta control de tareas, no espere disponer de estos comandos.

1.2.5. Otros comandos útiles

Compresión y descompresión

En Linux los archivos normalmente se comprimen utilizando la herramienta “tar” y “GZip”. La primera no comprime, simplemente es capaz de agrupar varios archivos en uno solo (y desagruparlos, claro). La segunda es la que realmente comprime. El proceso suele ser

de la siguiente forma: primero se empaquetan todos los archivos y directorios en un archivo tar y después se comprimen con gzip. De todas formas, todo este proceso se realiza con un solo comando, ya que la opción de comprimir utilizando gzip ya ha ido incluida en la herramienta tar.

Entonces, la herramienta tar acepta los siguientes parámetros: -c, -f nombre_del_archivo, -z, -v. Los significados son los que siguen: "-c" indica que queremos empaquetar; "-f" seguido del nombre del archivo para crear el nuevo fichero; "-z" para comprimir usando gzip y "-v" para indicar modo "verbose" (modo detallado). El modo detallado simplemente enseña por pantalla lo que está haciendo el tar.

```
pepe@nino:~/papers$ tar -c -f cosas.tar.gz -z -v *
english-lit
history-final
master-tesis
notes/
pepe@nino:~/papers$ ls
cosas.tar.gz  english-lit  history-final  master-tesis  notes
pepe@nino:~/papers$
```

En este ejemplo se comprime todos los archivos y directorios de la carpeta en la que nos encontramos (*). Normalmente, los archivos que empaquetan a otros usan la extensión .tar y, si además están comprimidos, .tar.gz.

Para descomprimir archivos, se utilizan los mismos parámetros, pero se interprentan de manera diferente. No se usa el -c, que es específico para comprimir, y en vez de eso, se utiliza -x, para extraer. Ahora, -f sirve para indicar qué archivo queremos descomprimir, y -z indica que el fichero a descomprimir usa compresión gzip.

Como ejemplo, movemos el archivo que acabamos de crear (cosas.tar.gz) al directorio /tmp (directorio de archivos temporales del sistema), creamos allí una carpeta y lo descomprimimos...

```
pepe@nino:~/papers$ mkdir /tmp/prueba
pepe@nino:~/papers$ mv cosas.tar.gz /tmp/prueba/
pepe@nino:~/papers$ cd /tmp/prueba/
pepe@nino:/tmp/prueba$ tar -x -v -z -f cosas.tar.gz
english-lit
history-final
master-tesis
notes/
pepe@nino:/tmp/prueba$ ls
cosas.tar.gz  english-lit  history-final  master-tesis  notes
pepe@nino:/tmp/prueba$
```

find

El comando find busca archivos por el árbol de subdirectorios de forma recursiva. Esto es, si le decimos que busque en el directorio raiz (/) buscara en toda la estructura de carpetas, pero si le decimos que busque en /home/pepe, buscará solamente en el home de pepe. Para ello se pueden utilizar los comodines que vimos anteriormente. Los parámentros de find son la carpeta donde queremos buscar y "-name nombre_archivo". Se puede observar más claramete en el ejemplo...

```
pepe@nino:~$ find . -name *tesis*
./papers/master-tesis
pepe@nino:~$ find /usr/bin -name k*
/usr/bin/kbd_mode
/usr/bin/killall
/usr/bin/kpsewhich
/usr/bin/kpsestat
/usr/bin/kpsetool
/usr/bin/kpsepath
/usr/bin/kpsexpand
/usr/bin/keytool
pepe@nino:~$
```

Primero se ha pedido una búsqueda de algún fichero que contenga la cadena "tesis", en el directorio actual (.). Nos dice que existe el archivo "master-tesis" dentro del directorio "tesis". Después se pide que busque todos los archivos (o directorios) que comiencen por la letra "k" en el directorio /usr/bin.

grep

Grep es un comando capaz de buscar expresiones regulares en ficheros o la entrada estandar. Por ejemplo, imagine un archivo con una relación nombre – teléfono por cada línea. Si quisiése buscar un teléfono en particular, al hacer un "cat" del fichero, le saldrían todos por pantalla. Utilizando tuberías y "grep", esto sería muy sencillo...

```
pepe@nino:~$ cat agenda
Manolo 123421
Raul 124554
Amancio 988832
Pedrito 3246546
[... etc ...]
pepe@nino:~$ cat agenda | grep Roberto
Roberto 2342342
pepe@nino:~$ grep Roberto agenda
Roberto 2342342
pepe@nino:~$ grep berto agenda
Alberto 3421333
Roberto 2342342
pepe@nino:~$
```

Como podemos observar, el primer argumento es la cadena a buscar, y el segundo es el archivo donde buscar. Si éste último se omite, buscará en la entrada estandar.

Como opciones a destacar, tendríamos "-i", para que no distinga entre mayusculas o minúsculas, y "-r", para hacer la búsqueda recursiva.

Ejercicio: *poner un ejercicio.*

hostname

Comando sencillo para conocer el nombre del sistema:

```
pepe@nino:~$ hostname
nino
pepe@nino:~$
```

uptime

Devuelve el tiempo que lleva el sistema activo (levantado, up), así como algunas estadísticas de la carga del mismo, usuarios conectados, etc.

```
pepe@nino:~$ uptime
 21:57:54 up 4 days,  3:46,  3 users,  load average: 1.00, 1.00, 1.00
pepe@nino:~$
```

su

El comando su permite cambiar de usuario. Lleva como argumento el usuario al que queremos cambiar. Sin argumentos, sobreentiende que queremos pasar a la cuenta de administrador (root). Para volver al usuario anterior, solamente tenemos que usar "exit".

```
pepe@nino:~/papers$ cd ..
pepe@nino:~$ su
Password:
[root@nino]:/home/pepe/>
```

clear

Limpia el contenido de la pantalla. Similar al "cls" de MS-DOS.

lpr y lpq

El comando lpr sirve para mandar archivos a la cola de la impresora, para ser impresos; el comando lpq muestra el estado actual de la cola.

date

Muestra la hora y la fecha del sistema.

```
pepe@nino:~$ date
dom nov 3 22:00:46 CET 2002
pepe@nino:~$
```

1.3. Editores de ficheros de texto

Un editor de texto es un programa usado para editar (es decir, crear o modificar) archivos que contienen texto, tales como una carta, un programa en Pascal, o un archivo de configuración del sistema.

Existen muchos editores disponibles para Unix, pero el único que estará seguro de encontrar en cualquier sistema Unix es vi : el "editor visual". vi no es el editor más fácil de usar, porque no es muy amigable. Sin embargo, debido a que es tan común en el mundo Unix y es muy poderoso, téngase por seguro que en alguna oportunidad deberá vérselas con él.

Por otra parte, hay otros editores mucho más amigables que el vi. Sin embargo, en este curso se ofrece una noción del primero, por estar disponible en la mayor parte de los sistemas, y ser, quizá el que presente mayores dificultades de uso.

1.3.1. vi

Mientras use vi, en cualquier momento se encontrará en uno de los tres modos de operación. Estos tres modos son conocidos como modo de comandos, modo de inserción, y modo de última línea.

Cuando empieza vi, te encuentras en el modo de comandos. Este modo permite usar ciertos comandos para editar archivos o cambiar a los otros modos. Por ejemplo, escribiendo "x" mientras esté en el modo de comandos borrará el caracter bajo el cursor. Las teclas de flechas mueven el cursor a través del archivo que estás editando. Generalmente, los comandos usados en el modo de comandos son de uno o dos caracteres de largo, y no es necesario presionar [Enter] después de darlos, pues su efecto es inmediato.

Para insertar o editar texto debe estar en el modo de inserción. Cuando use vi, probablemente estará la mayor parte del tiempo en este modo. Puede partir el modo inserción usando un comando como "i" (por "insert") desde el modo de comandos. Mientras esté en el modo de inserción, estará insertando texto en el documento en el punto del cursor. Para terminar el modo de inserción y volver al modo de comandos, presione Esc.

El modo de última línea es un modo especial usado para dar ciertos comandos extendidos a vi. Mientras teclea estos comandos, ellos aparecen en la última línea de la pantalla (de ahí su nombre), y para ejecutarlos debes terminar presionando Enter. Por ejemplo, cuando teclea ":" desde el modo de comandos, salta dentro del modo de última línea, y puede usar comandos como "wq" para escribir el archivo y salir de vi, o "q!" para salir de vi sin grabar los cambios. El modo de última línea es usado generalmente para comandos de vi que son más largos de un caracter.

Comenzando con vi

La mejor forma de entender estos conceptos es simplemente usando vi para editar un archivo de prueba. En el ejemplo que veremos, la pantalla será solo de 4 líneas, en vez de las 25 usuales.

La sintaxis para vi es: "vi nombre_de_archivo", donde nombre_de_archivo es el nombre del archivo que quieres editar.

```
pepe@nino:~/papers$ vi prueba
```

lo cual editará el archivo prueba. Debería ver algo como esto...

```
~
~
~
"prueba" [New file]
```

La columna de caracteres ~ indica que está al final del archivo.

Inserando texto

Ahora nos encontramos en el modo de comandos; para insertar texto dentro del archivo, presiona [i] (lo cual nos pondrá en el modo de edición) y comienza a escribir.

```
Hola, estoy escribiendo con vi
~                                 _
~
"prueba" [New file]
```

Mientras esté insertando texto, puede escribir cuantas líneas quiera (presionando Enter después de cada una, por supuesto), y puede corregir los errores usando la tecla Backspace. Téngase en cuenta que al borrar de esta manera, los caracteres se mantienen en pantalla hasta que son reescritos.

Para finalizar el modo de edición, y volver al modo de comandos, presione Esc.

Mientras esté en modo de comandos, puede usar las teclas de flechas para moverte por todo el archivo. Aquí, como tenemos solo una línea de texto, al tratar de usar las flechas para subir y para bajar vi emitirá un pitido.

Existen varias otras formas para insertar texto además del comando i. Por ejemplo, el comando a inserta texto inmediatamente después de la posición actual del cursor. Por ejemplo, use la flecha izquierda para mover el cursor sobre la "n" de "con".

```
Hola, estoy escribiendo con vi
~
~
"prueba" [New file]
```

Presione "a" para comenzar el modo de edición, tipea "tra", y presione Esc para volver al modo de comandos.

```
Hola, estoy escribiendo contra vi
~
~
"prueba" [New file]
```

Para empezar a insertar texto en la linea debajo de la actual, use el comando "o". Por ejemplo, presione "o" y escriba una o más lineas:

```
Hola, estoy escribiendo contra vi
pero creo que ya aprendi a usarlo
~
~
"prueba" [New file]
```

Tan solo recuerde que en cualquier momento está "o" en el modo de comandos (donde comandos tales como i, a, o, son válidos), o en modo de edición (donde se inserta texto, y se presiona Esc para retornar al modo de comandos).

Borrando texto

Desde el modo de comandos, el comando "x" borra el caracter bajo el cursor. Si presiona "x" cuatro veces, terminará con...

```
Hola, estoy escribiendo contra vi
pero creo que ya aprendi a us
~
~
"prueba" [New file]
```

Puede borrar lineas enteras usando el comando "dd" (es decir, presionando "d" dos veces). Si el cursor está en la segunda linea, y presiona dd..

```
Hola, estoy escribiendo contra vi
~
~
"prueba" [New file]
```

Para borrar la palabra en la cual se encuentra el cursor, se usa el comando "dw". Ponga el cursor al comienzo de la palabra "contra", y teclee "dw"...

```
Hola, estoy escribiendo vi
~
~
"prueba" [New file]
```

Grabando archivos y cerrando el vi

Para salir de vi sin hacer cambios en el archivo (manteniendo la copia original), se usa el comando ":q!" . Cuando teclea el ":", el cursor se mueve a la ultima linea de la pantalla; está entonces en el modo de última linea.

```
Hola, estoy escribiendo vi
~
~
:
```

En el modo de última linea, hay disponibles ciertos comandos extendidos. Uno de ellos es “q!”, que sale de vi sin grabar. El comando “wq” (también desde la última línea) graba el archivo y sale. El comando “ZZ” (desde el modo de comandos, sin los “:”) es equivalente a “:wq”.

Para grabar el archivo sin salir de vi, use solo “:w”...

```
Hola, estoy escribiendo vi
~
~
"prueba" [New file] 1 line, 38 characters
```

Incluyendo otros archivos

Usando el comando “:r” incluiye el contenido de otro archivo en el actual. Por ejemplo, el comando

```
:r tarea.txt
```

insertará el contenido del archivo tarea.txt en la posición actual del cursor en el texto.

1.3.2. Otros editores

En Linux hay una gran variedad de editores de texto. Se recomienda probar el mayor número de ellos posible, puesto que cada usuario tiene distintas experiencias con otros sistemas operativos y puede estar más habituado a unos tipos que a otros.

- vim

  Es el sucesor del vi. Muy potente.

- gvim

  Versión para las X de vim.

- emacs

  Uno de los más utilizados, porque tiene un gran soporte para la programación de nuevos programas. Potentísimo.

- joe

  El competidor del pico. Es bastante similar, pero cambian los comandos.

- fte

  Otro editor muy utilizado para programar. Es menos potente que emacs, pero mucho mas amigable, y existen versiones para las X, para consolas virtuales, etc.

- pico

  Editor de texto en modo consola, muy sencillo de utilizar. Las operaciones básicas de edición (copiar y pegar, buscar, sustituir...) se realizan con combinaciones de teclas del tipo Ctrl+"tecla". Éstas combinaciones se muestran continuamente en la parte inferior de la pantalla. Comentar que existe un clon del pico llamado "jpico" que suele encontrarse con más facilidad en las distribuciones de linux que el propio pico.

- nano

  Un clon exacto del pico. Es la alternativa cuando el editor habitual es el pico pero la máquina utilizada no lo tiene instalado.

- kedit y gedit

  Los editores de KDE y GNOME, dos entornos gráficos de los que hablaremos más adelante. Por supuesto, son para las X. De un estilo al "Block de notas" de MS-Windows.

1.4. Clientes de correo electrónico (e-mail)

En un sistema UNIX/Linux, el correo no sólo es externo (para el resto de Internet), sino que también es interno (para usuarios registrados en la máquina).

En las carpetas "home" de los usuarios, existe por defecto una subcarpeta llamada "mail", donde se almacenan los correos entrantes (INBOX) y salientes(OUTBOX).

Al igual que existen infinidad de editores de texto, también existen diferentes programas clientes de correo. Dos de los clientes más ampliamente empleados son el mail y el pine. El pine proporciona una interfaz de usuario para el manejo del correo sencilla e intuitiva. En su contra, el cliente "mail" es algo menos autoexplicativo, y por ello se describe a continuación una breve introducción.

1.4.1. mail

Éste es el cliente de correo más básico y estándar que se encuentra en todas las máquinas UNIX. El comando "mail", sin argumentos, comprobará si hay correo entrante...

```
pepe@nino:~$ mail
No mail for pepe
pepe@nino:~$
```

En el caso de que hubiese correo, saldrán enumerados, y sólo tendremos que teclear el número del mensaje que queramos ver...

```
pepe@nino:~$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/pepe": 1 message 1 new
>N  1 peer@nino  Sun Nov 03 16:16   15/481   hola Pepe
& 1
Message 1:
From peer@nino Sun Nov 03 16:16:42 2002
Envelope-to: pepe@nino
To: pepe@nino
Subject: hola Pepe
From: Da Peer <peer@nino>
Date: Sun, 03 Nov 2002 16:16:41 +0100

Como te va la vida? Me alegro.
Saludos

&
```

El símbolo "&" es el prompt del programa mail. Si queremos guardar el mesaje, utilizaremos "s 1", por ser el primer mensaje que queremos guardar. Si en vez de eso, queremos eliminarlo, sería "d 1". Acuda a las páginas del manual para mas información.

Para enviar un correo, llamaremos al programa mail con un argumento, que será la dirección de correo del destinatario. Si éste es un usuario de la máquina, bastará con escribir su nombre de usuario...

```
pepe@nino:~$ mail marcos
Subject: La cena de manhana
Quedamos manhana en el restante a las 22:00.
Un saludo.
.
Cc:
pepe@nino:~$
```

Primero, se nos invitará a introducir el "subject" (asunto) del mensaje. Despues escribiremos el cuerpo del correo. Para terminar de escribirlo, sólo es necesario comenzar una nueva línea, introducir un punto (".") y pulsar "Intro". Por último, nos será solicitado la "Carbon Copy" (Cc), por si queremos enviar el mail otras direcciones.

1.5. Sistema gráfico (XWindow)

El Sistema de Ventanas X es un metodo de trabajo grafico y distribuido, desarrollado principalmente en el Instituto Tecnologico de Massachusetts. Actualmente esta a cargo de un consorcio de fabricantes (debidamente llamado "El Consorcio X") y es mantenido por ellos.

El Sistema de Ventanas X (que a partir de ahora abreviaremos como "X" tiene revisiones cada pocos años, conocidas como lanzamientos. El numero indica la version oficial pero no ha habido cambios en los ultimos años y tampoco hay planes para cambiarla en un futuro proximo. La versión actual es la 11.

Internamente, el sistema funciona con una arquitectura cliente-servidor. El servidor (en Linux habitualmente xfree86) se comunica con las aplicaciones clientes y las muestra por pantalla, segun las solicitudes de éstas. Al ser cliente y servidores programas diferentes, es posible ejecutar cada uno en en maquinas completamente diferentes. Ademas de constituir un metodo estandar para aplicaciones graficas, es posible ejecutar un programa en una maquina remota (¡incluso al otro lado del pais, si quiere!) y que los resultados aparezcan en la estacion de trabajo que tiene enfrente suyo. Un tercer concepto con el que debe familiarizarse es el de gestor de ventanas. El gestor de ventanas es un cliente especial que le dice al servidor en que posicion deben colocarse las diferentes ventanas y permite al usuario moverlas. El servidor, por si mismo, no interacciona con el usuario. Se trata de un medio que conecta el usuario y el cliente.

La mayor parte de las ventajas que ofrece el modo gráfico consiste en convinar varias ventanas simultáneas: correo, navegador, juegos, compilador, depurador, editor, shell... Todo está disponible a la vista, para su mejor combinación.

1.5.1. Configuración y arranque

No es el cometido de este curso explicar la configuración del sistema gráfico, pues merecería otro curso completo. La mayor parte de las distribuciones actuales de linux instalan y configuran perfectamente las "X". Aún así existen varios programas para hacer la configuración mucho mas sencilla, como son el XF86Setup, el XF86Config o el XConfigurator.

De todas formas, remarcar que el archivo de configuración de las "X" se encuentra en /etc/X11, y se llama XF86Config. Aún no siendo demasiado intuitivo este fichero de configuración, se puede entender sin mucho problema, aunque se recomienda no editarlo manualmente, sino permitir a los programas anteriores que lo hagan.

El sistema de ventanas se suele arrancar con el comando "startx".

1.5.2. Gestores de ventanas

Un Administrador de Ventanas es un programa que gestiona las estructuras creadas por medio del X11 y provee al usuario de una interfaz amigable (GUI), que en la mayoría de las ocasiones es altamente configurable.

El manejador de ventanas es el que se encarga de hacer los menús, usar un fondo de pantalla, poner barras de títulos y bordes en las ventanas, permite cambiarles el tamaño, moverlas, y todas estas funciones con las cuales estamos tan familiarizados.

Existe una gran cantidad de manejadores de ventanas, con elementos comunes y cada uno ofreciendo características propias. Los hay mejores que otros, más rápidos, más configurables, más fáciles... en fin. El manejador de ventanas que vayamos a utilizar depende única y exclusivamente de nuestros gustos personales. Los más ampliamente usados y conocidos son Kde, Gnome, AfterStep, Xfce, BlackBox, Enlightenment, IceWM, Sawfish y Window Maker, entre otros.

