

# ADMINISTRACIÓN DE SISTEMAS LINUX RED HAT

---

Ramón M. Gómez Labrador

([ramon.gomez@eii.us.es](mailto:ramon.gomez@eii.us.es))

Febrero de 2.003

# Administración de sistemas Linux Red Hat

## ÍNDICE

<b>1. Introducción.....</b>	<b>5</b>
1.1. Tareas del administrador. ....	5
1.1.1. Planificación y previsión de necesidades. ....	5
1.1.2. Documentación. ....	6
1.1.3. Automatización. ....	6
1.1.4. Informar a los usuarios. ....	6
1.1.5. Control de la seguridad del sistema. ....	7
1.1.6. Previsión de fallos. ....	7
<b>2. Usuarios y grupos.....</b>	<b>9</b>
2.1. Características generales de una cuenta. ....	9
2.2. Ficheros del sistema. ....	9
2.3. Usuarios y grupos predefinidos. ....	11
2.4. Clave de acceso. ....	12
2.4.1. Restricciones para tener claves seguras. ....	13
2.5. Permisos. ....	13
2.5.1. Permisos normales. ....	14
2.5.2. Permisos especiales. ....	15
2.5.3. Notaciones simbólica y octal. ....	15
2.6. Configuración del entorno. ....	17
2.7. Gestión de cuentas. ....	17
2.7.1. Planificación. ....	18
2.7.2. Ejemplo: servidor de prácticas universitarias. ....	19
2.8. Métodos de acceso (PAM). ....	20
2.8.1. Servicios PAM. ....	20
2.9. Cuotas. ....	21
<b>3. Intérpretes de mandatos.....</b>	<b>23</b>
3.1. Redirecciones. ....	23
3.2. Tuberías. ....	24
3.3. Variables. ....	25
3.3.1. Tipos de variables. ....	25
3.4. Expresiones. ....	27
3.5. Entrecomillado. ....	27
3.6. Programación estructurada. ....	28
3.6.1. Listas de mandatos. ....	28
3.6.2. Estructuras condicionales y selectivas. ....	28
3.6.3. Bucles. ....	29
3.6.4. Funciones. ....	30
<b>4. Sistemas de archivos.....</b>	<b>31</b>
4.1. Normas para la Jerarquía de Sistemas de Archivos (FHS). ....	31
4.2. Discos y particiones. ....	32
4.3. Sistemas de archivos de Linux (ext2 y ext3). ....	33
4.4. Paginación y procesos. ....	35

4.4.1. Espacios de paginación. ....	35
4.4.2. Sistema de archivos virtual /proc. ....	35
4.5. Discos redundantes (RAID). ....	36
4.6. Volúmenes lógicos. ....	38
4.7. Sistemas de archivos remotos. ....	38
4.7.1. NFS.....	39
4.7.2. CIFS. ....	40
<b>5. Arranque y servicios. ....</b>	<b>42</b>
5.1. Proceso de arranque. ....	42
5.2. Cargadores de arranque. ....	42
5.2.1. GRUB. ....	43
5.2.2. LILO.....	44
5.3. El Núcleo.....	44
5.3.1. Módulos.....	45
5.3.2. Parámetros de operación. ....	46
5.4. Niveles de arranque. ....	47
5.5. Configuración del sistema. ....	48
5.6. Servicios.....	48
<b>6. Instalación de programas. ....</b>	<b>50</b>
6.1. RPM. ....	50
6.2. Archivadores y compresores. ....	51
6.3. Programas GNU. ....	52
6.3.1. Compilar un paquete RPM.....	52
6.3.2. Compilar un paquete comprimido.....	52
<b>7. Configuración de la red. ....</b>	<b>54</b>
7.1. Interfaces de red. ....	54
7.2. TCP/IP.....	55
7.3. Configuración de la red. ....	56
7.4. Servicios de red. ....	57
7.4.1. Breve descripción de los principales servicios de red.....	57
<b>8. Utilidades de administración. ....</b>	<b>59</b>
8.1. Ejecución cronológica. ....	59
8.1.1. cron. ....	59
8.1.2. at, batch. ....	60
8.2. Control de procesos. ....	61
8.3. Medidas de rendimiento. ....	62
8.3.1. Mandatos systat.....	62
8.3.2. IPTraf. ....	62
8.4. Revisión de ficheros históricos. ....	63
<b>9. Impresoras.....</b>	<b>65</b>
9.1. Servicios de impresión. ....	65
9.1.1. LPRng.....	66
9.1.2. CUPS.....	67
<b>10. Copias de seguridad. ....</b>	<b>70</b>
10.1. Planificación. ....	70
10.2. Utilidades. ....	71
10.2.1. tar.....	71
10.2.2. AMANDA. ....	72
<b>11. Seguridad. ....</b>	<b>73</b>
11.1. Tipos de ataques.....	73