Un Administrador de Ventanas ofrece las funciones que necesitan en un entorno gráfico. Pero los administradores de ventanas no cuentan con otras características comunes, a las cuales el usuario puede estar acostumbrado, como es un entorno de escritorio con todo lo que esto implica: íconos, manejo gráfico de archivos, drag'n'drop, copiar y pegar, eventos de sonido, etc. Es por esto que se han creado los Ambientes de Escritorio, que funciona en conjunto con un adminstador de de ventanas (ejemplo: pieles de ventanas). Y generalmente el manejador de ventanas debe ser compatible con el ambiente de escritorio en cuestión.

Los ambientes de escritorio más comunes y ampliamente usados son Gnome y KDE, asi como XFCE, BlackBox.

Mucha gente prefiere utilizar los administrador de ventanas, solo prefiere administrar su sistema en modo consola, pero, se recomienda tener al menos uno de ellos instalado, puesto que una gran cantidad de programas requieren para funcionar las librerías de estos.

Gestores mas usados

- GNOME (GNU Network Object Model Enviroment)

  Es un entorno de escritorio gráfico muy potente, y muy amigable, fácil de usar y de configurar, incluye un panel para iniciar las aplicaciones, un escritorio donde se sitúan un conjunto de herramientas estándares y varias aplicaciones, los usuarios de otros sistemas operativos o entornos pueden sentirse a gusto usando el potente entorno de manejo gráfico que provee.

- KDE (K Desktop Enviroment)

  El gran competidor de KDE. Es un entorno de escritorio gráfico muy potente. Es muy amigable para el usuario y una muy buena opción tanto para aquellos que recién empiezan a utilizar una computadora como para los usuarios a vanzados, combina facilidad de uso, funcionalidad actual y un diseño gráfico sobresaliente junto con la superioridad tecnológica del sistema operativo Linux.

- Window Maker

  Es parte del proyecto GNUstep y es una implementación libre, específica del OpenStep(tm). Window Maker es un entorno de escritorio y manejador de ventanas muy versátil, veloz y muy estable. Consta de barras de botones, ademas posee una herramienta de configuración con muchas características. Es una buena opción para aquellos que necesitan velocidad.

- After Step

  Es un manejador de ventanas que fue creado para emular la apariencia y sentido de NeXTSTEP(tm), pero tiene incorporadas características que pueden hacer mucho más. El no utilizar mucha memoria hace de AfterStep un manejador de ventanas rápido y muy cómodo.

- Enlightenment

  Es un manejador de ventanas que a pesar que aún está en desarrollo es muy estable, tiene una apariencia gráfica extraordinaria y muy original, posee muchos apliques y un menú bien balanceado. Otra de sus mejores características es su extrema configurabilidad y efectos que van más allá de lo común. Generalmente administra las ventanas de GNOME pero puede usarse por separado.

- Sawfish

  Es un manejador de ventanas muy versátil y extremadamente configurable, con una interface de apariencia sencilla, es bastante rápido, dinámico y estable, además posee temas para configurar sus tipos de ventanas, colores o degrades de la barra de título. Generalmente administra las ventanas de GNOME pero puede usarse por separado.

- FVWM (FV Window Manager)

  Es un manejador de ventanas de multiples escritorios virtuales originalmente derivado de TWM. FVWM tiende a tener una huella de memoria más pequeña pero un conjunto de muchas características, es muy configurable y extensible, y tiene un alto grado de compatibilidad con Motif mwm.

- FV Window Manager 95

  Es una versión en entorno de escritorio del manejador de ventanas FVWM pero con la apariencia de Windows 95, los agregados son la barra que contiene un botón para iniciar aplicaciones y minimizarlas o cerrarlas,también posee otra con las aplicaciones más comunes como Netscape, la consola para X etc...

- Ice Window Manager

  Es un manejador de ventanas y entorno de escritorio que provee una interface pequeña, rápida y familiar que trabaja con el sistema de ventanas X11. Originalmente fue diseñado para emular la apariencia de Motif, OS/2 Warp 4, OS/2 Warp 3 y Windows 95. Ademas tiene muchos temas de escritório donde es posible elegir otros estilos.

- QV Window Manager

  Es un entorno de escritorio que tiene por finalidad emular la apariencia de Windows 95 y lo hace muy bien es un clónico hasta en sus "accesos directos". QVWM es muy rápido y versátil, ademas posee el paginador de escritorios virtuales. Es una excelente opción para la migración.

- Blackbox

  Es un manejador de ventanas con un entorno de escritorio muy sencillo y bastante rápido, ideal para máquinas con escasos recursos como Intel 386, 486 soporta temas de colores de menús y de escritorio. Blackbox posee un diseño muy atractivo y original.

- XFce

  Es un entorno de escritorio simple, rápido y eficiente para sistemas Linux y otros sistemas basados en Unix. Es fácil de usar y configurar, estable, rápido, y de apariencia agradable. La ventaja que tiene XFce es que mantiene la mayoría de los recursos para las aplicaciones y no consume toda la memoria ni el uso del procesador. De versión en versión, XFce llegará a ser más y más configurable por el usuario.

Capítulo 2

Administración

Este capitulo es una visión general de la administración de un sistema Linux, incluyendo un número de posibilidades avanzadas que no son, necesariamente, solo para administradores de sistemas. Cada sistema tiene su administrador, y poner en marcha el sistema es un trabajo muy importante y a veces consume mucho tiempo, incluso si se es el único usuario en el sistema.

Hemos intentado cubrir aquí los elementos más importantes acerca de la administración de sistemas que se necesitan conocer cuando se use Linux, en suficiente detalle para empezar confortablemente. Para mantenerlo corto y agradable, solo hemos cubierto los niveles más básicos y nos hemos saltado muchos e importantes detalles. Se debe leer el Linux System Administrator's Guide si se quiere ejecutar Linux en serio. Le ayudara a comprender mejor como funcionan las cosas y como se ensamblan juntas.

Como sabe, UNIX distingue entre diferentes usuarios para que lo que hagan a los demás y al sistema pueda ser regulado (uno no desearía que nadie pudiese leer nuestras cartas de amor, por ejemplo).

Cada usuario recibe una cuenta que incluye un nombre de usuario, un directorio inicial, y otras cosas por el estilo. Además de las cuentas dadas a personas reales, existen cuentas especiales, definidas por el sistema, que tienen privilegios especiales. La más importante de estas es la cuenta raíz, con el nombre de usuario root.

2.1. La cuenta root

Los usuarios normales están restringidos normalmente para que no puedan dañar a nadie mas en el sistema, solo a ellos mismos. Los permisos de los ficheros en el sistema están preparados para que los usuarios normales no tengan permitido borrar o modificar ficheros en directorios compartidos por todos los usuarios (como son /bin y /usr/bin). Muchos usuarios también protegen sus propios ficheros con los permisos adecuados para que otros usuarios no puedan acceder o modificar estos ficheros.

Estas restricciones desaparecen para root. El usuario root puede leer, modificar o borrar cualquier fichero en el sistema, cambiar permisos y pertenencias en cualquier fichero, y ejecutar programas especiales, como pueden ser los que particionan un disco o crean sistemas de ficheros. La idea básica es que la persona o personas que ejecutan y cuidan del sistema entren como root cuando sea necesario para realizar tareas que no pueden ser ejecutadas por un usuario normal. Puesto que root puede hacer todo, es fácil cometer errores que tengan consecuencias catastróficas cuando se trabaja utilizando esta cuenta.

Por ejemplo, como un usuario normal, si inadvertidamente se intentase borrar todos los ficheros en /etc, el sistema no lo permitiría. Sin embargo, como usuario root, el sistema no diría nada. Es muy simple el dañar el sistema utilizando root. La mejor forma de evitar accidentes es:

- Pensárselo dos veces antes de apretar Return en un comando que pueda causar daño. Por ejemplo, si se va a borrar un directorio, antes de pulsar Return, releer el comando completo y asegurarse que es correcto.
- No acostumbrarse a utilizar root. Cuanto más confortable se encuentre uno trabajando con el usuario root, más seguro que se confundirán los privilegios con los de un usuario normal.
- Conectarse como root solo cuando sea absolutamente necesario. Y desconectarse tan pronto como se haya terminado el trabajo. Cuanto menos se use la cuenta root, menos posibilidades habrá de dañar el sistema.

2.2. Arranque del sistema

Hay varias maneras de arrancar el sistema, bien sea desde disquete o bien desde el disco duro.

2.2.1. Utilizando un disquete de arranque

Mucha gente arranca Linux utilizando un "disquete de arranque" que contiene una copia del núcleo de Linux. Este núcleo tiene la partición raíz de Linux codificada en el, para que sepa donde buscar en el disco duro el sistema de ficheros raíz. Por ejemplo, este es el tipo de disquete creado por la distribución Slackware durante la instalación.

Para crear su propio disquete de arranque, localice en primer lugar la imagen del núcleo en su disco duro. Debe estar en el fichero /vmlinux o /boot/vmlinux. Algunas instalaciones utilizan el fichero /Image para el núcleo.

En su lugar, puede que tenga un núcleo comprimido. Un núcleo comprimido se descomprime a si mismo en memoria en tiempo de arranque, y utiliza mucho menos espacio en el disco duro. Si se tiene un núcleo comprimido, puede encontrarse en el fichero /vmlinuz o /boot/vmlinuz. Algunas instalaciones utilizan el fichero /zImage para el núcleo comprimido.

Una vez que se sabe donde esta el núcleo, hay que poner el nombre de la partición raíz de un dispositivo raíz en la imagen del núcleo, utilizando el comando rdev. El formato de este comando es

```
rdev <nombre-de-nucleo> <dispositivo-raiz>
```

donde nombre-del-nucleo es el nombre de la imagen del núcleo, y dispositivo-raiz es el nombre de la partición raíz de Linux. Por ejemplo, para hacer que el dispositivo raíz en el núcleo /etc/zImage sea /dev/hda2...

```
[root@nino]~/> rdev /etc/zImage /dev/hda2
```

rdev también puede poner otras opciones en el núcleo, como puede ser el modo SVGA por defecto a utilizar en tiempo de arranque. Tan solo utilice "rdev -h" para obtener un mensaje de ayuda. Una vez puesto el dispositivo raíz, tan solo hay que copiar la imagen del núcleo al disquete. Por ejemplo, para copiar el núcleo en el fichero /etc/Image al disquete en /dev/fd0, se puede utilizar el comando...

```
[root@nino]:~/> dd if=/etc/zImage of=/dev/fd0
```

De todas formas, junto con los cargadores habituales de linux, como "lilo" (siguiente apartado), suele venir una aplicación para crear discos de arranque de una manera mucho más sencilla e intuitiva, cuyo nombre es "mkboot". Invocando esta utilidad, simplemente pedirá insertar un disco y ella se ocupará del resto, preguntando qué debe hacer si encuentra alguna ambigüedad.

2.2.2. LILO (LInux LOader)

Otro método de arranque es utilizar LILO, un programa que reside en el sector de arranque del disco duro. Este programa se ejecuta cuando el sistema se inicia desde el disco duro, y puede arrancar automáticamente Linux desde una imagen de núcleo almacenada en el propio disco duro.

LILO puede utilizarse también como una primera etapa de carga de varios sistemas operativos, permitiendo seleccionar en tiempo de arranque que sistema operativo (como Linux o MS-DOS) arrancar. Cuando se arranca utilizando LILO, se inicia el sistema operativo por defecto, a menos que pulse Ctrl, Alt o Shift durante la secuencia de arranque. Si se pulsa cualquiera de estas teclas, se le presentara un indicador de arranque, donde debe teclear el nombre del sistema operativo a arrancar (como puede ser "linux" o "msdos"). Si se pulsa la tecla Tab en el indicador de arranque, se le presentara una lista de los sistemas operativos disponibles.

La forma mas simple de instalar LILO es editar el fichero de configuración, /etc/lilo.conf, y ejecutar el comando

```
/sbin/lilo
```

El fichero de configuración de LILO contiene una "estrofa" para cada sistema operativo que se pueda querer arrancar. La mejor forma de mostrarlo es con un ejemplo de un fichero de configuración LILO. El ejemplo siguiente es para un sistema que tiene una partición raíz Linux en /dev/hda1 y una partición MS-DOS en /dev/hda2. (En cada línea, cualquier cadena de texto que siga a caracter "#" no es considerada por LILO).

```
# Le indicamos a LILO que modifique el registro de arranque de
# /dev/hda (el primer disco duro no-SCSI). Si se quiere arrancar desde
# una unidad distinta de /dev/hda, se debe cambiar la siguiente linea

boot = /dev/hda


# Nombre del cargador de arranque. No hay razon para cambiarlo, a menos
# que se este haciendo una modificacion seria del LILO

install = /boot/boot.b


# Dejemos a LILO efectuar alguna optimizacion.

compact


# Estrofa para la particion raiz de Linux en /dev/hda1.

image = /etc/Image # Ubicacion del kernel

label = linux # Nombre del SO (para el menu de arranque de LILO)

root = /dev/hda1 # Ubicación de la partición raiz

vga = ask # Indicar al nucleo que pregunte por modos SVGA
          # en tiempo de arranque


# Estrofa para la partición MSDOS en /dev/hda2.

other = /dev/hda2 # Ubicacion de la particion

table = /dev/hda # Ubicacion de la tabla de particion para /dev/hda2

label = msdos # Nombre del SO (para el menu de arranque)
```

La primera “estrofa” de sistema operativo en el menú del fichero de configuración será el sistema operativo que arrancara LILO por defecto. Se puede seleccionar otro sistema operativo en el indicador de arranque de LILO, tal y como se indico anteriormente.

Recuerde que cada vez que actualice la imagen del núcleo en disco, se debe reejecutar /sbin/lilo para que los cambios queden reflejados en el sector de arranque de su unidad.

También tenga en cuenta que si utiliza la línea “root =”, no hay motivo para utilizar rdev para poner la partición raíz en la imagen del núcleo. LILO se encarga de ponerlo en tiempo de arranque.

2.3. Cerrando el sistema

Cerrar un sistema Linux tiene algo de truco. Recuerde que nunca se debe cortar la corriente o pulsar el botón de reset mientras el sistema este ejecutándose. El núcleo sigue la pista de la entrada/salida a disco en buffers de memoria. Si se reinicializa el sistema sin darle al núcleo la oportunidad de escribir sus buffers a disco, puede corromper sus sistemas de ficheros.

En tiempo de cierre se toman también otras precauciones. Todos los procesos reciben una señal que les permite morir airosamente (escribiendo y cerrando todos los ficheros y ese tipo de cosas).

Los sistemas de ficheros se desmontan por seguridad. Si se desea, el sistema también puede alertar a los usuarios de que se esta cerrando y darles la posibilidad de desconectarse.

La forma mas simple de cerrar el sistema es con el comando shutdown. El formato del comando es

```
shutdown <tiempo> <mensaje-de-aviso>
```

El argumento “tiempo” es el momento de cierre del sistema (en el formato hh:mm:ss), y “mensaje-de-aviso” es un mensaje mostrado en todos los terminales de usuario antes de cerrar. Alternativamente, se puede especificar el parámetro tiempo como “now”, para cerrar inmediatamente. Se le puede suministrar la opción -r a shutdown para reinicializar el sistema tras el cierre.

Por ejemplo, para cerrar el sistema a las 8:00pm...

```
[root@nino]:~/> shutdown -r 20:00
```

El comando halt puede utilizarse para forzar un cierre inmediato, sin ningún mensaje de aviso ni periodo de gracia. halt se utiliza si se es el único usuario del sistema y se quiere cerrar el sistema y apagarlo.No apague o reinicialice el sistema hasta que vea el mensaje:

```
The system is halted
```

Es muy importante que cierre el sistema "limpiamente" utilizando el comando shutdown o el halt.

En algunos sistemas, se reconocerá el pulsar Ctrl+Alt+Del, que causara un shutdown; en otros sistemas, sin embargo, el utilizar esta secuencia reinicializará el sistema inmediatamente y puede causar un desastre.

2.4. Gestión de usuarios y grupos

2.4.1. Gestión de usuarios

El sistema mantiene una cierta cantidad de información acerca de cada usuario. Dicha información se resume a continuación:

- Nombre de usuario

  El nombre de usuario es el identificador único dado a cada usuario del sistema. Ejemplos de nombres de usuario son pepe, karl y mdw. Se pueden utilizar letras y dígitos junto a los caracteres "_" (subrayado) y "." (punto). Los nombres de usuario se limitan normalmente a 8 caracteres de longitud.

- User ID

  El user ID, o UID, es un numero único dado a cada usuario del sistema. El sistema normalmente mantiene la pista de la información por UID, no por nombre de usuario.

- Group ID

  El group ID, o GID, es la identificación del grupo del usuario por defecto. En la sección 3.9 discutimos los permisos de grupo; cada usuario pertenece a uno o mas grupos definidos por el administrador del sistema.

- Password

  El sistema también almacena la clave encriptada del usuario. El comando passwd se utiliza para poner y cambiar las claves de los usuarios.

- Nombre completo

  El "nombre real" o "nombre completo" del usuario se almacena junto con el nombre de usuario. Por ejemplo, el usuario schmoj puede tener el nombre "Jos Schmo" en la vida real.

- Directorio inicial

  El directorio inicial es el directorio en el que se coloca inicialmente al usuario en tiempo de conexión. Cada usuario debe tener su propio directorio inicial, normalmente situado bajo /home.

- Intérprete de inicio

  El interprete de inicio del usuario es el interprete de comandos que es arrancado para el usuario en tiempo de conexión. Ejemplos pueden ser /bin/bash y /bin/tcsh.

El fichero /etc/passwd contiene la información anterior acerca de los usuarios. Cada línea del fichero contiene información acerca de un único usuario; el formato de cada línea es

```
nombre:clave encriptada:UID:GID:nombre completo:dir.inicio:interprete
```

Por ejemplo, nuestro usuario Pepe, tiene una linea tal que esta:

```
pepe:x:1021:1021:Pepiño López:/home/pepe:/bin/bash
```

Como puede verse, el primer campo , "pepe", es el nombre de usuario.

El siguiente campo, "x", nos indica que el sistema utiliza "claves en sombra", o "shadow passwords". Esto es que la información de las claves se relega al fichero /etc/shadow. Puesto que /etc/passwd es legible por todo el mundo, /etc/shadow suministra un grado extra de seguridad, puesto que este no lo es. Las claves en sombra suministran algunas otras funciones como puede ser la expiración de claves; no entraremos a detallar estas funciones aquí .

Las claves se encriptan utilizándose a si mismas como clave secreta. En otras palabras, solo si se conoce la clave, esta puede ser desencriptada. Esta forma de encriptación es bastante segura.

El tercer campo "1021", es el UID. Este debe ser único para cada usuario. El cuarto campo, "1021", es el GID. Este usuario pertenece al grupo numerado 100. La información de grupos, como la información de usuarios, se almacena en el fichero /etc/group. Véase la sección 4.4.5 para mas información.

El quinto campo es el nombre completo del usuario, "Pepiño López". Los dos últimos campos son el directorio inicial del usuario (/home/pepe) y el interprete de conexión (/bin/bash), respectivamente. No es necesario que el directorio inicial de un usuario tenga el mismo nombre que el del nombre de usuario. Sin embargo, ayuda a identificar el directorio.

Añadiendo usuarios

Cuando se añade un usuario hay varios pasos a seguir. Primero, se le debe crear una entrada en /etc/passwd, con un nombre de usuario y UID únicos. Se debe especificar el GID, nombre completo y resto de información. Se debe crear el directorio inicial, y poner los permisos en el directorio para que el usuario sea el dueño. Se deben suministrar ficheros de comandos de inicialización en el nuevo directorio y se debe hacer alguna otra configuración del sistema (por ejemplo, preparar un buzón para el correo electrónico entrante para el nuevo usuario).

Aunque no es difícil el añadir usuarios a mano, cuando se esta ejecutando un sistema con muchos usuarios, es fácil el olvidarse de algo. La manera mas simple de añadir usuarios es utilizar un programa interactivo que vaya preguntando por la información necesaria y actualice todos los ficheros del sistema automáticamente. El nombre de este programa es useradd o adduser dependiendo del software que este instalado. Las paginas man para estos comandos deberían ser suficientemente autoexplicatorias.

Por ejemplo, añadamos un nuevo usuario al sistema, llamado “prueba”...

```
[root@nino]:~/> adduser
Enter a username to add: prueba
Adding user prueba...
Adding new group prueba (1023).
Adding new user prueba (1023) with group prueba.
Creating home directory /home/prueba.
Copying files from /etc/skel
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for prueba
Enter the new value, or press return for the default
        Full Name []: Usuario de prueba
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []: Va a ser eliminado pronto
Is the information correct? [y/n] y
[root@nino]:~/>
```

Borrando usuarios

De forma parecida, borrar usuarios puede hacerse con los comandos userdel o deluser dependiendo de que software fuera instalado en el sistema.

```
[root@nino]:/home/pepe/> deluser prueba
Removing user prueba...
done.
[root@nino]:/home/pepe/>
```

Si se desea "deshabilitar" temporalmente un usuario para que no se conecte al sistema (sin borrar la cuenta del usuario), se puede prefijar con un asterisco ("*") el campo de la clave en /etc/passwd.

Por ejemplo, cambiando la línea de /etc/passwd correspondiente a pepe a

```
pepe:*x:1021:1021:Pepiño López:/home/pepe:/bin/bash
```

evitara que pepe se conecte.

2.4.2. Gestión de grupos

Cada usuario pertenece a uno o mas grupos. La única importancia real de las relaciones de grupo es la perteneciente a los permisos de ficheros. Como veremos en la siguiente sección, cada fichero tiene un "grupo propietario" y un conjunto de permisos de grupo que define de que forma pueden acceder al fichero los usuarios del grupo.

Hay varios grupos definidos en el sistema, como pueden ser bin, mail, y sys. Los usuarios no deben pertenecer a ninguno de estos grupos; se utilizan para permisos de ficheros del sistema. En su lugar, los usuarios deben pertenecer a un grupo individual, como users. Si se quiere ser detallista, se pueden mantener varios grupos de usuarios como por ejemplo estudiantes, soporte y facultad.

El fichero /etc/group contiene información acerca de los grupos. El formato de cada línea es

```
nombre de grupo:clave:GID:otros miembros
```

Algunos ejemplos de grupos pueden ser:

```
root:*:0:
usuarios:*:100:pepe,lolo
invitados:*:200:
otros:*:250:prueba
```

El primer grupo, root, es un grupo especial del sistema reservado para la cuenta root. El siguiente grupo, users, es para usuarios normales. Tiene un GID de 100. Los usuarios pepe y lolo tienen acceso a este grupo. Recuérdese que en /etc/passwd cada usuario tiene un GID por defecto. Sin embargo, los usuarios pueden pertenecer a mas de un grupo, añadiendo sus nombres de usuario a otras líneas de grupo en /etc/group. El comando groups lista a que grupos se tiene acceso.

El tercer grupo, invitados, es para usuarios invitados, y otros es para "otros" usuarios. El usuario prueba tiene acceso a este grupo.

Como se puede ver, el campo “clave” de /etc/group raramente se utiliza. A veces se utiliza para dar una clave para acceder a un grupo. Esto es raras veces necesario. Para evitar el que los usuarios cambien a grupos privilegiados (con el comando newgroup), se pone el campo de la clave a “*”.

Se pueden usar los comandos addgroup o groupadd para añadir grupos a su sistema. Normalmente es mas sencillo añadir líneas a /etc/group uno mismo, puesto que no se necesitan mas configuraciones para añadir un grupo. Para borrar un grupo, solo hay que borrar su entrada de /etc/group.

2.5. Sistema de permisos

Al ser UNIX un sistema multiusuario, para proteger ficheros de usuarios particulares de la manipulación por parte de otros, UNIX proporciona un mecanismo conocido como permisos de ficheros.

Este mecanismo permite que ficheros y directorios “pertenezcan” a un usuario en particular. Por ejemplo, como Pepe creó ficheros en su directorio “home”, Pepe es el propietario de esos ficheros y tiene acceso a ellos.

UNIX también permite que los ficheros sean compartidos entre usuarios y grupos de usuarios. Si Pepe lo desea, podría restringir el acceso a sus ficheros de forma que ningún otro usuario tenga acceso. De cualquier modo, en la mayoría de los sistemas por defecto se permite que otros usuarios puedan leer tus ficheros pero no modificarlos o borrarlos.

Como hemos explicado arriba, cada fichero pertenece a un usuario en particular. Por otra parte, los ficheros también pertenecen a un grupo en particular, que es un conjunto de usuarios definido por el sistema. Cada usuario pertenece al menos a un grupo cuando es creado. El administrador del sistema puede hacer que un usuario tenga acceso a mas de un grupo.

Los grupos usualmente son definidos por el tipo de usuarios que acceden a la máquina. Por ejemplo, en un sistema UNIX de una universidad, los usuarios pueden ser divididos en los grupos estudiantes, dirección, profesores e invitados. Hay también unos pocos grupos definidos por el sistema (como bin y admin) los cuales son usados por el propio sistema para controlar el acceso a los recursos, muy raramente los usuarios normales pertenecen a estos grupos.

Los permisos están divididos en tres tipos: lectura, escritura y ejecución. Estos permisos pueden ser fijados para tres clases de usuarios: el propietario del fichero, el grupo al que pertenece el fichero y para todos los usuarios independientemente del grupo.

El permiso de lectura permite a un usuario leer el contenido del fichero o en el caso de un directorio, listar el contenido del mismo (usando ls). El permiso de escritura permite a un usuario escribir y modificar el fichero. Para directorios, el permiso de escritura permite crear nuevos ficheros o borrar ficheros ya existentes en dicho directorio. Por último, el permiso de ejecución permite a un usuario ejecutar el fichero si es un programa o guión del intérprete de comandos. Para directorios, el permiso de ejecución permite al usuario cambiar al directorio en cuestión con cd.

2.5.1. Interpretando los permisos

Veamos un ejemplo del uso de permisos de ficheros. Usando el comando ls con la opción -l se mostrará un listado "largo" de los ficheros, el cual incluye los permisos de ficheros.

```
pepe@nino:~/papers/notes$ ls -l
total 4
-rw-r--r--    1 pepe     users          555 2002-12-07 18:20 carta
pepe@nino:~/papers/notes$
```

El primer campo impreso en el listado representa los permisos de ficheros. El tercer campo es el propietario del fichero (pepe), y el cuarto es el grupo al cual pertenece el fichero (users).

Este fichero pertenece a pepe y al grupo users. Echemos un vistazo a los permisos. La cadena -rw-r–r– nos informa, por orden, de los permisos para el propietario, el grupo del fichero y cualquier otro usuario.

El primer carácter de la cadena de permisos ("-") representa el tipo de fichero. El "-" significa que es un fichero regular. Las siguientes tres letras ("rw-") representan los permisos para el propietario del fichero, pepe. El "r" para "lectura" y "w" para escritura. Luego Pepe tiene permisos de lectura y escritura para el fichero stuff.

Como ya mencionamos, aparte de los permisos de lectura y escritura esta el permiso de ejecución, representado por una "x". Como hay un "-" en lugar del "x", significa que pepe no tiene permiso para ejecutar ese fichero. Esto es correcto, puesto que stuff no es un programa de ningún tipo. Por supuesto, como el fichero es de Pepe, puede darse a si mismo permiso de ejecución si lo desea. Esto será cubierto en breve.

Los siguientes tres caracteres, r– representan los permisos para los miembros del grupo. El grupo al que pertenece el fichero es users. Como solo aparece un "r" cualquier usuario que pertenezca al grupo users puede leer este fichero.

Los últimos tres caracteres, también r–, representan los permisos para cualquier otro usuario del sistema (diferentes del propietario o de los pertenecientes al grupo users). De nuevo, como solo esta presente el "r", los demás usuarios pueden leer el fichero, pero no escribir en el o ejecutarlo.

Aquí tenemos otros ejemplos de permisos:

- - r w x r - x r - x

  El propietario del fichero puede leer, escribir y ejecutar el fichero. Los usuarios pertenecientes al grupo del fichero, y todos los demás usuarios pueden leer y ejecutar el fichero.

- - r w - - - - - - -

  El propietario del fichero puede leer, escribir y ejecutar el fichero. Los usuarios pertenecientes al grupo del fichero, y todos los demás usuarios pueden leer y ejecutar el fichero.

- - r w x r w x r w x

  El propietario del fichero puede leer, escribir y ejecutar el fichero. Los usuarios pertenecientes al grupo del fichero, y todos los demás usuarios pueden leer y ejecutar el fichero.

Ejercicio:

2.5.2. Dependencias de permisos

Es importante darse cuenta de que los permisos de un fichero también dependen de los permisos del directorio en el que residen. Por ejemplo, aunque un fichero tenga los permisos -rwxrwxrwx, otros usuarios no podrán acceder a él a menos que también tengan permiso de lectura y ejecución para el directorio en el cual se encuentra el fichero. Si Pepe quiere restringir el acceso a todos sus ficheros, podría simplemente poner los permisos de su directorio "home" /home/pepe a drwx——. De esta forma ningún usuario podrá acceder a su directorio ni a ninguno de sus ficheros o subdirectorios (la "d" inicial indica que es un directorio).

Pepe no necesita preocuparse de los permisos individuales de cada uno de sus ficheros.

En otras palabras, para acceder a un fichero, debes de tener permiso de ejecución de todos los directorios a lo largo del camino de acceso al fichero, además de permiso de lectura (o ejecución) del fichero en particular.

Habitualmente, los usuarios de un sistema UNIX son muy abiertos con sus ficheros. Los permisos que se dan a los ficheros usualmente son -rw-r–r–, lo que permite a todos los demás usuarios leer los ficheros, pero no modificarlos de ninguna forma. Los directorios, usualmente tienen los permisos drwxr-xr-x, lo que permite que los demás usuarios puedan moverse y ver los directorios, pero sin poder crear o borrar nuevos ficheros en ellos.

Muchos usuarios pueden querer limitar el acceso de otros usuarios a sus ficheros. Poniendo los permisos de un fichero a -rw——- no se permitirá a ningún otro usuario acceder al fichero.

Igualmente, poniendo los permisos del directorio a -rwx—— no se permitirá a los demás usuarios acceder al directorio en cuestión.

2.5.3. Cambiando los permisos

El comando chmod se usa para establecer los permisos de un fichero. Solo el propietario puede cambiar los permisos del fichero. La sintaxis de chmod es:

```
chmod {a,u,g,o}{+,-}{r,w,x} <filenames>
```

Brevemente, indicamos a que usuarios afecta all, user, group o other. Entonces se especifica sise están añadiendo permisos (+) o quitándolos (-). Finalmente se especifica que tipo de permiso read, write o execute. Algunos ejemplos:

- `chmod a+r carta`
  Da a todos los usuarios acceso al fichero.

- `chmod +r carta`
  Como arriba, si no se indica a, u, g u o por defecto se toma a.

- `chmod og-x carta`
  Quita permisos de ejecución a todos los usuarios excepto al propietario.

- `chmod u+rwx carta`
  Permite al propietario leer, escribir y ejecutar el fichero.

- `chmod o-rwx carta`
  Quita permisos de lectura, escritura y ejecución a todos los usuarios menos al propietario y a los usuarios del grupo del fichero.

2.6. Gestión de sistemas de ficheros

Para empezar, algunos conceptos acerca de sistemas de ficheros. Antes de que un sistema de ficheros sea accesible al sistema, debe ser montado en algún directorio. Por ejemplo, si se tiene un sistema de ficheros en un disquete, se debe montar bajo algún directorio, digamos /mnt, para poder acceder a los ficheros que contiene. Tras montar el sistema de ficheros, todos los ficheros en dicho sistema aparecen en ese directorio. Tras desmontar el sistema de ficheros, el directorio (en este caso, /mnt) estará vacío.

Lo mismo es valido para los sistemas de ficheros del disco duro. El sistema monta automáticamente los sistemas de ficheros del disco duro en tiempo de arranque. El así llamado "sistema de ficheros raíz" es montado en el directorio /. Si se tiene un sistema de ficheros separado para /usr, por ejemplo, se monta en /usr. Si solo se tiene un sistema de ficheros raíz, todos los ficheros (incluyendo los de /usr) existen en ese sistema de ficheros.

El comando mount se utiliza para montar un sistema de ficheros. El comando "mount -av" se ejecuta desde el fichero /etc/rc (que es el fichero de inicialización del sistema, ejecutado en tiempo de arranque; hablaremos un poco mas adelante de él). El comando mount -av obtiene información de los sistemas de ficheros y puntos de montaje del fichero /etc/fstab. Este es un ejemplo de fichero fstab:

```
# dispositivo directorio tipo opciones

/dev/hda1 / ext2 defaults 0 1
/dev/hda2 /usr ext2 defaults 0 0
/dev/hdb1 /windows vfat defaults 0 0
/dev/hda4 none swap sw 0 0
/dev/hdc /cdrom iso9660 0 0
```

El primer campo es el dispositivo, el nombre de la partición a montar. El segundo campo es el punto de montaje. El tercero es el tipo de sistema de ficheros, como puede ser ext2 (para ext2fs) o vfat (para sistemas de ficheros FAT32 de MS-Windows). La siguiente tabla lista los distintos tipos de sistemas de ficheros disponibles en Linux. Puede que no todos estos tipos de sistemas de ficheros estén disponibles en su sistema; el núcleo debe tener soporte para ellos compilado en el. El cuarto campo contiene las opciones del comando mount, normalmente, esta puesto a "defaults" (opciones por defecto). De los últimos campos ("dump" y "pass") no hablaremos, solamente decir que deben estar puestos a 0, a excepción del sistema de archivos raíz, que debe tener el dump a 1.

Sistema de ficheros	Nombre del tipo	Comentarios
Second Extended Filesystem	ext2	Sistema de ficheros mas común en Linux
Extended Filesystem	ext	Reemplazado por ext2
Minix Filesystem	minix	Sistema de ficheros Minix original; pocas veces usado.
MS-DOS Filesystem	msdos	Utilizado para acceder a ficheros MS-DOS.
FAT32 Filesystema	vfat	Utilizado para acceder a ficheros de Windows.
NTFS Filesystem	ntfs	Utilizado para acceder a ficheros de Windows 2000 y XP.
/proc Fliesystem	proc	Suministra información de proceso para ps, etc.
ISO 9660 Filesystem	iso9660	Formato utilizado por la mayoría de CD-ROMs.
UDF Filesystem	udf	Formato utilizado en los DVDs.
System V Filesystem	sysv	Variantes del System V para el x86.

El comando mount solo puede ser utilizado por root. Esto es así para garantizar la seguridad del sistema; no es deseable que usuarios normales estén montando y desmontando

sistemas de ficheros a su antojo. Existen varios paquetes disponibles que permiten a los usuarios normales montar y desmontar sistemas de ficheros (disquetes en particular) sin comprometer la seguridad del sistema.

El comando mount -av realmente monta todos los sistemas de ficheros excepto el sistema de ficheros raíz (en la tabla anterior, /dev/hda2). El sistema de ficheros raíz es montado automáticamente en tiempo de arranque por el núcleo.

En vez de utilizar el comando mount -av, se puede montar un sistema de ficheros a mano. El comando

```
mount -t iso9660 /dev/hdc /cdrom
```

es equivalente a montar el sistema de ficheros con la entrada /dev/hda3 del ejemplo de fichero fstab anterior.

En general, nunca se debe montar o desmontar sistemas de ficheros a mano. El comando mount -av en /etc/rc se encarga de montar los sistemas de ficheros en tiempo de arranque. Los sistemas de ficheros son desmontados por los comandos shutdown o halt antes de cerrar el sistema.

Ejercicio: *trate de montar un dispositivo en dos directorios diferentes. Observe lo que ocurre y busque la explicación en las páginas de manual.*

2.6.1. Comprobando los sistemas de ficheros

Normalmente es una buena idea el comprobar de vez en cuando los sistemas de ficheros en busca de ficheros dañados o corrompidos. Algunos sistemas comprueban automáticamente sus sistemas de ficheros en tiempo de arranque (con los comandos apropiados en /etc/rc).

El comando utilizado para comprobar un sistema de ficheros depende del tipo de sistema de ficheros en cuestión. Para sistemas de ficheros ext2fs (el tipo mas utilizado normalmente), el comando es e2fsck. Por ejemplo, el comando

```
e2fsck -av /dev/hda2
```

comprobara el sistema de ficheros ext2fs de /dev/hda2 y corregirá automáticamente cualquier error.

Normalmente es una buena idea el desmontar un sistema de ficheros antes de comprobarlo. Por ejemplo, el comando

```
umount /dev/hda2
```

desmontara el sistema de ficheros en /dev/hda2, tras lo cual podrá ser comprobado. La única excepción es que no se puede desmontar el sistema de ficheros raíz. Para poder comprobar el sistema de ficheros raíz cuando está desmontado, se debe utilizar un disquete de arranque/raíz (véase la sección 4.11.1). Tampoco se puede desmontar un sistema de ficheros si alguno de sus ficheros esta "ocupado", esto es, siendo utilizado por un proceso en ejecución. Por ejemplo, no se puede desmontar un sistema de ficheros si el directorio de trabajo de algún usuario esta en ese sistema de ficheros. Se recibirá un error "Device busy" (dispositivo ocupado) si se intenta desmontar un sistema de ficheros que este en uso.

Otros tipos de sistemas de ficheros utilizan formas diferentes del comando e2fsck, como pueda ser efsck y dosfsck. En algunos sistemas, se puede utilizar el comando fsck, que determina el tipo de sistema de ficheros y ejecuta el comando apropiado.

Es importante que se reinicialice el sistema inmediatamente después de comprobar un sistema de ficheros montado, si es que se hizo alguna corrección al sistema de ficheros. (Sin embargo, en general, no se deben comprobar sistemas de ficheros que estén montados.) Por ejemplo, si e2fsck informa que ha corregido algún error en el sistema de ficheros, se debe apagar el sistema con shutdown -r para rearrancarlo. Esto permite al sistema resincronizar su información acerca del sistema de ficheros cuando e2fsck lo modifica.

2.6.2. Los disquetes y las "mtools"

Las mtools son un conjunto de utilidades que permiten manipular archivos DOS. Normalmente son útiles para manejar disquetes sin necesidad de montar y desmontar dicha unidad. A esta unidad se refiere como se hacía típicamente en el sistema operativo MS-DOS, con "A:". Los comandos que incluye son:

- mcopy <origen> <destino>
  Para copiar archivos, análogo a "cp".
- mcd <directorio>
  Para cambiar de directorio dentro del disquete, análogo a "cd".
- mdir
  Para listar los archivos de un directorio, análogo a "ls".
- mmd <directorio>
  Para crear un directorio, análogo a "mkdir".
- mdel <fichero>
  Para eliminar uno o un conjunto de ficheros, análogo a "rm".
- mdeltree <directorio>
  Para eliminar un conjunto de ficheros y subdirectorios que esten dentro del directorio de destino, análogo a "rm -r".

2.7. Gestión de software

Cada distribución tiene una forma distinta de instalar softwar. Muchas tienen un programa que le guía paso a paso en este proceso. En otras, usted tendrá que montar sus sistemas de ficheros en un directorio (como /tmp) y copiar el software a este a mano. En las distribuciones en CD-ROM puede seguir la opción de instalar una parte de lo que contiene en su disco duro y dejar el resto (la mayor parte) en el CD-ROM.

Algunas distribuciones ofrecen diversos mecanismos para instalar el software. Por ejemplo, puede instalarlo directamente desde una partición MS-DOS de su disco duro, en lugar de hacerlo desde los disquetes. O incluso puede hacerlo a través de una red TCP/IP mediante FTP o NFS. Consulte la documentación de la distribución para ver detalles.

Este curso basará su explicación de gestión de software para sistemas de paquetes RPM (ver más abajo), por ser el más extendido.

2.7.1. Paquetería

En Linux, el software se gestiona por paquetes. Un paquete es normalmente un archivo que contiene una librería, una aplicación, documentación, etc. con todos los ficheros necesarios para que instalarse en el sistema y que funcione perfectamente.

El software para Linux lo podemos encontrar de dos formas: en un paquete binario o en un paquete con su código fuente.