11.2. Utilidades de seguridad. ....	74
11.2.1. TCP-Wrappers, Xinetd. ....	74
11.2.2. OpenSSH, OpenSSL. ....	75
11.2.3. Tripwire. ....	76
11.2.4. Netfilter. ....	76
11.2.5. Sudo. ....	77
11.2.6. chkrootkit. ....	78
11.2.7. SNORT. ....	78
<b>12. Referencias. ....</b>	<b>79</b>
<b>APÉNDICE A. Cuestionario del curso. ....</b>	<b>80</b>
<b>APÉNDICE B. Respuestas al cuestionario. ....</b>	<b>83</b>
<b>APÉNDICE C. Ejemplos de programas. ....</b>	<b>84</b>
C.1. variables. ....	84
C.2. especiales. ....	84
C.3. dircom. ....	84
C.4. usuario. ....	85
C.5. comprus. ....	86
C.6. crearus. ....	88
C.7. borrar.cuentas. ....	90
C.8. borrar.cuentas. ....	92
C.9. dnis.repes. ....	93
C.10. listar.expiracion. ....	94
C.11. borrado.diario. ....	95
C.12. comprobar.paquetes. ....	97
C.13. tararear. ....	97

# 1. Introducción.

Linux es un sistema operativo de la familia Unix, gratuito, creado mediante la política de “código abierto”. Estas características implican un gran ahorro en los costes de instalación de los equipos, pero también una mayor especialización por parte del personal informático.

En todo sistema Unix existe un usuario administrador (**root**), que controla el funcionamiento completo del sistema, tiene acceso universal y puede realizar cualquier operación con los datos y los dispositivos de la máquina.

Este curso de formación pretende ofrecer una guía que ayude al administrador de sistemas a configurar, refinar, proteger y mantener sistemas que trabajen bajo el sistema operativo Linux, en particular con el “dialecto” de Red Hat.

## 1.1. *Tareas del administrador.*

El administrador de cualquier tipo de servidor debe ser una persona especializada, que conozca lo mejor posible sus equipos, sus aplicaciones y sus usuarios; manteniéndose al día en los avances tecnológicos, en las revisiones y parches de los programas instalados y en las necesidades de su empresa.

### 1.1.1. Planificación y previsión de necesidades.

Una de las funciones principales en la administración de sistemas informáticos es la planificación detallada de las tareas de gestión, lo que puede evitar sorpresas desagradables en el momento de ejecutarlas.

El analista de sistemas tiene la obligación de asesorar al personal administrativo de su empresa sobre las necesidades tecnológicas en la adquisición de material informático, estimando los recursos que precisen los usuarios, en relación con las posibilidades económicas de la empresa.

Una vez recibido el equipo debe realizarse un plan de instalación, en el que se incluya, al menos la siguiente información:

- Documentación y estudio de los recursos disponibles.
- Previsión de posibles ampliaciones.
- Relleno de solicitud de alta en la red informática corporativa y activación de los parámetros de conexión.
- Documentación de necesidades del entorno de operación (SAI, aire acondicionado, etc.).
- Documentación sobre registro, configuración, instalación y actualización del sistema operativo, de las aplicaciones requeridos y de los programas propios, de acuerdo con los servicios que debe prestar el nuevo equipo.

- Creación y publicación de solicitudes de apertura y modificación de cuentas de usuarios, de instalación de programas, de mejora de recursos, etc.