Paquetes binarios

Existen varios tipos de paquetes binarios. Todos tienen una característica en común, y es que contienen código de máquina, no código fuente, por eso cada tipo de procesador necesita su propia versión de cada paquete. Hay varios tipos de paquetes binarios, uno para cada distribución, aunque hay varias distribuciones que comparten sistema de paquetes. Los más comunes son:

- Paquetes RPM

  Los usan las distribuciones RedHat, Caldera, Madrake, SuSE y TurboLinux entre otras. Su uso está muy extendido y es posible instalar este tipo de paquetes mediante la aplicación rpm. El nombre de los paquetes rpm es del tipo nombredelpaquete_version_plataforma.rpm

- Paquetes deb

  Los usa la distribución Debian y sus derivadas. Es un sistema de paquetes muy potente y que facilita en gran medida la actualización del sistema, además de resolver las dependencias (ver más abajo) y satisfacerlas instalando todos los paquetes necesarios automáticamente. La aplicaciones que gestionan este sistema de paquetes se llaman apt y dpkg. Los paquetes Debian se suelen nombrar de la forma nombredelpaquete_version_plataforma.deb

- Paquetes tgz

  Sistema arcaico utilizado en la actualidad solo por la distribuciçon Slackware.

Paquetes de código fuente

Estos paquetes contienen los archivos que salen del ordenador del programador o programadores, lo que quiere decir que ya hay que aportar algo de nuestra parte para que funcionen los programas que contienen. El proceso de instalación de este tipo de paquetes se llama compilación. Una compilación nos permite que el programa que hemos instalado se haya optimizado totalmente para el tipo de componentes que tenemos en nuestro ordenador y el tipo de versión de Linux. Este programa que hemos compilado correrá más rápido que si nos hubiéramos limitado a instalar un paquete binario normal. La compilación de un programa requiere de compiladores, que son unos programas que junto con unas librerías de lenguajes de programación, consiguen transformar el código fuente en lenguaje de máquina.

Son también de amplio uso los Source RPM, que son paquetes RPM pero que en vez de ser binarios, llevan código fuente. Mediante la instalación de este tipo de paquetes, lo que hacemos es crear un nuevo paquete optimizado (compilar un nuevo paquete) para nuestra máquina. Después instalamos este último.

2.7.2. Gestión de paquetes RPM

RPM es un sistema es empaquetamiento que se está convirtiendo en un estándar de hecho en el mundo Linux por las ventajas que supone sobre otros modelos de empaquetamiento y gestión de las instalaciones. Además se distribuye bajo los términos de la GPL (General Public License).

Proporciona al usuario final una serie de facilidades que hacen el mantenimiento del sistema mas sencillo de gestionar ya que mantiene una base de datos de los paquetes instalados y de sus archivos, lo que permite realizar consultas y verificaciones del sistema.

Otra ventaja es que al actualizar software, los archivos de configuración se respetan de manera que no sea necesario volver a realizar los ajustes específicos que ya tuviera definidos, o si no fuera posible, realiza una copia de seguridad de los mismos.

Instalar un paquete

Para instalar un paquete, simplemente debemos usar el comando rpm con la opción -i (de instalación), y el nombre del paquete. Además podemos utilizar la opción -v para el modo "verbose", que saca por pantalla detalles de la instalación.

Imaginemos que tenemos el programa "gcc-3.2-1.i386.rpm". Como ya se ha comentado, "gcc" es el nombre del programa, "3.2-1" es la versión y "i386" es el tipo de máquina para la cual está compilado, en este caso PCs normales. El comando sería (con modo verbose...):

```
rpm -i gcc-3.2-1.i386.rpm
```

Si el paquete ya estuviese instalado en el sistema, el comando rpm nos dará un código de error de la siguiente forma...

```
[root@nino]:~/> rpm -i gcc-3.2-1.i386.rpm
gcc package gcc-3.2-1 is already installed
error: gcc-3.2-1.i386.rpm cannot be installed
```

Para forzar la reinstalación, utilizaremos la opción -replacepkgs.

Desinstalar un paquete

Desinstalar un paquete es tan sencillo como instalarlo, rpm garantiza la limpieza del procedimiento y que se desinstalarán todos los ficheros, no importa donde estén instalados. La opción "-e" desinstalaría un paquete, pero en este caso sólo hay que pasar el nombre del programa:

```
rpm -e gcc
```

Consulta e información de paquetes

Otra de las grandes característica s del sistema rpm es el mantenimiento de una base de datos de todo lo instalado, lo que permite interrogar al sistema para obtener determinada información sobre los paquetes que tenemos instalados, qué ficheros contenían estos paquetes, donde se encuentran ubicados, que tipo de documentación aportaban o cuales son los ficheros de configuración.

Consultar la base de datos de paquetes instalados tampoco es complicado: lo conseguimos con la opción rpm -q...

```
[root@nino]:~/> rpm -q gcc
gcc-3.2-1
```

Para consultar todos los paquetes instalados en el sistema, solo tenemos que añadir la opción -a...

```
rpm -q -a
```

que mostrara una pantalla (sumamente larga) con todos los paquetes.

Para una consulta mucho mas detallada de un paquete, usaremos la opción -i que aporta todo tipo de información sobre el mismo...

```
[root@nino]:~/> rpm -q -i fileutils
Name: fileutils Relocations: (not relocateable)
Version: 4.0 Vendor: Red Hat, Inc.
Release: 21 Build Date: mié 08 mar 2000 02:42:54 CET
Install date: mar 18 abr 2000 18:32:12 CEST Build Host:
porky.devel.redhat.com
Group: Aplicaciones/Ficheros Source RPM: fileutils-4.0-21.src.rpm
Size: 1608922 License: GPL
Packager: Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Summary: Utilidades de gestión de los ficheros realizados por GNU
Description:
El paquete fileutils contiene varias utilidades de gestion
de los ficheros realizados por la GNU. En este paquete
encontrara: chgrp (cambia el grupo de un fichero), chown
(cambia el propietario del grupo) chmod (cambia los permisos
de un fichero), cp (copia los ficheros), dd (copia y convierte
los ficheros), df (visualiza el uso del disco), dir (proporciona
una lista de los ficheros), dircolors (proporciona la coloracion
de los ficheros al comando ls), du (visualiza el uso del disco),
install (copia ficheros y inicializa los permisos), ln (crea
los link), ls (enumera el contenido de un directorio), mkdir
(crea directorios), mkfifo (crea FIFOs, que son pipe), mknod
(crea ficheros especiales), mv (renombra los ficheros), rm
(cancela los ficheros), rmdir (cancela los directorios vacios),
sync (descarga los buffers en el disco), touch (cambia el tiempo
de impresion de los ficheros), y vdir (visualizacion en si del
contenido de un directorio).
```

Por último, si queremos conocer cuales son los ficheros de configuración de ese paquete, utilizamos la opción -c...

```
[root@nino]:~/> rpm -c -l fileutils
/etc/DIR_COLORS
/etc/profile.d/colorls.csh
/etc/profile.d/colorls.sh
```

Actualización de un paquete

Actualizar un paquete como instalar un paquete con la salvedad de que rpm desinstala automáticamente cualquier versión antigua del paquete. En este caso, la opción adecuada será -U.

```
rpm -U -v gcc-3.2-1.i386.rpm
```

Comentamos anteriormente que en la actualización se mantienen los ficheros de configuración. Esto es cierto siempre que sea posible para rpm. Si el formato del archivo ha sufrido

tantos cambios que no es compatible con el antiguo, rpm hace una copia de seguridad del fichero antiguo.

Ejercicio: *poner aqui nu par de ejercicios*

2.7.3. Dependencias de paquetes

En Linux, los programas suelen necesitar librerías y otros programas para poder funcionar correctamente. Por ejemplo, puede existir una librería de visualización de distintos archivos gráficos (jpg, gif, bmp...) y un programa con una interfaz gráfica para visualizarlos. Este programa utilizará esa librería, pero puede que otro programa, de edición de imágenes, por ejemplo, también la utilice. Incluir todas estas librerías en cada programa sería una pérdida absoluta de espacio en disco y de rendimiento. Por ello, los paquetes no son solo de aplicaciones, sino también de librerías.

Si quisiésemos instalar el paquete del programa de visualización de imágenes del que hablábamos antes, pero en nuestro sistema no está instalada la librería, obtendríamos un problema de dependencias.

Para el siguiente ejemplo, imaginemos que poseemos el paquete del programa de visualización "visor-1.0.i386.rpm", que usa la librería "libvisualizacion-2.3.i386.rpm".

```
[root@nino]:~/> rpm -i visor-1.0.i386.rpm
failed dependencies:
libvisualizacion >= 2.1 is needed by visor-1.0.i386.rpm
```

Este error nos indica que visor depende de libvisualización, y además necesita, al menos, la versión 2.1 de esa librería.

Para resolver este tipo de problemas (de dependencias) debemos conseguir todos los paquetes que nos indica los errores que lanza rpm, y además de las versiones concretas que requiere.

Para nuestro ejemplo, bastaría con instalar previamente el paquete libvisualizacion:

```
[root@nino]:~/> rpm -i libvisualizacion-2.3.i386.rpm
[root@nino]:~/> rpm -i visor-1.0.i386.rpm
```

Este problema puede ser "recursivo", en el sentido que quizá un paquete que debamos instalar previamente para resolver un conflicto de dependencias puede depender de otros paquetes, y así sucesivamente.

Existen ya diversos programas que, indicándoles fuentes de paquetes rpm como FTPs o directorios locales, son capaces de instalar software y resolver automáticamente los problemas de dependencias de la forma más eficiente posible. Algunos ejemplos son el xrpm, el GRAB o el apt-rpm, que es un intento de introducir el sistema de paquetes de la distribución Debian en distribuciones basadas en RedHat.

Ejercicio: *Buscar un par de paquetes con dependencias y pedir que se instale*

2.8. Algunas otras tareas

Hay muchas tareas que un administrador debe realizar para mantener el sistema en "buen estado". Desde la instalación de nuevo hardware, con la consiguiente actualización del sistema, a la seguridad de la máquina dentro de una red. Aquí expondremos dos de ejemplo:

2.8.1. Ficheros de arranque

Cuando el sistema arranca, se ejecutan automáticamente una serie de ficheros de comandos (scripts) en el sistema, antes de que ningún usuario entre. Aquí tenemos una descripción de lo que ocurre:

En tiempo de arranque, el núcleo arranca el proceso /etc/init. init es un programa que lee su fichero de configuración, /etc/inittab, y arranca otros procesos basados en el contenido de este fichero. Uno de los procesos mas importantes arrancado desde inittab es el proceso /etc/getty, arrancado en cada consola virtual. El proceso getty dispone la consola virtual para ser utilizada y arranca un proceso login en ella. Esto le permite conectarse a cada consola virtual; si /etc/inittab no contiene un proceso getty para una consola virtual determinada, no se podrá conectar nadie a ella.

Otro proceso ejecutado desde /etc/inittab es /etc/rc, el fichero de inicialización principal del sistema. Este fichero es simplemente un fichero de comandos que ejecuta cualquier comando de inicialización necesario en tiempo de arranque, como es montar los sistemas de ficheros e inicializar el espacio de intercambio (memoria virtual).

Su sistema puede ejecutar otros ficheros de comandos de inicialización también, como puede ser /etc/rc.local. /etc/rc.local contiene normalmente comandos de inicialización específicos de su sistema, como puede ser el establecimiento del nombre del ordenador (véase la siguiente sección).

rc.local puede ser arrancado desde /etc/rc o directamente desde /etc/inittab.

2.8.2. Estableciendo el nombre del equipo

En un entorno de red el nombre del ordenador es utilizado para identificar unívocamente una máquina particular, mientras que en un entorno autónomo, el nombre del ordenador

da a la máquina personalidad y encanto. Es como darle nombre a un animal doméstico: siempre puede dirigirse a su perro como "El perro", pero es mucho mas interesante asignarle al perro un nombre como "Sultán" o "Pisti". Poner el nombre del sistema se limita a utilizar el comando hostname. Si se esta en una red, su nombre debe ser el nombre completo de su máquina, por ejemplo, nino.rnasalab.com. Si no se está en una red de ningún tipo, se pueden escoger nombre de ordenador y de dominio arbitrarios, como por ejemplo loomer.vpizza.com o caspita.org.

Cuando se pone el nombre del ordenador, dicho nombre debe aparecer en el fichero /etc/hosts, que asigna una dirección IP a cada ordenador. Incluso si su ordenador no esta en una red, se debe incluir el nombre del ordenador en /etc/hosts.

Por ejemplo, si no se esta en una red TCP/IP, y el nombre del ordenador es caspita.org, incluya la línea siguiente en /etc/hosts:

```
127.0.0.1 caspita.org localhost
```

Esto asigna el nombre del ordenador, caspita.org, a la dirección de bucle 127.0.0.1 (utilizada si no se esta en una red). El alias localhost se asigna también a dicha dirección.

Si se esta en una red TCP/IP, sin embargo, su dirección y nombre de ordenador real deben aparecer en /etc/hosts. Por ejemplo, si su nombre de ordenador es nino.rnasalab.com y su dirección IP es 158.153.54.32 (IP escogida al azar), añada la siguiente línea a /etc/hosts:

```
158.153.54.32 nino.rnasalab.com
```

Si el nombre de su ordenador no aparece en /etc/hosts no será posible establecerlo. Para establecer el nombre de su ordenador, utilice el comando hostname. Por ejemplo, el comando

```
# hostname -S nino.rnasalab.com
```

pone el nombre del ordenador a nino.rnasalab.com. En muchos casos, el comando hostname se ejecuta en alguno de los ficheros de inicialización del sistema, como puede ser /etc/rc o /etc/rc.local. Edite estos ficheros y cambie el comando hostname existente para poner su propio nombre de ordenador; al rearrancar el sistema, el nombre del ordenador cambiara al nuevo valor.

2.8.3. Ficheros de "log"

El administrador de una máquina Linux debe revisar los ficheros de registro (o de "log"). Sirven para detectar posibles fallos de seguridad en nuestra máquina Linux. Los ficheros de registro se encuentran en /var/log:

- auth y auth-priv
  Mensajes de seguridad y autenticación.

- cron

  Mensajes generados por el *daemon* "cron".

- secure

  Mensajes de seguridad de acceso (conexiones, wrappers...).

- htmlaccess.log

  Mensajes de acceso al servidor WWW, si lo hubiese.

- utmp

  Contiene información sobre los usuarios que se encuentran actualmente en el sistema. La información la consultamos con comandos como "who", "w", "users"...

- wtmp

  Contiene información sobre las entradas y salidas de los usuarios al sistema. La información la consultamos con el comando "last".

Si ve que en estos ficheros se repiten registros continuos desde máquinas desconocidas, probablemente su máquina esté siendo objeto de una posible incidencia de seguridad.

El fichero de configuración de los registros es el /etc/syslog.conf y el proceso encargado de registrar los eventos es, como ya decíamos, el syslogd.

Capítulo 3

Servicios

3.1. Introducción a las redes

3.1.1. Conceptos previos

Una red de ordenadores es un conjunto de material preparado para que los nodos puedan comunicarse uno con otro (con «nodos» nos referimos a ordenadores, impresoras, y cualquier otra cosa que se le ocurra). No es realmente importante cómo estén conectados: pueden usar cables de fibra óptica o palomas mensajeras. Obviamente, algunas elecciones son mejores que otras.

Normalmente, si se va a limitar a conectar dos ordenadores, no se le llama red; realmente, necesitará tres o más para tener una red. Pasa como con la palabra «grupo»: dos personas son sólo una pareja, pero tres ya pueden ser «grupo». Además, las redes suelen estar conectadas unas con otras, para constituirse en redes más grandes. Cada pequeña red (normalmente llamada «subred») puede ser parte de una red más grande.

La verdadera conexión entre dos ordenadores se llama a menudo “enlace de red” (network link). Si hay un cable que va de la parte posterior de su ordenador hasta las otras máquinas, ese es su enlace de red.

Hay cuatro cuestiones que generalmente tenemos en cuenta al hablar de redes de ordenadores:

Tamaños

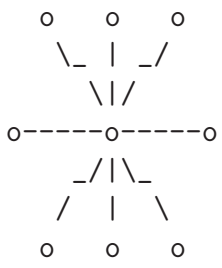
Si va a conectar algunos ordenadores de su casa, tiene lo que llamamos LAN (Local Area Network o Red de Area Local). Si todo está a una distancia razonable que se pueda cubrir caminando, se le suele llamar LAN, da igual cuántas máquinas estén conectadas, y de qué manera esté hecha l la red.

El otro extremo del espectro es una WAN (Wide Area Network o Red de Area Amplia). Si tiene un ordenador en Coruña otro en Londres y otro en Santiago de Chile, e intenta conectarlos, tendrá una WAN.

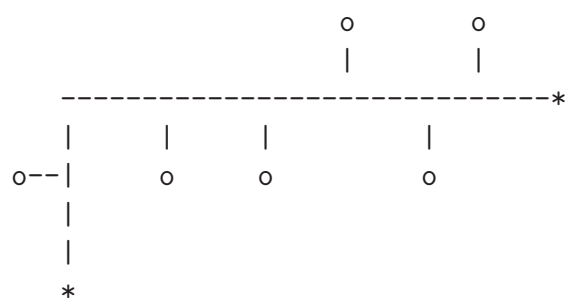
Topologías

La topología es la "forma" que tiene una red, es decir, el grafo que la describe. Se verá de una manera mas sencilla conociendo los algunos de los tipos mas comunes:

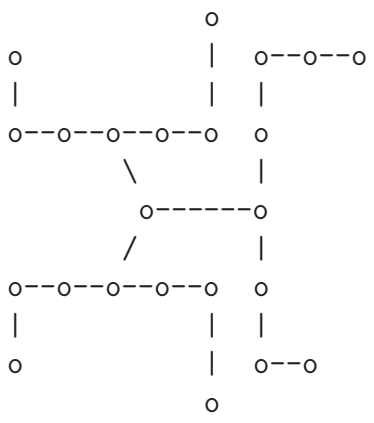
En estrella: cada línea lleva a un nodo central, como una gran estrella, lo que quiere decir que todo el mundo se comunica a través de un punto...



En bus: todos los nodos se conectan siguiendo la misma "línea"...



Subredes conectadas por un solo nodo: en este caso tenemos tres subredes, que se interconectan por un nodo central...



Aspectos físicos

El dispositivo más común usado en casa para conectar a redes mayores se llama "módem" (MODulador/DEModulador), que convierte una línea de teléfono normal en un enlace de red. Transforma la información del ordenador en sonidos, y escucha los sonidos que vienen del otro extremo para convertirlos de nuevo en información para el ordenador. Como puede imaginar, esto no es muy eficiente, y las líneas de teléfono no fueron diseñadas para este uso; pero es popular porque las líneas de teléfono son comunes y baratas.

La manera más común de conectar máquinas en una LAN es usar Ethernet. Ethernet se presenta en las siguientes modalidades principales (listadas de más antigua a la más reciente): Thinwire/Coax/10base2, UTP (Unshielded Twisted Pair/10baseT y UTP/100baseT. También se está empezando a difundir Gigabit ethernet. El cable 10base2 suele ser coaxial negro, con enlaces en forma de T para conectarlos a los objetos: todos están conectados en una gran fila, con "terminadores" especiales en ambos extremos. UTP suele ser cable azul con conectores transparentes al estilo de los teléfonos que se enchufan: cada cable conecta un nodo a un "hub" (un concentrador) central.

En el otro extremo tenemos la Fibra; un delgado filamento de cristal, encerrado en una capa protectora que se puede tender entre continentes.

Generalmente llamamos a cada conexión a un nodo "interfaz de red", o "interfaz" para abreviar. Linux les da nombres como "eth0" para la primera interfaz ethernet, y «fddi0» para la primera interfaz de fibra. La orden /sbin/ifconfig las enumera.

Protocolos

El último detalle por tener en cuenta es el lenguaje que van a hablar los ordenadores. Cuando dos módems se comunican por una línea de teléfono, se tienen que poner de acuerdo en el significado de cada sonido, porque de lo contrario no funcionará. Esta convención se denomina "protocolo". Según se descubren nuevas formas de codificar lo que dicen las computadores en sonidos más pequeños, se inventan nuevos protocolos, y la mayoría de los módemes probarán con varios protocolos hasta que encuentren uno que el otro extremo entienda.