### **1.1.2. Documentación.**

El responsable del sistemas se compromete a realizar también documentación interna para el Centro de Cálculo, que debe describir las siguientes necesidades:

- Registro actualizado de los usuarios y grupos del sistema.
- Políticas de utilización y permisos para cada grupo de usuarios.
- Descripción de los procedimientos comunes que deben ejecutar los operadores del sistema (copias de seguridad, gestión de cuentas, informes, etc.).
- Registro completo y actualizado de los cambios en la configuración del servidor (sistema operativo, aplicaciones, ficheros, etc.).
- Recogida periódica y archivado de datos sobre el rendimiento del sistema y de sus componentes.

### **1.1.3. Automatización.**

El personal informático de una empresa ha de ejecutar periódicamente las funciones definidas en el plan de actuación. El programador necesita automatizar la mayoría de estos procedimientos repetitivos para evitar errores tipográficos o conceptuales, y para mejorar el tratamiento general de las aplicaciones.

En cada servidor deben automatizarse, al menos, las siguientes tareas:

- Comprobación del espacio libre en los discos.
- Gestión de cuentas de usuarios y revisión periódica de las cuotas de disco.
- Procedimientos para crear, comprobar y restaurar copias de seguridad, según el plan de actuación.
- Comprobación y registro del rendimiento general del sistema y de la red informática.
- Trabajos específicos (informes, gestión de servicios, creación de documentación, etc.).
- Creación de alertas de seguridad (comprobación de cambios, detección de intrusos, etc.).

### **1.1.4. Informar a los usuarios.**

El administrador de sistema debe también mantener informados a sus usuarios y darles unas guías de operación y buen uso, lo que puede evitar errores provocados por desconocimiento.

También es necesario informar sobre los cambios que pueden afectar a cada grupo de usuarios, indicando la siguiente información <sup>[1]</sup>:

- La naturaleza de los cambios que van a realizarse en el sistema y su evolución temporal.
- Cuándo se realizará cada modificación.
- Qué resultados se esperan obtener con la operación y cuáles son los obtenidos.
- Tiempo estimado y tiempo real de la duración de la operación.
- Impacto posible sobre los usuarios (nueva configuración, parada del sistema, etc.).
- Información de contacto para recoger dudas y consultas.

Por otro lado, el encargado del sistema tiene la obligación de conocer profundamente el comportamiento general de sus usuarios, registrando sus consultas, sus sugerencias y los datos de rendimiento y utilización de recursos. Esto permite ofrecer una mejor calidad en los servicios ofertados.

### **1.1.5. Control de la seguridad del sistema.**

Dependiendo del tipo de información tratada por el sistema, el administrador debe definir sus políticas de seguridad, tanto para el servidor, como para la red corporativa, ya que los usuarios tienen derecho a la privacidad e integridad de sus datos.

Deben ponerse los medios para evitar posibles ataques o fallos informáticos que afecten –o incluso paralicen– el funcionamiento normal de la máquina.

Nunca hay que tener la presunción de que un sistema es completamente seguro o de que sólo puede ser atacado desde fuera. Por ello, el *superusuario* debe realizar las siguientes operaciones:

- Activar y revisar los registros históricos de incidencias.
- Realizar revisiones periódicas sobre posibles cambios no deseados en el sistema.
- Instalar aplicaciones y dispositivos que protejan a los servidores y a la red informática (sistemas de detección de intrusos, cortafuegos, filtros, lectores de tarjetas de acceso, etc.).

### **1.1.6. Previsión de fallos.**

Por último, la empresa debe poner los medios físicos necesarios para prevenir y corregir los posibles fallos informáticos.

Por otra parte, los cambios ambientales (eléctricos, temperatura, humedad, ...) son algunos de los aspectos más importantes y costosos en la prevención de errores

. Debe hacerse hincapié en los siguientes temas:

- Tener una correcta instalación eléctrica, que evite caídas y subidas inesperadas de tensión, así como instalar sistemas de alimentación ininterrumpida (SAI) que protejan los servicios críticos de la empresa (armarios de comunicaciones, servidores, etc.).
- Tener un adecuado sistema de aire acondicionado, que filtre y regule la temperatura y la humedad del ambiente, sin que afecte a la salud de los operadores.

- Contar con un alumbrado adecuado, que no afecte al tendido eléctrico informático.
- Mantener una adecuada infraestructura en la red informática, con acceso cómodo y restringido a los dispositivos de comunicaciones.