3.1.2. Internet

Internet es una WAN que abarca todo el planeta: es una de las más grandes redes de ordenadores existentes. La expresión "internetworking" se refiere a conectar redes separadas para construir una más grande, de manera que "La Internet" es la conexión de un gran conjunto de subredes.

De manera que examinando la lista anterior nos preguntamos: ¿cuál es el tamaño de Internet, sus detalles físicos y protocolos?

- El tamaño ya lo hemos establecido: es mundial.

- Los detalles físicos, sin embargo, son variados: cada pequeña subred se conecta de forma diferente, con un aspecto y naturaleza física distinta. Los intentos de hacer un mapa útil de Internet han acabado de forma general en un abyecto fracaso.
- Los protocolos que se hablan entre cada enlace también son diferentes a menudo: todos los protocolos de nivel de enlace que nombramos antes, y muchos más.

El funcionamiento de Internet

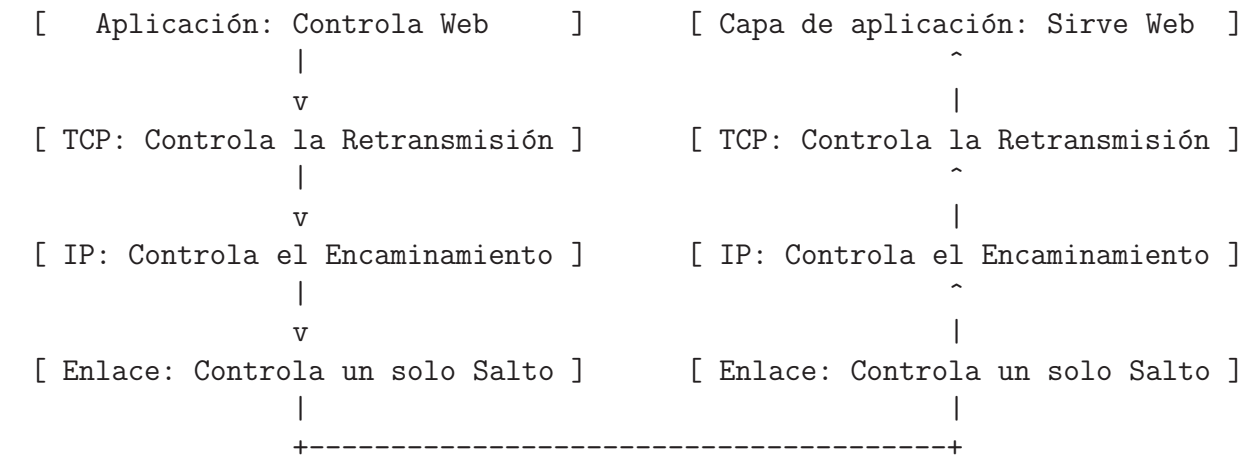
Lógicamente, se nos plantea la pregunta: ¿cómo puede hablar cada nodo de Internet con otros, si todos utilizan diferentes protocolos de nivel de enlace?

La respuesta es muy sencilla: necesitamos otro protocolo que controle cómo fluyen las cosas a través de la red. El protocolo de nivel de enlace describe cómo llegar de un nodo a otro si están conectados de forma directa: el «protocolo de red» nos dice cómo llegar de un punto de la red a otro, yendo a través de otros enlaces si fuera necesario.

Para la Internet, el protocolo de red es el Internet Protocol (versión 4), o «IP». No es el único que hay (tenemos otros como el Appletalk de Apple, IPX de Novell, DECNet de Digital y el NetBEUI de Microsoft) pero es el más ampliamente adoptado. Hay una nueva versión de IP denominada IPv6, pero aún no es tan común.

Para enviar un mensaje de una parte a otra del planeta, su ordenador escribe un fragmento de Internet Protocol, lo envía por el módem, que usa algún protocolo de nivel de enlace de módems para enviarlo al otro módem al que está llamando, que posiblemente esté enchufado a un servidor terminal (básicamente una gran caja de módems), que lo envía a otro nodo dentro de la red del ISP (Internet Service Provider - Proveedor de Servicios de Internet), que lo envía normalmente a otro nodo mayor, que lo manda al siguiente, y así sucesivamente. Un nodo que conecte dos o más redes se llama “router” tendrá una interfaz para cada red.

Llamamos a este conjunto de protocolos una “pila de protocolos”, que a veces se representa de esta manera:



De manera que en el diagrama vemos un Netscape (la Aplicación de la izquierda) obteniendo una página web de un servidor web (la Aplicación de la derecha). Para hacerlo utiliza el "Transmission Control Protocol" o "TCP": alrededor del 90 % del tráfico de la Internet hoy día es TCP, y se emplea para Web y correo electrónico.

Entonces, el Netscape hace una consulta mediante una conexión TCP al servidor web remoto: esto lo controla la capa TCP, que se la pasa a la capa IP, que se hace cargo de la dirección que tiene que seguir, y la pasa a la capa de enlace apropiada, que la transmite al otro extremo del enlace.

En el otro extremo, la capa de enlace la pasa a la capa IP, que comprueba que vaya destinado a esa máquina (si no, puede enviarla a otra capa de enlace diferente para que pase al siguiente nodo), se la entrega a la capa TCP que, por último, se la manda al servidor.

De esta forma, tenemos los siguientes elementos:

1. La aplicación (Netscape, o el servidor web en el otro extremo) decide con quién quiere hablar, (y qué le quiere enviar).

2. La capa TCP envía paquetes especiales para iniciar la conversación con el otro extremo, y entonces empaqueta los datos en "paquetes" TCP: un paquete es sólo un término para describir un grupo de datos que pasan a través de la red. La capa TCP delega este paquete en la capa IP: estará mandándoselo a la capa IP hasta que la capa TCP del otro extremo responda diciendo que lo ha recibido. Esto se llama "retransmisión", e implica gran cantidad de reglas complejas que deciden cuándo retransmitir, cuánto esperar, etc. También le da a cada paquete un número, lo que significa que el otro extremo podrá ponerlos en el orden correcto.

3. La capa IP comprueba el destino del paquete, y averigua el siguiente nodo al que mandárselo. Este sencillo acto se llama "encaminamiento" (routing), y va desde lo realmente sencillo (si sólo tiene un módem, y no hay otra interfaz de red, todos los paquetes saldrán por ahí) a lo extremadamente complejo (si tiene 15 grandes redes conectadas directamente con usted).

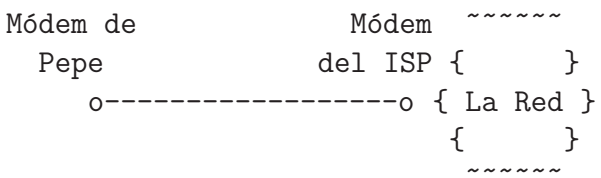
3.1.3. Las IPs

De manera que el papel de la capa IP es averiguar cómo "encaminar" paquetes a su destino final. Para hacerlo posible, cada interfaz en la red necesita una dirección IP. Una dirección IP consiste en cuatro números separados por puntos, tal como "167.216.245.249". Cada número estará entre cero y 255.

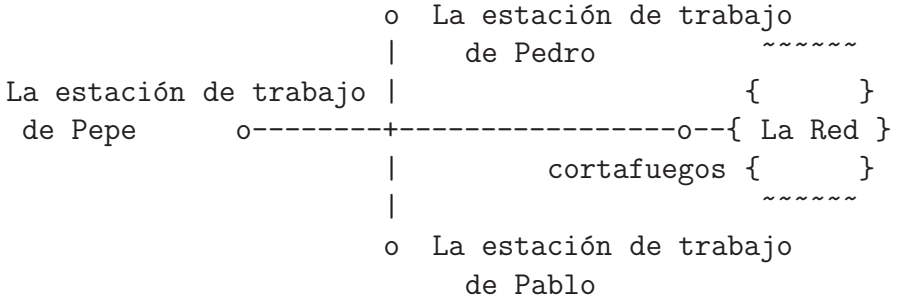
Las interfaces de la misma red tienden a tener direcciones IP vecinas. Por ejemplo "167.216.245.250" estará cerca de la máquina con la dirección IP "167.216.245.249". Recuerde también que un router es un nodo con interfaces en una o más redes, de manera que el router tendrá una dirección IP por cada interfaz.

Por tanto la capa IP del Núcleo de Linux tiene una tabla con diferentes rutas, que describe cómo llegar a varios grupos de direcciones IP. La más sencilla de ellas se llama "ruta por defecto": si la capa IP no sabe qué hacer, es ahí a donde envía los paquetes. Puede ver una lista de las rutas usando /sbin/route.

Las rutas pueden indicar tanto un enlace, como un nodo particular que está conectado a otra red. Por ejemplo, cuando llamamos a un ISP, la ruta por defecto indicará el enlace del módem, porque por ahí se llega al mundo entero.



Pero si tenemos una máquina en nuestra red que conecta con el mundo exterior, es un poco más complejo. En el siguiente diagrama, mi máquina puede comunicarse directamente con las de Pedro y Pablo, y con el cortafuegos ("firewall"), pero necesita saber que los paquetes dirigidos al resto del mundo han de pasar por el cortafuegos, que los reenviará. Esto significa que hay dos rutas: una dice "si está en mi red, sencillamente suéltalo ahí", y luego la ruta por defecto que dice "en cualquier otro caso, envíalo al cortafuegos".



3.1.4. Servidores de nombres (DNS)

De manera que cada interfaz en cada nodo tiene una dirección IP. Rápidamente la gente se dio cuenta que era bastante difícil tener que recordar números, de manera que se decidió (igual que con los números de teléfono) tener un directorio de nombres. Pero como de todas maneras estamos utilizando ordenadores, es mejor que él mismo haga las consultas por nosotros de forma automática.

De manera que tenemos el Domain Name System (DNS, o Sistema de Nombres de Dominio). Hay nodos que tienen direcciones IP bien conocidas a las que los programas pueden preguntar nombres, para obtener direcciones IP a cambio. Casi todos los programas que use podrán hacerlo, y por ello usted puede poner "www.google.com" en el Netscape, en lugar de "216.239.51.101".

Por supuesto, necesita al menos la dirección IP de uno de estos servidores de nombres. Normalmente están almacenados en el fichero /etc/resolv.conf.

Como las consultas y respuestas DNS son bastante pequeñas (un paquete cada una), no se suele usar el protocolo TCP: proporciona retransmisión automática, ordenación, y fiabilidad en general, pero al coste de enviar paquetes adicionales por la red. En su lugar se utiliza el más sencillo "User Datagram Protocol", que no ofrece ninguna de las características de TCP que no necesitamos.

3.2. Gestión de servicios

3.2.1. Servicios y Puertos

Los servidores de red y los servicios son aquellos programas que permiten a un usuario remoto hacer uso de su máquina Linux de alguna forma, ya sea abriendo un terminal remoto, o sirviendo páginas web. Los programas servidores escuchan en los puertos de red.

Los puertos son el medio de llegar a un servicio en particular en una máquina en particular, y es así como un servidor conoce la diferencia entre una conexión telnet y otra de FTP que le lleguen. El usuario remoto establece una conexión de red con la máquina, y el programa servidor, el demonio de red que esté escuchando en ese puerto, aceptará la conexión y se ejecutará. Los puertos se referencian con un número que va del 1 al 65536.

En la siguiente tabla podemos observar los servicios más comunes, junto con sus puertos y protocolos:

Servicio	Puerto	Protocolo	Descripción
echo	7	tcp, udp	Todo lo que se mande a ese puerto lo devuelve
daytime	13	tcp, udp	Devuelve la fecha y la hora del sistema
ftp	21	tcp	Servidor FTP
telnet	23	tcp	Permite conectarnos y abrir una consola remotamente
smtp	25	tcp	Gestiona la distribución del correo de la máquina
timeserver	37	tcp, udp	Devuelve la hora el sistema
named	53	tcp, udp	Servidor de nombres (DNS)
gopher	70	tcp	Sistema de indexación de los servidores FTP (obsoleto)
finger	79	tcp	Devuelve información sobre los usuarios del sistema

Servicio	Puerto	Protocolo	Descripción
http (www)	80	tcp	Servidor WWW
linuxconf	98	tcp	Sistema de configuración remota
pop2	109	tcp	Servidor de correo Pop versión 2
pop3	110	tcp	Servidor de correo Pop versión 3
rpcbind	111	tcp, udp	Servicio de RPC (portmapper)
auth (ident)	113	tcp	Identifica y registra a los usuarios que hace uso de servicios tcp
innd	119	tcp	Servidor de News
netbios	137, 138, 139	tcp, udp	Servidor Samba. Windows for Workgroups
imap2	143	tcp	Servidor de correo Imap versión 2
login	513	tcp	Permite logins remotos (rlogin) a usuarios autorizados
shell	514	tcp	Permite shells remotos (rshell) a usuarios autorizados
syslog	514	udp	Registra todos los sucesos del sistema y los guarda en logs
lpd	515	tcp	Servidor de Impresión
uucp	540	tcp	Antiguo protocolo de comunicación de unix
mountd	635	udp	NFS mount daemon
nfsd	2049	udp	Sistema de ficheros de red de Unix
x-windows	6000		Acepta conexiones de servidores X autorizados

Hay dos modos de operación para los demonios de red. Ambos se usan por igual en la práctica. Las dos maneras son:

- Autónomo (standalone)

  El programa demonio de red escucha en el puerto de red asignado y, cuando llega una conexión, se ocupa él mismo de dar el servicio de red. Estos servicios son demonios que se activan en algún momento en el arranque (probablemente en el /etc/rc.d).

- Esclavo del servidor inetd

  El servidor inetd es un demonio de red especial que se especializa en controlar las conexiones entrantes. Tiene un fichero de configuración que le dice qué programa debe ser ejecutado cuando se reciba una conexión.

A servidor web, por ejemplo, le interesa el modo standalone para el servicio web. Supuestamente este servidor recibirá mucha carga de red por ese puerto, así que no interesa iniciar y terminar el programa cada vez que le llegue una petición http (de web).

Sin embargo, el servicio de telnet, que abre una consola remota al sistema, interesa que esté esclavo de initd. Así, cuando el demonio initd detecte una conexión por ese puerto, iniciará el servicio de telnet, y cuando acabe, lo terminará. De esta forma, un servicio que se utiliza de manera esporádica es iniciado y terminado según se requiera, ahorrando carga del sistema y memoria.

3.2.2. Abrir y cerrar servicios

Hay dos ficheros importantes que necesitamos configurar. Son el fichero /etc/services que asigna nombres a los números de puerto y el fichero /etc/inetd.conf que es el fichero de configuración del demonio de red inetd.

/etc/services

El fichero /etc/services es una base de datos sencilla, que asocia un nombre que nosotros podamos entender, con un puerto de servicio de la máquina. Su formato es bastante simple. Es un fichero de texto en el que cada línea representa una entrada a la base de datos. Cada entrada comprende tres campos separados por cualquier número de espacios en blanco (espacio o tabulador). Los campos son:

```
nombre   puerto/protocolo   sobrenombres   # comentario
```

- nombre

  Una sola palabra que representa el servicio descrito.

- puerto/protocolo

  Este campo se divide en dos subcampos:

  - puerto

    Un número que especifica el número de puerto del servicio que estará disponible. La mayoría de los servicios comunes tienen asignados números de servicio.

  - protocolo

    Este subcampo debe tener como valor tcp o udp, en función del protocolo al que nos refiramos.

- sobrenombres o alias

  Otros nombres que pueden usarse para referirse a esta entrada de servicio.

Cualquier texto que aparezca en una línea después de un caracter # es ignorado y se trata como un comentario.

A continuación podemos ver las primeras líneas de un fichero /etc/services típico:

```
tcpmux          1/tcp                           # TCP port service multiplexer
echo            7/tcp
echo            7/udp
```

```
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
qotd            17/tcp          quote
msp             18/tcp                          # message send protocol
msp             18/udp                          # message send protocol
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
# 24 - private
smtp            25/tcp          mail
# 26 - unassigned
time            37/tcp          timserver
time            37/udp          timserver

[ ... etc ... ]
```

En el día a día, este fichero se encuentra en proceso de continuo crecimiento según se van creando nuevos servicios.

/etc/inetd.conf

/etc/inetd.conf es el fichero de configuración para el demonio servidor inetd. Su función es la de almacenar la información relativa a lo que inetd debe hacer cuando recibe una petición de conexión a un servicio en particular. Para cada servicio que desee que acepte conexiones deberá decirle a inetd qué demonio servidor de red ejecutar, y cómo ha de hacerlo.

Su formato también es relativamente sencillo. Es un fichero de texto en el que cada línea describe un servicio que desee proporcionar. Igual que antes, cualquier texto en una línea que siga a # es ignorado y se considera un comentario. Cada línea contiene siete campos separados por cualquier número de espacios en blanco (espacio o tabulador). El formato general es el siguiente:

```
servicio  tipo-socket  proto  flags  usuario  servidor  argumentos
```

- servicio

  Es el servicio correspondiente a esta configuración, tomado del fichero /etc/services.

- tipo-socket

  Describe el tipo de socket que esta entrada considerará relevante. Los valores permitidos son: stream, dgram, raw, rdm o seqpacket. Es un poco técnico por naturaleza, pero por regla general casi todos los servicios basados en tcp usan stream, y casi todos los basados en udp usan dgram. Sólo algunos demonios servidores muy particulares usarán otros valores.

- protocolo

  El protocolo considerado válido para este servicio. Debería corresponder con la entrada apropiada en el fichero /etc/services y suele ser tcp o udp.

- flags

  Sólo hay dos valores posibles. Este campo le dice a inetd si el programa servidor de red libera el socket después de comenzar la ejecución, y si por tanto inetd podrá ejecutar otro servidor para la siguiente petición de conexión, o si inetd deberá esperar y asumir que el demonio servidor que esté ejecutándose controlará las nuevas peticiones de conexión. Esto tiene su dificultad, pero por norma general todos los servidores tcp deberían tener esta entrada con el valor nowait y la mayoría de servidores udp deberían tener wait.

- usuario

  Este campo indica qué cuenta de usuario de /etc/passwd será asignada como dueña del demonio de red cuando se ejecute. Esto es a menudo útil si quiere protegerse ante riesgos de seguridad. Puede asignar el usuario nobody a una entrada, por lo que si la seguridad del servidor de red es traspasada el posible daño queda minimizado. Habitualmente, sin embargo, este campo está asignado a root, porque muchos servidores requieren privilegios de administrador para funcionar correctamente.

- servidor

  Este campo es el camino completo hasta el programa servidor a ejecutar para esta entrada.

- argumentos

  Este campo comprende el resto de la línea de órdenes y es opcional. Es en donde se pone cualquier argumento de línea de órdenes que desee pasar al programa demonio servidor cuando es ejecutado.