Otras posibles causas de fallos más difíciles de prever son:

- Saturación o fallo de los recursos del sistema (procesadores, memoria, discos, etc.). Hay que sopesar la necesidad de solicitar la ampliación o sustitución de los componentes afectados.
- Fallos de programación, tanto en el S.O., como en las aplicaciones instaladas o en los programas propios. El administrador debe mantenerse informado sobre las actualizaciones y parches que tenga que instalar.
- Errores humanos del propio administrador, de los operadores, del servicio técnico o de los usuarios finales.



## 2. Usuarios y grupos.

Un usuario Unix representa tanto a una persona (**usuario real**) como a una entidad que gestiona algún servicio o aplicación (**usuario lógico**) <sup>[2]</sup>.

Todo usuario definido en el sistema se corresponde con un identificador único (**UID**) y con una **cuenta**, donde se almacenan sus datos personales en una zona de disco reservada.

Un **grupo** es una construcción lógica –con un nombre y un identificador (**GID**) únicos– usada para conjuntar varias cuentas en un propósito común <sup>[1]</sup>, compartiendo los mismos permisos de acceso en algunos recursos. Cada cuenta debe estar incluida como mínimo en un grupo de usuarios, conocido como **grupo primario**.

### 2.1. Características generales de una cuenta.

Las características que definen la cuenta de un usuario son:

- Tiene un nombre y un identificador de usuario (UID) únicos en el sistema.
- Pertenece a un grupo principal.
- Puede pertenecer a otros grupos de usuarios.
- Puede definirse una información asociada con la persona propietaria de la cuenta.
- Tiene asociado un directorio personal para los datos del usuario.
- El usuario utiliza en su conexión un determinado intérprete de mandatos, donde podrá ejecutar sus aplicaciones y las utilidades del sistema operativo.
- Debe contar con una clave de acceso personal y difícil de averiguar por parte de un impostor.
- Tiene un perfil de entrada propio, donde se definen las características iniciales de su entorno de operación.
- Puede tener una fecha de caducidad.
- Pueden definirse cuotas de disco para cada sistema de archivos.
- Es posible contar con un sistema de auditoria que registre las operaciones realizadas por el usuario.

### 2.2. Ficheros del sistema.

Linux proporciona varios métodos para la definir los usuarios que pueden conectarse al sistema. Lo típico es definir localmente en cada servidor las cuentas de los usuarios y grupos, aunque también pueden usarse métodos externos de autenticación, que permiten que varias máquinas compartan las mismas definiciones para sus usuarios comunes.

La siguiente tabla muestra los ficheros del sistema involucrados en el proceso de definición de los usuarios locales.

Formato	Descripción
<b>/etc/passwd</b>	
<i>Usuario:x:UID:GID:Descrip:Direct:Shell</i> ...	<p>Fichero principal de descripción de usuarios locales. Sus campos son:</p> <ol style="list-style-type: none"> <li>1. Nombre de usuario.</li> <li>2. No usado (antiguamente, clave).</li> <li>3. Identificador de usuario (UID).</li> <li>4. identificador del grupo primario.</li> <li>5. Descripción o nombre completo de la persona que representa dicho usuario.</li> <li>6. Directorio personal.</li> <li>7. Intérprete de mandatos.</li> </ol>
<b>/etc/shadow</b>	
<i>Usuario:clave:F1:N1:N2:N3:N4:Caduc:</i> ...	<p>Fichero oculto que incluye la codificación y las restricciones de las claves de acceso a las cuentas. Sus campos son:</p> <ol style="list-style-type: none"> <li>1. Nombre de usuario.</li> <li>2. Clave codificada.</li> <li>3. Fecha del último cambio de clave.</li> <li>4. Días hasta que la clave pueda ser cambiada.</li> <li>5. Días para pedir otro cambio de clave.</li> <li>6. Días para avisar del cambio de la clave.</li> <li>7. Días para deshabilitar la cuenta tras su caducidad.</li> <li>8. Fecha de caducidad.</li> <li>9. Reservado (ignorado por Red Hat).</li> </ol> <p>Nota: Las fechas se expresan como el nº de días desde el 1/1/1.970.</p>
<b>/etc/group</b>	
<i>Grupo:x:GID:Usuarios</i> ...	<p>Contiene la definición de los grupos de usuarios. Sus campos son:</p> <ol style="list-style-type: none"> <li>1. Nombre del grupo.</li> <li>2. No usado (antiguamente, clave del</li> </ol>