Al igual que pasa con el /etc/services, todas las distribuciones modernas incluirán un buen fichero /etc/inetd.conf para trabajar con él. Aquí incluímos, como ejemplo, algunas de las primeras líneas del fichero /etc/inetd.conf de la distribución Debian:

```
# /etc/inetd.conf:  see inetd(8) for further informations.
#
# Internet server configuration database
#
```

```
#
# Modified for Debian by Peter Tobias <tobias@et-inf.fho-emden.de>
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
# Internal services
#
#echo           stream  tcp     nowait  root    internal
#echo           dgram   udp     wait    root    internal
discard         stream  tcp     nowait  root    internal
discard         dgram   udp     wait    root    internal
daytime         stream  tcp     nowait  root    internal
daytime         dgram   udp     wait    root    internal
#chargen        stream  tcp     nowait  root    internal
#chargen        dgram   udp     wait    root    internal
time            stream  tcp     nowait  root    internal
time            dgram   udp     wait    root    internal
#
# These are standard services.
#
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  /usr/sbin/in.telnetd
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  /usr/sbin/in.ftpd

[ ... etc ... ]
```

3.2.3. Conocer el estado de los servicios de nuestra máquina(netstat)

netstat informa acerca de casi cualquier cosa que se pueda imaginar relacionada con la red. Es especialmente buena para sacar listados de conexiones y sockets activos. Al usar netstat se puede encontrar qué interfaces están activas en qué puertos. Lo que viene a continuación es la salida típica de un servidor, con netstat -a:

```
pepe@nino:~/$ netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:cvspserver            *:*                     LISTEN
tcp        0      0 *:swat                  *:*                     LISTEN
tcp        0      0 *:time                  *:*                     LISTEN
tcp        0      0 *:discard               *:*                     LISTEN
tcp        0      0 *:netbios-ssn           *:*                     LISTEN
tcp        0      0 *:daytime               *:*                     LISTEN
tcp        0      0 *:www                   *:*                     LISTEN
tcp        0      0 *:ftp                   *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 *:smtp                  *:*                     LISTEN
tcp        0      0 nino.dc.fi.:netbios-ssn pon.dc.fi.udc.es:3051   ESTABLISHED
tcp        0      0 nino.dc.fi.udc.es:ssh   tic162.udc.es:1329      ESTABLISHED
```

```
tcp        0    272 nino.dc.fi.udc.es:ssh    213-98-209-45.uc.:33047 ESTABLISHED
tcp        0      0 nino.dc.fi.:netbios-ssn  grenouille.dc.fi.u:3405 ESTABLISHED
udp        0      0 nino.dc.fi.u:netbios-ns  *:*
udp        0      0 *:netbios-ns             *:*
udp        0      0 *:discard                *:*
udp        0      0 nino.dc.fi.:netbios-dgm  *:*
udp        0      0 *:netbios-dgm            *:*
udp        0      0 nino:33205               *:*
udp        0      0 nino:33242               *:*
[...]
pepe@nino:~/$
```

Además también nos devolvería los sockets que están establecidos en ese instante, pero ya no nos resulta interesante para nuestros propósitos.

3.2.4. Conocer el estado de los servicios de otra máquina (nmap)

El programa nmap es una aplicación muy potente para intentar descubrir datos de nodos de una red o incluso de redes, en base a técnicas como el escaneo de puertos. Veremos aquí, como ejemplo, un simple escaneo para intentar averiguar qué puertos tiene abiertos un servidor web, y así conocer que servicios nos puede ofrecer. Probemos por ejemplo...

```
pepe@nino:~$ nmap www.fi.udc.es

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on suevos.fi.udc.es (193.144.60.48):
(The 1594 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
22/tcp     open        ssh
25/tcp     open        smtp
80/tcp     open        http
113/tcp    open        auth
515/tcp    open        printer
748/tcp    open        ris-cm

Nmap run completed -- 1 IP address (1 host up) scanned in 1 second
pepe@nino:~$
```

En este ejemplo vemos que el servidor www.fi.udc.es tiene todos estos puertos abiertos. La primera columna es el número de puerto y protocolo; la segunda, el estado, que puede ser open (abierto), filtered (filtrado por algún cortafuegos) o closed (cerrado). Los puertos cerrados no los enseña.

3.3. Telnet

El protocolo telnet permite establecer una comunicación orientada a caracteres a través de la red, entre una estación cliente y otra servidora.

3.3.1. El cliente

El cliente de telnet puede conectarse a un servidor de cualquier tipo a través del puerto por el que escucha este, enviándole todo lo que reciba por su entrada estándar y mostrando por su salida todo lo que responda el servidor. Por ejemplo, utilizando el cliente básico de Telnet en Linux (comando telnet) se puede establecer una conexión con un servidor Web o de correo electrónico especificando su nombre o número IP y el puerto por el que escucha:

```
pepe@nino:~$ telnet 192.168.168.1 80
Trying 192.168.168.1...
Connected to 192.168.168.1
Escape character is '^]'.
GET /
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
  <TITLE>Test Page for the Apache Web Server on Red Hat Linux</TITLE>
</HEAD>
<!-- Background white, links blue (unvisited), navy (visited), red (active) -->
<BODY BGCOLOR="#FFFFFF">
...
</BODY>
</HTML>
Connection closed by foreign host.

pepe@nino:~$ telnet deltha
Trying 192.168.168.76...
telnet: Unable to connect to remote host: Connection refused
```

La combinación de caracteres `^]` permite establecer el prompt del cliente (este se activa también cuando se invoca telnet sin argumentos). En este punto se pueden ejecutar varios comandos:

- close

  Cierra una conexión.

- help

  Lista todos los comandos disponibles.

- open

  Abre una nueva conexión.

- quit

  Termina la sesión de Telnet.

Cualquiera de estos comandos se puede escribir de forma abreviada siempre y cuando no haya ambigüedades. Por ejemplo para usar help se podría poner “h”, “he” o “hel” que todos harían la misma función.

En prácticamente todos los sistemas operativos modernos incluyendo a Windows existen clientes de Telnet. Estos pueden ser más o menos funcionales en comparación con el cliente de Linux.

3.3.2. El servidor

El servidor de Telnet permite establecer una conexión remota con la máquina Linux donde se ejecute desde un cliente en cualquier otro sistema operativo. Cuando un cliente de Telnet se conecta a un servidor de Telnet este ejecuta un procesos de login de forma similar a como se hace localmente. En caso de que la autenticación inicial sea válida se creará un shell para el usuario conectado.

El servidor de Telnet está controlado por el super-server inetd. Se instala a través del paquete telnet-server. El puerto por el que escucha por defecto es el 23. Para modificarlo se deberá cambiar la línea correspondiente en /etc/services. El fichero de configuración por defecto del servicio es /etc/inetd.d/telnet.

Para deshabilitar el servicio Telnet de una forma diferente se puede crear un fichero cuyo contenido se mostrará siempre que un usuario trate de autenticarse. Este fichero se debe nombrar /etc/nologin y no se utiliza solamente para negar conexiones a través de Telnet, sino también para conexiones locales o mediante otros servicios. La conexión se negará a todos los usuarios con excepción de root.

Existe un fichero nombrado /etc/issue.net cuyo contenido se muestra antes del prompt de login de la sesión Telnet. Existe también /etc/issue que es para las sesiones locales en el sistema. Por defecto en el caso de Red Hat, ambos ficheros tienen un contenido similar al siguiente:

```
Red Hat Linux release 7.1 (Seawolf)
Kernel 2.4.2-2 on an i686
```

En Red Hat estos ficheros se sobreescriben al iniciar el sistema mediante un código en el script de inicio /etc/rc.d/rc.local, por lo que para cambiar su contenido de forma permanente, se necesita eliminar ese código en dicho script.

Desde el punto de vista de la seguridad el servicio de Telnet es totalmente vulnerable pues toda la información que circula a través de la red lo hace de forma natural, sin encriptación, incluyendo los passwords. Es por ello que el usuario root no puede conectarse mediante Telnet, para ello deberá utilizarse otro usuario y luego ejecutar el comando su. Existen otras formas más seguras de establecer una conexión remota utilizando el servicio SSH (Secure SHell). Este servicio (que por defecto utiliza el puerto 23) a diferencia de Telnet, encripta toda la información intercambiada, utilizando para ello diferentes mecanismos.

3.4. File Transfer Protocol (FTP)

Un servicio muy utilizado para acceder a ficheros de forma remota es el servicio FTP. Este se basa en el protocolo del mismo nombre cuyas siglas significan File Transfer Protocol, o sea, es un protocolo para la transferencia de ficheros. Básicamente FTP ofrece facilidades para trasladar ficheros desde el servidor al cliente o viceversa, teniendo en cuenta los permisos establecidos.

El servicio FTP tiene dos variantes:

- Anónima

  La conexión al servidor FTP no requiere de una cuenta previamente creada en este, o sea, no se realiza autenticación basada en usuario. Normalmente el login que se utiliza en este caso es anonymous y el password una dirección válida de correo. Los documentos compartidos mediante FTP anónimo normalmente se agrupan en cierta estructura de directorios con permisos bien restrictivos.

- No anónima

  La conexión al servidor FTP se establece mediante una cuenta de usuario previamente creada en este. El usuario una vez autenticado prodrá acceder a todos los ficheros en el servidor de acuerdo a los permisos del sistema de ficheros aplicados a su caso.

Al igual que en el Telnet, una sesión FTP es insegura, pues toda la información, incluyendo la de autenticación, no se encripta al pasar por la red. Es por ello que normalmente el servidor está configurado para que el usuario root no puede conectarse.

3.4.1. El cliente

El comando que constituye un cliente básico de FTP en Linux, es precisamente ftp. Este recibe como argumento principal la identificación del servidor (nombre o número IP), además del número del puerto (por defecto 21). También se pueden indicar algunas opciones.

Si se logra la conexión con el servidor se realiza la autenticación. En cualquier caso, falle o no la autenticación o no se logre abrir la conexión, se activará el prompt del cliente de forma similar a:

```
ftp>
```

A partir de aquí se podrán ejecutar una gran cantidad de comandos del protocolo. A continuación se listan los más comunes:

- help comando

  Sin argumento muestra la lista de todos los comandos posibles. Si se le indica un comando como argumento muestra la utilización del mismo.

- open host

  Abre una conexión FTP con el host especificado.

- user login

  Permite autenticarse como un usuario determinado.

- ls o dir

  Muestran el contenido del directorio actual en el servidor en formato largo (similar a ls -al).

- get fichero

  Traslada un fichero desde el servidor al cliente.

- put fichero

  Traslada un fichero desde el cliente al servidor.

- mget patrón

  Traslada todos los ficheros que satisfagan cierto patrón (nombre de fichero con comidines) del servidor al cliente.

- mput patrón

  Traslada todos los ficheros que satisfagan cierto patrón del cliente al servidor.

- binary

  Establece el modo de transferencia binario (todo lo que no sea texto ASCII).

- ascii

  Establece el modo de transferencia para texto ASCII.

- cd directorio

  Cambia el directorio actual en el servidor.

- prompt

  Habilita o deshabilita el modo interactivo. Por ejemplo, cuando se transfieren a la vez más de un fichero utilizando un patrón, en el modo interactivo se pregunta para cada uno de ellos si se desea transferir o no.

- lcd directorio

  cambia el directorio actual en el cliente.

- !comando
  Ejecuta un comando en un shell en el cliente.
- close
  cierra la conexión actual.
- exit, by o quit
  Cierran la conexión actual y terminan el cliente.

Al igual que en las sesiones Telnet cualquiera de estos comandos se puede invocar de forma abreviada siempre que no haya ambigüedades.

Otro comando que constituye un cliente de FTP es el ncftp. Este posee numerosas ventajas. Ejemplo de ello son: el auto-comlpetado durante la edición de la línea de comando, la indicación del progreso de las transferencias, la realización de resúmenes de las transferencias, la transferencia recursiva de directorios, la ejecución de trabajos en background, la administración de bookmarks (favoritos), el trabajo con proxies y firewalls, etc.

También existen clientes FTP con interfaz gráfica como el gftp de GNOME. Es posible además emplear los navegadores que soportan el protocolo para establecer sesiones FTP, como por ejemplo el Netscape, o el Galeón. En las powertools de Red Hat se distribuyen otros clientes de FTP con interfaz gráfica o sin ella.

3.4.2. El servidor

La versión del servidor FTP disponible para Linux más común se nombra wu-ftpd y se instala a través de un paquete del mismo nombre. Una vez instalado, el servidor estará preparado para brindar el servicio de forma no anónima, por lo que si se quiere contar además con la variante anónima se deberá instalar el paquete anonftp que creará la estructura especial de directorios donde se guarda la información accesible a través del servicio en dicha variante, además de adecuar la configuración para habilitarla.

El servidor de FTP es controlado por el super-server xinetd. El fichero correspondiente en /etc/xinetd.d/ se nombra wu-ftpd. Por defecto el servidor funciona por el puerto 21 como está especificado en /etc/services.

Los ficheros de configuración fundamentales del servicio FTP se encuentran en el directorio /etc y son ftpaccess, ftpconversions, ftphosts, ftpusers y ftpgroups. En la configuración del FTP se pueden establecer aspectos como: el máximo de usuarios conectados simultáneamente, restricciones de acceso de acuerdo a clases de usuarios, grupos, clientes, límites en la cantidad de datos transmitidos por sesión, etc.

Para las versiones actuales de Red Hat la estructura de directorios accesibles por la variante anónima del FTP, comienza en /var/ftp/. Aquí se ubica a su vez el directorio pub donde se deben colocar todos los ficheros accesibles mediante el servicio.

Existe una herramienta gráfica del entorno KDE para configurar el servicio FTP nombrada kwuftpd.

Existen algunos comandos útiles para el manejo del servicio FTP. Estos son:

- ftpshut
  Detiene el servicio.
- ftprestart
  Reinicia el servicio.
- ftpwho
  Muestra los usuarios conectados al servidor.
- ftpacount
  Muestra la cantidad de usuarios conectados clasificados por clase.

```
[root@nino]:~/> ftpshut now "Sentimos las molestias causadas"
[root@nino]:~/> ftprestart
[root@nino]:~/> ftpwho
Service class all:
 5221 ?        SN     0:10 ftpd: gloin.disaic.cu: pepe: IDLE
 5267 ?        SN     0:00 ftpd: aries.disaic.cu: coco: IDLE
 5285 ?        SN     0:00 ftpd: deltha.disaic.cu: anonymous/alinac@disaic.colombus.cu:
   -   3 users (no maximum)

[root@nino]:~/> ftpcount
Service class all                -   3 users (no maximum)
[root@nino]:~/>
```

3.5. Servidor Web Apache

3.5.1. Introducción al Web

Actualmente uno de los servicios más difundidos y de gran utilidad en numerosos contextos es el World Wide Web o WWW. El Web, como se conoce de forma abreviada, no es más que un amplio y diverso conjunto de documentos vinculados de múltiples formas que se distribuyen por todo el mundo y pueden accederse a través de la red. La mayoría de la información distribuida a través del Web se organiza mediante las conocidas páginas Web, aunque también se pueden difundir objetos de otros tipos. Las páginas Web se escriben en HTML (HyperText Markup Language) el lenguaje para la definición de hipertexto o texto enriquecido. HTML es un lenguaje descriptivo muy sencillo y con numerosas potencialidades. En los inicios del Web todas las páginas eran estáticas, o sea siempre se mostraban de la misma forma en todas las circunstancias posibles. En la actualidad existe una tendencia a definir estas de forma dinámica de acuerdo a diversas situaciones y necesidades. Debido

a esto numerosas aplicaciones han adoptado una interfaz Web para interactuar con sus usuarios y gracias a ello ser más accesibles mediante la red. Esto se logra como resultado de la arquitectura propia del servicio.

La arquitectura del Web es del tipo cliente-servidor. En el servidor es donde se almacena la información estática accedida y/o las aplicaciones que la generan. Los clientes por lo general son los programas conocidos como navegadores o browsers que se encargan de contactar a un servidor ante la solicitud de un usuario y visualizar el resultado de acuerdo a su implementación propia. Ejemplos de navegadores muy conocidos son el Netscape y el Internet Explorer. También está el Konqueror del entorno gráfico KDE, que posee un desarrollo considerable para su poco tiempo de vida, y el Lynx que es el browser que consume menos recursos de todos los disponibles debido a que su interfaz es completamente textual. Los browsers también pueden funcionar como clientes de otros servicios para acceder a recursos a través de la red como son el FTP, NNTP, Wais y Gopher. Es importante aclarar que la dinamicidad de las páginas Web puede estar dada tanto en el lado cliente como en el servidor en dependencia de la tecnología que se emplee, por ejemplo: CGI, ASP y Servlet son tecnologías a aplicarse en el servidor, mientras que: JavaScript y CSS (Cascading Style Sheet) se utilizan en la parte cliente.

El protocolo que emplean el servidor y el cliente para comunicarse es el HTTP (HyperText Transport Protocol) o protocolo para la transmición de hipertexto. Este es un protocolo orientado a caracteres del tipo solicitud/respuesta. Por lo general los servidores Web escuchan las solicitudes de los clientes a través del puerto 80 y es a este a donde se van a dirigir los clientes por defecto para hacer sus solicitudes.

La forma que tiene un usuario en el Web de acceder a una página u objeto de forma general es mediante el empleo de su URL (Uniform Resource Locator) que es una especie de dirección que indica la localización exacta de un documento en el Web. Esta dirección está formada fundamentalmente por dos aspectos: el nombre o dirección IP del servidor y el camino relativo del documento dentro del servidor. También pueden incluirse otros aspectos tales como: el puerto por el que se solicita el servicio (se asume por defecto el 80 para HTTP, el 21 para FTP, etc.), y login y password en documentos que requieran autenticación para ser accedidos.

3.5.2. Conceptos sobre Apache

Apache surgió a partir del servidor de HTTP más famoso y difundido en su época: NCSA. Desde entonces se convirtió en un poderoso rival de todos los servidores Unix utilizados hasta la fecha por su eficiencia, funcionalidad y rapidez. Es por ello que se conoce como el rey de los servidores Web. Se desarrolla de forma estable y segura gracias a la cooperación y los esfuerzos de un grupo de personas conocidas como grupo Apache (Apache Group), los cuales se comunican a través de Internet y del Web. Juntos se dedican a perfeccionar el servidor y su documentación regidos por la ASF (Apache Software Foundation).

El paquete de la distribución Red Hat que contiene la implementación del servidor Apache para Linux se nombra apache. También se dispone de un manual del mismo en el paquete apache-manual. Existe una aplicación con interfaz gráfica en el paquete apacheconf que permite configurar al Apache con las limitaciones propias de dichas interfaces.

El Apache en Red Hat se ejecuta a través de un daemon llamado httpd que se manipula utilizando el script de inicio del mismo nombre en /etc/rc.d/init.d/. Por tanto la forma más sencilla de iniciar, detener, conocer el estado o indicar que relea su configuración al daemon es como se muestra en los ejemplos:

```
[root@nino]:~/> service httpd stop
   Stopping httpd:                                 [  OK  ]

[root@nino]:~/> service httpd start
   Starting httpd:                                 [  OK  ]

[root@nino]:~/> service httpd status
   httpd (pid 6973 6972 6971 6970 6969 6968 6967 6966 6963) is running...

[root@nino]:~/> service httpd reload
   Reloading httpd:                                [  OK  ]
```

3.5.3. Configuración de Apache

Ahora si abrimos una ventana de navegador y escribimos el Nombre del PC, que como sabemos equivale a la dirección IP de LAN del ordenador, veremos la página que Apache pone por defecto como página "root".

Esto quiere decir que funciona el servidor Apache, ya que nos ha "servido" la pagina web mostrada que es la que trae por default. Pero ahora queremos poner nuestra propia pagina Web. ¿Como lo hacemos? Puede que tengamos que hacer unos cambios en la configuración del Apache. La configuración de apache se encuentra por defecto en /etc/apache/httpd.conf. Desde aquí se configura todo el Apache y parece complicado (y la verdad lo es), pero la mayoría de las líneas son comentarios (todas las líneas que empiezan por #) para ayudar a saber que es cada cosa

Por ejemplo, queremos que el apache muestre una Web a los visitantes, la cual esta en "/home/pepe/mi_web" y son documentos .htm o .html. Lo podemos hacer de dos formas:

1. Meter nuestros documentos de paginas Web en el directorio "/var/www/", que es el lugar al que apunta la página "root" del apache por defecto.

2. Hacer que el Apache coja las paginas directamente de "/home/pepe/mi_web".

Para la opción 2 hay que cambiar dos cosas en la configuración. El parámetro del httpd.conf "DocumentRoot" nos indica donde apunta la página raíz. Sólamente modificamos es línea de la forma que nos interesa:

```
DocumentRoot "\home\pepe\mi_web"
```

Pero si seguimos buscando encontraremos la siguiente líneas

```
# This should be changed to whatever you set DocumentRoot to.
<Directory "/var/www/">
```

Deberemos configurar ambas líneas a la dirección que nos interesa...

```
DocumentRoot "/home/pepe/mi_web"

[...]

<Directory "/home/pepe/mi_web">
```

Ya esta. Ahora muy importante; después de cada cambio que hagamos en la configuración deberemos Guardar el archivo httpd.conf y reiniciar el servidor para que los cambios hagan efecto. Para ello, normalmente usaremos el comando apachectl, con el parámetro restart...

```
[root@nino]:~/> apachectl restart
/usr/sbin/apachectl restart: httpd restarted
[root@nino]:~/>
```

Si no nos devuelve ningún error, todo va bien. Si hay errores deberemos repasar los cambios, utilizando apachectl con el parámetro "configtest".

```
[root@nino]:~/> apachectl restart
/usr/sbin/apachectl restart: configuration broken, ignoring restart
/usr/sbin/apachectl restart: (run 'apachectl configtest' for details)
[root@nino]:~/> apachectl configtest
Syntax error on line 324 of /etc/apache/httpd.conf:
DocumentRoot must be a directory
[root@nino]:~/>
```

Este ejemplo nos indica que hemos escrito un directorio no válido en el campo "DocumentRoot", y que debemos repasarlo.

3.6. Compartiendo archivos con máquinas MS-Windows: SAMBA

Samba es una *suite* de utilidades pensadas para trabajar con el protocolo Session Message Block (SMB), también llamado protocolo NetBIOS o LanManager. El protocolo SMB es usado por Microsoft Windows 3.11, NT y 95 para compartir discos e impresoras. Usando el paquete de herramientas Samba creado por Andrew Tridgell, las máquinas UNIX (incluyendo Linux) pueden compartir discos e impresoras con servidores Windows.

Hay cuatro tareas fundamentales para las que se usa samba:

- Compartir una unidad de Linux con máquinas Windows.
- Compartir una unidad de Windows con máquinas Linux.
- Compartir una impresora de Linux con máquinas Windows.
- Compartir una impresora de Windows con máquinas Linux.

Para ello, disponemos de un demonio que hace de servidor de samba, el "smbd", y tambien de clientes capaces de conectar con máquinas que utlicen samba, ya sean Linux o Windows.

3.6.1. El servidor

La configuración de samba se encuentra en /etc/samba/smb.conf o /etc/smb.conf, dependiendo de la distribución.

El formato de este fichero es de la forma...
workgroup = your$_w orkgroup_n ame$
server string = your$_s erver_n ame$
security = share
guest account = guest
encrypt passwords = yes
smb passwd file = /etc/samba/smbpasswd
[homes]
comment = Homes Directories
[cdrom]
    comment = CDROM
    path = /mnt/cdrom
    read only = no
    browseable = yes
    public = yes
A printer can be added like this:
[ljet]
    comment = "Laserjet"
    path = /var/spool/lpd/lp
    printer = lp

```
public = yes
printable = yes
print command = lpr -r -h
  -P
```

Lo del principio es la configuración global:

Workgroup: el nombre del grupo de trabajo

Server string: lo que quieres que se vea como descripción del equipo

Security: share si hay usuarios para samba que no son del sistema o user si son los mismo que los del sistema

Guest account: cuenta de invitado, sin password (por defecto, "nobody").

Encrypt password: si quieres encriptar los passwords.

Smb passwd file: el sitio donde tengas los usuarios y passwords (para usar con el modo "share").

Invalid users: si usas el modo "user", una lista con usuarios no validos (root, por ejemplo).

A partir de aquí, cada servicio se marca entre corchetes. Hacemos una reseña al servicio "homes", que a cada usuario le muestra su directorio home (si usamos la opción "user" en security).

Después, cada servicio lleva los siguientes parámetros:

Comment: el comentario que aparece para cada servicio.

Path: la ruta al servicio.

Read only: si se quiere como "sólo lectura".

Browseable: si se puede navegar por el servicio (si ese usuario tiene permisos, claro).

Public: si es visible para todos los usuarios (para la opcion share).

En las impresoras, además, aparecen las siguientes opciones:

Printer: haciendo referencia a como se llama la impresora (que estará en /var/spool/cups/).

Printable: si se puede imprimir con ella.

Print command: el comando con lo que imprimiras si estubieses en el sistema. En el comando, archivo.

Apéndice A

Estructura de directorios

Un sistema Unix/Linux típico puede tener, entre otros, los siguientes directorios:

- /

  Directorio raíz. Donde comienza el árbol de directorios.

- /bin

  Binarios esenciales en modo monousuario para reparación y arranque.

- /boot

  Fichero de inicio (núcleo), y otros ficheros de carga.

- /dev

  Ficheros de dispositivos.

- /etc

  Ficheros de configuración de la máquina, en los que puede haber directorios dependiendo del programa que contenga los ficheros de configuración. Algunos programas guardarán sus ficheros de configuración en /etc o /usr/etc (que podrían ser enlaces).

- /etc/skel

  Guardará el .^esqueleto", que tendrá en común todo nuevo usuario que se cree.

- /etc/X11

  Ficheros de configuración del sistema X11.

- /home

  Directorio donde se guardarán los usuarios.

- /lib

  Bibliotecas compartidas (shared) o librerías dinámicas, necesarias para el funcionamiento del sistema.

- /mnt

  Directorio sobre el que se montarán las sistemas de ficheros.

- /proc

  Información acerca del estado del núcleo.

- /sbin

  Directorio que contiene comandos, sólo ejecutables para el Superusuario

- /tmp

  Este directorio contiene ficheros temporales que pueden borrarse sin previo aviso. o durante el arranque del sistema.

- /usr

  Normalmente, este directorio se monta desde una partición separada. Debería contener solamente datos compartibles de sólo lectura, de forma que pueda ser montado por varias máquinas que usen Linux.

- /usr/X11R6

  El sistema X-Window, versión 11 distribución 6.

- /usr/X11R6/bin

  Binarios pertenecientes al sistema X11R6 (X Versión 11, revisión 6).

- /usr/X11R6/lib

  Bibliotecas de programas (librerías) asociadas a los binarios.

- /usr/X11R6/lib/X11

  Varios ficheros de distinta utilización para las X.

- /usr/X11R6/include/X11

  Ficheros de cabecera, necesarios para la compilación de las mismas X, o para cualquier otro programa que quieras compilar.

- /usr/bin

  Binarios para el funcionamiento del sistema, deberán de estar aquí todos aquellos programas que puedan usar los usuarios, EXCEPTO los programas que sean del administrador, que debieran estar en /usr/sbin.

- /usr/bin/X11

  Binarios de las X11, generalmente enlace a /usr/X11R6.

- /usr/dict

  Diccionarios de palabras para distintos correctores ortográficos.

- /usr/etc
  Directorio donde se guardan los ficheros de configuración de los distintos programas, los programas que funcionen en /usr/local/bin, tendrán, generalmente su configuración en /usr/local/etc.
- /usr/include
  Ficheros de cabecera para el compilador C.
- /usr/include/X11
  Ficheros de cabecera para el compilador C y el sistema X-Windows.
- /usr/include/asm
  Ficheros de cabecera que declaran algunas funciones de ensamblador.
- /usr/include/linux
  Información acerca de la versión de el sistema Linux. estas cabeceras son necesarias para la compilación de cualquier programa.
- /usr/include/g++
  Ficheros de cabecera para usar con el compilador GNU C++.
- /usr/lib
  Bibliotecas de programas (librerías), y ejecutables que son requeridos para el funcionamiento de algunos programas.
- /usr/lib/X11
  Librerías para las X.
- /usr/lib/zoneinfo
  Ficheros para la información de la zona horaria.
- /usr/local
  Aquí es donde van típicamente los programas que son locales a la máquina.
- /usr/local/bin
  Aquí van los binarios de los programas locales a la máquina.
- /usr/local/doc
  Documentación local.
- /usr/local/etc
  Ficheros de configuración instalados localmente.
- /usr/local/lib
  Aquí van los ficheros asociados a los programas instalados localmente.
- /usr/local/man
  Ayudas.

- /usr/local/src
  Código fuente para los programas instalados localmente.
- /usr/man/¡locale¿/man[1–9]
  Aquellos sistemas que den cabida a varios usuarios de distintas nacionalidades, podrán tener en la cadena ¡locale¿, el lenguaje al que pertenece cada ayuda.
- /usr/sbin
  Programas binarios para la administración del sis- tema.
- /usr/src
  Ficheros fuentes (incluido el núcleo).
- /usr/src/linux
  Núcleo en código fuente.
- /usr/tmp
  Directorio que contiene información temporal.
- /var
  Contenedor de información, como registros de último acceso, colas de impresión, peticiones..., PIDs.
- /var/lock
  En este directorio se crean los ficheros de bloqueo. La convención para nombrar los ficheros de bloqueo es LCK.¡device¿donde ¡device¿es el nombre del dispositivo en el sistema de ficheros. El formato utilizado es el de los ficheros de bloqueo HDU UUCP, esto es, ficheros de bloqueo que contienen un PID como un número decimal ASCII de 10 bytes, seguido por un carácter de salto de línea.
- /var/log
  Ficheros "log"misceláneos.
- /var/preserve
  Copias de seguridad del editor VI.
- /var/run
  Ficheros de variables de ejecución, como los ficheros que contienen los identificadores de proceso (PIDs) y la información de los usuarios "logeados". (utmp). Los ficheros de este directoio se suelen borrar cuando se arranca el sistema.
- /var/spool
  Ficheros en cola para varios programas.
- /var/spool/at
  Trabajos en cola para at(1).

- /var/spool/cron
  Trabajos en cola para cron.
- /var/spool/lpd
  Trabajos en cola para su impresión.
- /var/spool/mail
  Buzones de los usuarios.
- /var/spool/smail
  Ficheros en cola para el programa smail de distribución de correo.
- /var/spool/news
  Directorio de encolado para el subsistema de noticias.
- /var/spool/uucp
  Ficheros en cola para uucp
- /var/tmp
  Como /tmp este directorio contiene ficheros temporales, almacenados durante un tiempo no especificado.

Apéndice B

Comparación de comandos MSDOS - Linux

Esta tabla puede ser útil como toma de contacto para aquellos usuarios con conocimientos del sistema operativo MSDOS:

Comando de MSDOS	Comando de Linux
HELP	man
COPY	cp
MOVE	mv
ECHO	echo
MKDIR	mkdir
RMDIR	rmdir
DIR	ls
CD	cd
ATTRIB	chmod
DEL	rm
DELTREE	rm -r
TYPE	cat
EXIT	exit
UNDELETE	(Aplicación MC opción UNDELETE)
(No Aplicable)	alias
Procesos por Lotes ".BAT"	Scripts
DIR (fichero) /S	find -name fichero
(DOS n/a)	grep -e "cadena de algún fichero" nombre_fichero.txt
(DOS n/a)	strings "cadena de algún fichero" fichero.txt
(DOS n/a)	halt (apagar/detener equipo)
(DOS n/a)	reboot (reiniciar equipo)
PRINT	lpr
DOS (n/a)	lprm (limpiar trabajos de la cola de impresión)
ipconfig	ifconfig (Configuración IP)
winipcfg	ifconfig
ping	ping (Comprobar paquetes/comunicación)

Apéndice C

Licencia Pública GNU

Esta es la conocida GNU Public License (GPL), versión 2 (de junio de 1.991), que cubre la mayor parte del software de la Free Software Foundation, y muchos más programas.

Los autores de esta traducción son Jesús González Barahona y Pedro de las Heras Quirós.

Nota: Esta es una traducción no oficial al español de la GNU General Public License. No ha sido publicada por la Free Software Foundation, y no establece legalmente las condiciones de distribución para el software que usa la GNU GPL. Estas condiciones se establecen solamente por el texto original, en inglés, de la GNU GPL.

C.1. Comienzo de la Licencia Pública GNU

C.1.1. Preámbulo

Las licencias que cubren la mayor parte del software están diseñadas para quitarle a usted la libertad de compartirlo y modificarlo. Por el contrario, la Licencia Pública General de GNU pretende garantizarle la libertad de compartir y modificar software libre, para asegurar que el software es libre para todos sus usuarios. Esta Licencia Pública General se aplica a la mayor parte del software del la Free Software Foundation y a cualquier otro programa si sus autores se comprometen a utilizarla. (Existe otro software de la Free Software Foundation que está cubierto por la Licencia Pública General de GNU para Bibliotecas). Si quiere, también puede aplicarla a sus propios programas. Cuando hablamos de software libre, estamos refiriéndonos a libertad, no a precio. Nuestras Licencias Públicas Generales están diseñadas para asegurarnos de que tenga la libertad de distribuir copias de software libre (y

cobrar por ese servicio si quiere), de que reciba el código fuente o que pueda conseguirlo si lo quiere, de que pueda modificar el software o usar fragmentos de él en nuevos programas libres, y de que sepa que puede hacer todas estas cosas.

Para proteger sus derechos necesitamos algunas restricciones que prohíban a cualquiera negarle a usted estos derechos o pedirle que renuncie a ellos. Estas restricciones se traducen en ciertas obligaciones que le afectan si distribuye copias del software, o si lo modifica.

Por ejemplo, si distribuye copias de uno de estos programas, sea gratuitamente, o a cambio de una contraprestación, debe dar a los receptores todos los derechos que tiene. Debe asegurarse de que ellos también reciben, o pueden conseguir, el código fuente. Y debe mostrarles estas condiciones de forma que conozcan sus derechos.

Protegemos sus derechos con la combinación de dos medidas:

- Ponemos el software bajo copyright y le ofrecemos esta licencia, que le da permiso legal para copiar, distribuir y/o modificar el software. También, para la protección de cada autor y la nuestra propia, queremos asegurarnos de que todo el mundo comprende que no se proporciona ninguna garantía para este software libre. Si el software se modifica por cualquiera y éste a su vez lo distribuye, queremos que sus receptores sepan que lo que tienen no es el original, de forma que cualquier problema introducido por otros no afecte a la reputación de los autores originales.

- Cualquier programa libre está constantemente amenazado por patentes sobre el software. Queremos evitar el peligro de que los redistribuidores de un programa libre obtengan patentes por su cuenta, convirtiendo de facto el programa en propietario. Para evitar esto, hemos dejado claro que cualquier patente debe ser pedida para el uso libre de cualquiera, o no ser pedida.

Los términos exactos y las condiciones para la copia, distribución y modificación se exponen a continuación.

C.1.2. Términos y condiciones para la copia, distribución y modificación

Esta Licencia se aplica a cualquier programa u otro tipo de trabajo que contenga una nota colocada por el tenedor del copyright diciendo que puede ser distribuido bajo los términos de esta Licencia Pública General. En adelante, «Programa» se referirá a cualquier programa o trabajo que cumpla esa condición y «trabajo basado en el Programa» se referirá bien al Programa o a cualquier trabajo derivado de él según la ley de copyright. Esto es, un trabajo que contenga el programa o una proción de él, bien en forma literal o con modificaciones y/o traducido en otro lenguaje. Por lo tanto, la traducción está incluida sin limitaciones en el término «modificación». Cada concesionario (licenciatario) será denominado «usted».

Cualquier otra actividad que no sea la copia, distribución o modificación no está cubierta por esta Licencia, está fuera de su ámbito. El acto de ejecutar el Programa no está restringido, y los resultados del Programa están cubiertos únicamente si sus contenidos constituyen un trabajo basado en el Programa, independientemente de haberlo producido mediante la ejecución del programa. El que esto se cumpla, depende de lo que haga el programa.

Usted puede copiar y distribuir copias literales del código fuente del Programa, según lo has recibido, en cualquier medio, supuesto que de forma adecuada y bien visible publique en cada copia un anuncio de copyright adecuado y un repudio de garantía, mantenga intactos todos los anuncios que se refieran a esta Licencia y a la ausencia de garantía, y proporcione a cualquier otro receptor del programa una copia de esta Licencia junto con el Programa. Puede cobrar un precio por el acto físico de transferir una copia, y puede, según su libre albedrío, ofrecer garantía a cambio de unos honorarios.

Puede modificar su copia o copias del Programa o de cualquier porción de él, formando de esta manera un trabajo basado en el Programa, y copiar y distribuir esa modificación o trabajo bajo los términos del apartado 1, antedicho, supuesto que además cumpla las siguientes condiciones: Debe hacer que los ficheros modificados lleven anuncios prominentes indicando que los ha cambiado y la fecha de cualquier cambio. Debe hacer que cualquier trabajo que distribuya o publique y que en todo o en parte contenga o sea derivado del Programa o de cualquier parte de él sea licenciada como un todo, sin carga alguna, a todas las terceras partes y bajo los términos de esta Licencia. Si el programa modificado lee normalmente órdenes interactivamente cuando es ejecutado, debe hacer que, cuando comience su ejecución para ese uso interactivo de la forma más habitual, muestre o escriba un mensaje que incluya un anuncio de copyright y un anuncio de que no se ofrece ninguna garantía (o por el contrario que sí se ofrece garantía) y que los usuarios pueden redistribuir el programa bajo estas condiciones, e indicando al usuario cómo ver una copia de esta licencia. (Excepción: si el propio programa es interactivo pero normalmente no muestra ese anuncio, no se requiere que su trabajo basado en el Programa muestre ningún anuncio). Estos requisitos se aplican al trabajo modificado como un todo. Si partes identificables de ese trabajo no son derivadas del Programa, y pueden, razonablemente, ser consideradas trabajos independientes y separados por ellos mismos, entonces esta Licencia y sus términos no se aplican a esas partes cuando sean distribuidas como trabajos separados. Pero cuando distribuya esas mismas secciones como partes de un todo que es un trabajo basado en el Programa, la distribución del todo debe ser según los términos de esta licencia, cuyos permisos para otros licenciatarios se extienden al todo completo, y por lo tanto a todas y cada una de sus partes, con independencia de quién la escribió. Por lo tanto, no es la intención de este apartado reclamar derechos o desafiar sus derechos sobre trabajos escritos totalmente por usted mismo. El intento es ejercer el derecho a controlar la distribución de trabajos derivados o colectivos basados en el Programa.

Además, el simple hecho de reunir un trabajo no basado en el Programa con el Programa (o con un trabajo basado en el Programa) en un volumen de almacenamiento o en un medio de distribución no hace que dicho trabajo entre dentro del ámbito cubierto por esta Licencia.

Puede copiar y distribuir el Programa (o un trabajo basado en él, según se especifica en el apartado 2, como código objeto o en formato ejecutable según los términos de los apartados 1 y 2, supuesto que además cumpla una de las siguientes condiciones: Acompañarlo con el código fuente completo correspondiente, en formato electrónico, que debe ser distribuido según se especifica en los apartados 1 y 2 de esta Licencia en un medio habitualmente utilizado para el intercambio de programas, o Acompañarlo con una oferta por escrito, válida durante al menos tres años, de proporcionar a cualquier tercera parte una copia completa en formato electrónico del código fuente correspondiente, a un coste no mayor que el de realizar físicamente la distribución del fuente, que será distribuido bajo las condiciones descritas en los apartados 1 y 2 anteriores, en un medio habitualmente utilizado para el intercambio de programas, o Acompañarlo con la información que recibiste ofreciendo distribuir el código fuente correspondiente. (Esta opción se permite sólo para distribución no comercial y sólo si usted recibió el programa como código objeto o en formato ejecutable con tal oferta, de acuerdo con el apartado b anterior). Por código fuente de un trabajo se entiende la forma preferida del trabajo cuando se le hacen modificaciones. Para un trabajo ejecutable, se entiende por código fuente completo todo el código fuente para todos los módulos que contiene, más cualquier fichero asociado de definición de interfaces, más los guiones utilizados para controlar la compilación e instalación del ejecutable. Como excepción especial el código fuente distribuido no necesita incluir nada que sea distribuido normalmente (bien como fuente, bien en forma binaria) con los componentes principales (compilador, kernel y similares) del sistema operativo en el cual funciona el ejecutable, a no ser que el propio componente acompañe al ejecutable. Si la distribución del ejecutable o del código objeto se hace mediante la oferta acceso para copiarlo de un cierto lugar, entonces se considera la oferta de acceso para copiar el código fuente del mismo lugar como distribución del código fuente, incluso aunque terceras partes no estén forzadas a copiar el fuente junto con el código objeto.

No puede copiar, modificar, sublicenciar o distribuir el Programa excepto como prevé expresamente esta Licencia. Cualquier intento de copiar, modificar sublicenciar o distribuir el Programa de otra forma es inválida, y hará que cesen automáticamente los derechos que te proporciona esta Licencia. En cualquier caso, las partes que hayan recibido copias o derechos de usted bajo esta Licencia no cesarán en sus derechos mientras esas partes continúen cumpliéndola. No está obligado a aceptar esta licencia, ya que no la ha firmado. Sin embargo, no hay hada más que le proporcione permiso para modificar o distribuir el Programa o sus trabajos derivados. Estas acciones están prohibidas por la ley si no acepta esta Licencia. Por lo tanto, si modifica o distribuye el Programa (o cualquier trabajo basado en el Programa), está indicando que acepta esta Licencia para poder hacerlo, y todos sus términos y condiciones para copiar, distribuir o modificar el Programa o trabajos basados en él. Cada vez que redistribuya el Programa (o cualquier trabajo basado en el Programa), el receptor recibe automáticamente una licencia del licenciatario original para copiar, distribuir o modificar el Programa, de forma sujeta a estos términos y condiciones. No puede imponer al receptor ninguna restricción más sobre el ejercicio de los derechos aquí garantizados. No es usted responsable de hacer cumplir esta licencia por terceras partes. Si como consecuencia de una resolución judicial o de una alegación de infracción de patente o por cualquier otra razón (no limitada a asuntos relacionados con patentes) se le imponen condiciones (ya sea por manda-

to judicial, por acuerdo o por cualquier otra causa) que contradigan las condiciones de esta Licencia, ello no le exime de cumplir las condiciones de esta Licencia. Si no puede realizar distribuciones de forma que se satisfagan simultáneamente sus obligaciones bajo esta licencia y cualquier otra obligación pertinente entonces, como consecuencia, no puede distribuir el Programa de ninguna forma. Por ejemplo, si una patente no permite la redistribución libre de derechos de autor del Programa por parte de todos aquellos que reciban copias directa o indirectamente a través de usted, entonces la única forma en que podría satisfacer tanto esa condición como esta Licencia sería evitar completamente la distribución del Programa. Si cualquier porción de este apartado se considera inválida o imposible de cumplir bajo cualquier circunstancia particular ha de cumplirse el resto y la sección por entero ha de cumplirse en cualquier otra circunstancia.

No es el propósito de este apartado inducirle a infringir ninguna reivindicación de patente ni de ningún otro derecho de propiedad o impugnar la validez de ninguna de dichas reivindicaciones. Este apartado tiene el único propósito de proteger la integridad del sistema de distribución de software libre, que se realiza mediante prácticas de licencia pública. Mucha gente ha hecho contribuciones generosas a la gran variedad de software distribuido mediante ese sistema con la confianza de que el sistema se aplicará consistentemente. Será el autor/-donante quien decida si quiere distribuir software mediante cualquier otro sistema y una licencia no puede imponer esa elección.

Este apartado pretende dejar completamente claro lo que se cree que es una consecuencia del resto de esta Licencia.

Si la distribución y/o uso de el Programa está restringida en ciertos países, bien por patentes o por interfaces bajo copyright, el tenedor del copyright que coloca este Programa bajo esta Licencia puede añadir una limitación explícita de distribución geográfica excluyendo esos países, de forma que la distribución se permita sólo en o entre los países no excluidos de esta manera. En ese caso, esta Licencia incorporará la limitación como si estuviese escrita en el cuerpo de esta Licencia. La Free Software Foundation puede publicar versiones revisadas y/o nuevas de la Licencia Pública General de tiempo en tiempo. Dichas nuevas versiones serán similares en espíritu a la presente versión, pero pueden ser diferentes en detalles para considerar nuevos problemas o situaciones. Cada versión recibe un número de versión que la distingue de otras. Si el Programa especifica un número de versión de esta Licencia que se refiere a ella y a «cualquier versión posterior», tienes la opción de seguir los términos y condiciones, bien de esa versión, bien de cualquier versión posterior publicada por la Free Software Foundation. Si el Programa no especifica un número de versión de esta Licencia, puedes escoger cualquier versión publicada por la Free Software Foundation.

Si quiere incorporar partes del Programa en otros programas libres cuyas condiciones de distribución son diferentes, escribe al autor para pedirle permiso. Si el software tiene copyright de la Free Software Foundation, escribe a la Free Software Foundation: algunas veces hacemos excepciones en estos casos. Nuestra decisión estará guiada por el doble objetivo

de de preservar la libertad de todos los derivados de nuestro software libre y promover el que se comparta y reutilice el software en general.

C.1.3. Ausencia de garantía

Como el programa se licencia libre de cargas, no se ofrece ninguna garantía sobre el programa, en todas la extensión permitida por la legislación aplicable. Excepto cuando se indique de otra forma por escrito, los tenedores del copyright y/u otras partes proporcionan el programa «tal cual», sin garantía de ninguna clase, bien expresa o implícita, con inclusión, pero sin limitación a las garantías mercantiles implícitas o a la conveniencia para un propósito particular. Cualquier riesgo referente a la calidad y prestaciones del programa es asumido por usted. Si se probase que el Programa es defectuoso, asume el coste de cualquier servicio, reparación o corrección. En ningún caso, salvo que lo requiera la legislación aplicable o haya sido acordado por escrito, ningún tenedor del copyright ni ninguna otra parte que modifique y/o redistribuya el Programa según se permite en esta Licencia será responsable ante usted por daños, incluyendo cualquier daño general, especial, incidental o resultante producido por el uso o la imposibilidad de uso del Programa (con inclusión, pero sin limitación a la pérdida de datos o a la generación incorrecta de datos o a pérdidas sufridas por usted o por terceras partes o a un fallo del Programa al funcionar en combinación con cualquier otro programa), incluso si dicho tenedor u otra parte ha sido advertido de la posibilidad de dichos daños.

C.2. Cómo aplicar estos términos a sus nuevos programas

Si usted desarrolla un nuevo Programa, y quiere que sea del mayor uso posible para el público en general, la mejor forma de conseguirlo es convirtiéndolo en software libre que cualquiera pueda redistribuir y cambiar bajo estos términos. Para hacerlo, añada los siguientes anuncios al programa. Lo más seguro es añadirlos al principio de cada fichero fuente para transmitir lo más efectivamente posible la ausencia de garantía. Además cada fichero debería tener al menos la línea de «copyright» y un indicador a dónde puede encontrarse el anuncio completo.

```
<una línea para indicar el nombre del programa y una rápida idea de qué hace>

Copyright (C) 19aa <nombre del autor>
```

Este programa es software libre. Puede redistribuirlo y/o modificarlo bajo los términos de la Licencia Pública General de GNU según es publicada por la Free Software Foundation, bien de la versión 2 de dicha Licencia o bien (según su elección) de cualquier versión posterior.

Este programa se distribuye con la esperanza de que sea útil, pero SIN NINGUNA GARANTÍA, incluso sin la garantía MERCANTIL implícita o sin garantizar la CONVENIENCIA PARA UN PROPÓSITO PARTICULAR. Véase la Licencia Pública General de GNU para más detalles.

Debería haber recibido una copia de la Licencia Pública General junto con este programa. Si no ha sido así, escriba a la Free Software Foundation, Inc., en 675 Mass Ave, Cambridge, MA 02139, EEUU.Añada también información sobre cómo contactar con usted mediante correo electrónico y postal.

Si el programa es interactivo, haga que muestre un pequeño anuncio como el siguiente, cuando comienza a funcionar en modo interactivo:

```
Gnomovision versión 69, Copyright (C) 19aa nombre del autor
```

Gnomovision no ofrece ABSOLUTAMENTE NINGUNA GARANTÍA. Para más detalles escriba «show w».

Los comandos hipotéticos «show w» y «show c» deberían mostrar las partes adecuadas de la Licencia Pública General. Por supuesto, los comandos que use pueden llamarse de cualquier otra manera. Podrían incluso ser pulsaciones del ratón o elementos de un menú (lo que sea apropiado para su programa).

También debería conseguir que su empleador (si trabaja como programador) o tu Universidad (si es el caso) firme un «renuncia de copyright» para el programa, si es necesario. A continuación se ofrece un ejemplo, altere los nombres según sea conveniente:

```
Yoyodyne, Inc. mediante este documento renuncia a cualquier interés de
derechos de copyright con respecto al programa Gnomovision (que hace
pasadas a compiladores) escrito por Pepe Programador.

<firma de Pepito Grillo>, 20 de diciembre de 1996

Pepito Grillo, Presidente de Asuntillos Varios.
```

Esta Licencia Pública General no permite que incluya sus programas en programas propietarios. Si su programa es una biblioteca de subrutinas, puede considerar más útil el permitir el enlazado de aplicaciones propietarias con la biblioteca. Si este es el caso, use la Licencia Pública General de GNU para Bibliotecas en lugar de esta Licencia.

Apéndice D

Términos habituales en el argot de Linux

- administración

  Proceso por el cual se mantiene un sistema a punto y operativo. Es una tarea de la que se encarga el administrador o root y sus posibles colaboradores. Abarca acciones tales como: configurar nuevos dispositivos, administrar cuentas, seguridad del sistema...

- AT&T

  Compañía Estadounidense de telecomunicaciones. Una división de esta compañía, la Bells Lab, creó el primer Unix.

- bind

  Berkeley Internet Name Domain. Servidor de nombres de dominio.

- BSD

  Berkeley Software Distribution. Adaptación del UNIX original de AT&T por la universidad de Berkeley.

- boot

  Proceso de arranque en un sistema informático.

- case sensitivity

  GNU/Linux distingue entre minúsculas y mayúsculas, por lo que deberemos tener cuidado a la hora de teclear ordenes o nombres de ficheros.

- COMO

  Del ingles HOWTO. Texto explicativo de COMO hacer algo en particular. Se ocupan de un tema en concreto, por lo que suelen hacer referencia a otros textos. Los podrás encontrar en tu distribución bajo /usr/doc.

- compilar

  Proceso por el cual se "traduce" un programa escrito en un lenguaje de programación a lo que realmente entiende el ordenador.

- consola

  Terminal para introducir comandos en una estación UNIX.

- cron

  Demonio que usa el administrador para delegar ciertas tareas que pueden ser ejecutadas sin su participación. Este demonio puede ser programado para ejecutar las tareas a intervalos variables, anualmente, semanalmente, diariamente etc. Sus tareas típicas suelen ser el borrado de ficheros temporales, conexiones con otros equipos, backups, etc.

- cuenta

  Una cuenta en un sistema Unix/Linux puede ser algo así como la llave de un taller comunitario. Es decir, tenemos una llave personal que nos permite acceder a ese taller y utilizar algunas de las herramientas del mismo. Donde además tenemos que atenernos a las normas que rijan en ese taller.

- cuota

  Es un sistema del que se vale el administrador, para regular el espacio que los diferentes usuarios de un sistema ocupan con sus ficheros en disco.

- demonio

  Aparte del significado que todos conocemos, en Unix/Linux se conoce como un programa que permanece en segundo plano ejecutándose continuamente para dar algún tipo de servicio. Ejemplos de demonio, son los servidores de correo, impresora, sistemas de conexión con redes etc.

- display

  Variable de entorno, cuyo valor apunta al servidor Xwindow del usuario que lo esta ejecutando.

- dns

  Domain Name Server. Servidor de nombres de dominio. Servicio de red que nos facilita la búsqueda de ordenadores por su nombre de dominio.

- dosemu

  Emulador del sistema operativo DOS de Microsoft. Ejecuta gran parte de programas para este sistema operativo, incluidos juegos.

- dvi

  Formato de fichero de los formateadores de texto TeX y LaTeX.

- enlaces

  Los enlaces o links permiten tener "copias" de un mismo archivo, ocupando solo el espacio del archivo real. Es decir, el enlace no es mas que otro archivo que apunta a el original.

- ethernet

  Son redes que permiten distribuir datos a través de un solo cable por lo que necesitan de un protocolo especial que evite la colisión de los paquetes de datos, ya que solo se permite el envío de un solo paquete al mismo tiempo, encargándose el protocolo de su reenvío en caso de la colisión de ambos.

- expresiones regulares

  Las expresiones regulares o "regexp" permiten definir el patrón de análisis en una cadena de texto. De forma que a la hora de modificarlas, borrarlas, o lo que queramos hacer con ellas, sea de acuerdo a unas reglas que definimos.

- ext2

  Sistema de ficheros utilizado en GNU/Linux . Permite el uso de permisos para los ficheros y directorios, y tiende a fragmentarse mucho menos que los de otros sistemas operativos.

- filtro

  Un filtro es un programa o conjunto de estos, que procesan una serie de datos generando una salida modificada conforme a lo que nosotros le especifiquemos.

- finger

  Muestra información del usuario que le especificamos. Puede ser de nuestra misma máquina o de otra cualquiera. La información que aparece puede ser todo lo completa que haya querido el usuario que consultemos, ya que aunque el sistema nos muestra una información por defecto, el usuario puede completarla por medio de los ficheros .plan y .project.

- FSF

  Free Software Foundation. Fundación que pretende el desarrollo de un sistema operativo libre tipo UNIX. Fundada por Richard Stallman, empezó creando las herramientas necesarias para su propósito, de modo que no tuviera que depender de ninguna compañía comercial. Después vino la creación del núcleo, que todavía se encuentra en desarrollo.

- gcc

  GNU C Compiler. El compilador estándar de la FSF.

- getty

  Procesos que controlan cada una de las terminales que están conectadas al sistema, o las terminales virtuales que podamos tener abiertas. Establecen las características de los terminales y llaman al proceso encargado de validar la entrada al sistema de los usuarios.

- ghostscript

  Programa encargado de la visualización de ficheros de texto con formato postcript.

- GNU

  GNU is Not Unix. Proyecto de la FSF para crear un sistema UNIX libre.

- GPL

  General Public License. Una de las mejores aportaciones de la FSF. Es una licencia que protege la creación y distribución de software libre.

- GID

  Group IDentification. En UNIX/LINUX se definen grupos para administrar las herramientas a las que tienen acceso unos y otros, el pertenecer a un determinado grupo nos puede permitir, por ejemplo, tener acceso a internet. La pertenencia a algún grupo viene determinada por el numero GID establecido en el cuarto campo del fichero /etc/passwd.

- host

  Nombre de un ordenador en una red.

- HOWTO

  Véase COMO.

- http

  HyperText Transfer Protocol.Protocolo de red para la transferencia de paginas de hipertexto, o lo que es lo mismo, paginas web como esta.

- inetd

  Demonio encargado de mantener en escucha determinados puertos y de llamar a determinados programas en función de las señales recibidas.Por ejemplo, atiende a las llamadas de telnet, finger o ftp.

- init

  Init es el primer proceso que se ejecuta en un sistema UNIX/Linux y el que inicia todos los procesos getty. Tiene varios estados, llamados niveles de ejecución, que determinan los servicios que pueden ofrecer. Por ejemplo, dependiendo del nivel de ejecución podemos establecer la posibilidad de poner la máquina en modo monousuario, impidiendo la posibilidad de acceso a otras personas.

- inode

  Todos los archivos en UNIX/Linux tienen un inode que mantienen información referente al mismo, tal como situación, derechos de acceso, tamaño o tipo de fichero.

- kernel

  Véase núcleo

- latex

  Lenguaje para el formateado de textos. Muy potente y completo.

- lilo

  LInux LOader. Programa que nos permite elegir que sistema operativo arrancar, en el caso de tener varios.

- login

  Programa encargado de la validación de un usuario a la entrada al sistema. Primero pide el nombre del usuario y después comprueba que el password sea el asignado a este.

- loopback

  Sistema de trabajo en red en modo local. Con este sistema podemos trabajar en red con nuestro propio ordenador, su utilidad radica en probar programas de seguridad, leer las noticias o el correo de los servidores instalados en nuestro ordenador o simplemente poder ejecutar Xwindow.

- lpd

  Demonio encargado de asistir a las peticiones de impresión por parte del sistema.

- LuCaS

  Organización de voluntarios dedicada a la traducción de documentación del sistema GNU/Linux al castellano.

- man

  Manual en linea del sistema. Aquí puedes buscar casi cualquier cosa relacionada con el sistema, sus comandos, las funciones de biblioteca, etc.

- mbr

  Master Boot Record. Tabla de información referente al tamaño de las particiones.

- módulos

  Porciones de código que se añaden en tiempo de ejecución al kernel para el manejo de dispositivos o añadir funciones al núcleo.

- monousuario

  Sistema informático que solo admite el trabajo con una persona.

- montar

  Poner un dispositivo o un sistema de ficheros en disposición de ser usado por el sistema.

- mtools

  Conjunto de herramientas para la administración de ficheros, disquetes o discos duros con el sistema de archivos de msdos.

- multitarea

  Capacidad de un sistema para el trabajo con varias aplicaciones al mismo tiempo.

- multiusuario

  Capacidad de algunos sistemas para ofrecer sus recursos a diversos usuarios conectados a través de terminales.

- núcleo

  Parte principal de un sistema operativo, encargado del manejo de los dispositivos, la gestión de la memoria, del acceso a disco y en general de casi todas las operaciones del sistema que permanecen invisibles para nosotros.

- password

  Palabra clave personal, que nos permite el acceso al sistema una vez autentificada con la que posee el sistema en el fichero passwd.

- path

  Variable del entorno, cuyo valor contiene los directorios donde el sistema buscara cuando intente encontrar un comando o aplicación. Viene definida en los ficheros .bashrc o .bash_profile de nuestro directorio home.

- permisos

  Todos los archivos en UNIX/Linux tienen definido un set de permisos que permiten establecer los derechos de lectura, escritura o ejecución para el dueño del archivo, el grupo al que pertenece y los demás usuarios.

- PID

  Process IDentification. Numero que identifica un proceso en el sistema, este numero es único para cada proceso.

- postscript

  Formato profesional de impresión para impresoras de gama alta.

- ppp

  Point to Point Protocol. Protocolo de transmisión de datos, utilizado en la mayoría de las conexiones a internet domesticas.

- proceso

  Programa en ejecución en un sistema informático.

- prompt

  El prompt es lo siguiente que vemos al entrar al sistema, una linea desde donde el sistema nos indica que esta listo para recibir ordenes.

- redirección

  Con los operadores de redirección podemos dirigir la salida de un proceso hacia un dispositivo diferente al estándar o a un fichero.

- root

  Persona o personas encargadas de la administración del sistema Tiene TODO el privilegio para hacer y deshacer, por lo que su uso para tareas que no sean absolutamente necesarias es muy peligroso.

- señales

  Las señales son eventos que se hacen llegar a un proceso en ejecución para su tratamiento por este. Las señales las podemos mandar nosotros u otros programas a otros programas. Tienen diferentes valores, y en función a esos valores el proceso que las recibe actúa de una manera u otra.

- smtp

  Simple Mail Transfer Protocol. Más claro el agua.

- swap

  Memoria virtual. Espacio de disco duro que utiliza el kernel en caso de necesitar mas memoria de la que tengamos instalada en nuestro ordenador.

- terminal

  Una terminal es un teclado y una pantalla conectados por cable u otro medio a un sistema UNIX/Linux, haciendo uso de los recursos del sistema conectado.

- tubería

  Las tuberías son como conexiones entre procesos. La salida de un proceso la encadenamos con la entrada de otro, con lo que podemos procesar unos datos en una sola linea de comando.

- UID

  User IDentification. Número que identifica al usuario frente al sistema.

- UNIX

  Sistema operativo creado por AT&T a mediados de los 70.

- Window manager

  Gestor de ventanas.Programa que se encarga de dar apariencia a los programas que se ejecutan bajo Xwindow, también se encarga de maximizar/minimizar ventanas, ponerles el marco, un fondo al escritorio...

- X o XWindow

  Entorno gráfico. Es el programa que se encarga de dibujar en pantalla todo lo que le solicitan los procesos que corren bajo este entorno. Tiene la facultad de visualizar programas que están siendo ejecutados en otro ordenador de la red.

- X11R6

  Ultima versión utilizada del sistema de ventanas Xwindow.

- xterm

  Terminal virtual que funciona bajo el sistema de ventanas Xwindow.