	grupo). 3. Identificador del grupo (GID). 4. Lista de miembros (separada por comas).
<b>/etc/gshadow</b>	
<i>Grupo:Clave:Admins:Usuarios</i> ...	Fichero oculto y opcional que contiene las claves de grupos privados. Sus campos son: 1. Nombre del grupo. 2. Clave codificada (opcional). 3. Lista de usuarios administradores. 4. Lista de usuarios normales.

### 2.3. Usuarios y grupos predefinidos.

En todos los “dialectos” Unix existen algunos usuarios y grupos predefinidos por el sistema operativo, que se utilizan para la gestión y el control de los distintos servicios ofrecidos por el ordenador.

En especial el **usuario root** –con **UID 0**– es el administrador de la máquina, con un control total sobre el sistema. Existe también un **grupo root** –con **GID 0**– con características administrativas, al que pertenece el citado usuario.

La siguiente tabla lista algunos de los usuarios y grupos predefinidos en la versión 8.0 del Linux de Red Hat <sup>[1]</sup>.

Usuario	UID	GID	Descripción
root	0	0	Administrador con control total.
bin	1	1	Propietario de las utilidades del sistema operativo.
daemon	2	2	Gestor de servicios generales.
adm	3	4	Propietario de los archivos de registros históricos y administrativos.
lp	4	7	Administrador de los servicios de impresión.
ftp	14	50	Controlador del acceso al árbol del servicio FTP anónimo..
nobody	99	99	Gestor de servicios varios.
apache	48	48	Propietario de los ficheros y directorios del servicio de hipertexto Apache.
squid	23	23	Controlador del servicio de representación Squid.
ldap	55	55	Permite el acceso al servicio de directorios LDAP.

Grupo	GID	Descripción
root	0	Administradores con control total.
bin	1	Binarios del sistema.
daemon	2	Servicios generales.
sys	3	Control del sistema.
adm	4	Ficheros históricos y administrativos.
tty	5	Acceso a la consola.
lp	7	Servicio de impresión.
kmem	9	Control de memoria del núcleo de Linux.
man	15	Páginas de manuales.
ftp	50	Servicio FTP anónimo.
nobody	99	Control de servicios.
users	100	Usuarios normales.
floppy	19	Acceso a disquetes.
apache	48	Servicio de hipertexto HTTP.
squid	23	Servicio representante.
ldap	55	Servicio LDAP.

Los usuarios ficticios, que gestionan los servicios ofrecidos por el ordenador, deben tener su cuenta deshabilitada para evitar una posible puerta de entrada para los intrusos. Esto se consigue bloqueando la clave de acceso y asignando `/sbin/nologin` como intérprete de mandatos de la cuenta.

## 2.4. Clave de acceso.

Como se ha indicado anteriormente, las claves de los usuarios locales de Linux se guardan codificadas en el fichero inaccesible `/etc/shadow`.

Los algoritmos de codificación de las claves son “de sentido único”, o sea que impiden la descodificación directa de las claves. Por lo tanto, cuando un usuario entra en el sistema, se le codifica la clave y se compara con la clave válida encriptada. Si el resultado es correcto, el usuario puede conectarse.

El Linux de Red Hat puede utilizar el algoritmo de codificación **Crypt**, usado en los antiguos sistemas Unix y llamada así por la función del lenguaje C que realiza el algoritmo. Este método es inseguro porque genera códigos de sólo 13 caracteres, donde los 2 primeros son la semilla de generación. Las claves tienen que tener un máximo de 8 caracteres.

El sistema operativo también soporta la codificación **MD5**, mucho más robusta y con una longitud del código de 34 caracteres, permitiendo claves más extensas y difíciles de averiguar.

### 2.4.1. Restricciones para tener claves seguras.

El administrador debe recomendar a sus usuarios que creen claves que puedan resultar difíciles de averiguar para un pirata informático.

También debe hacer que el sistema cree dificultades al intruso, usando codificaciones complejas y creando restricciones que comprometan al usuario con la seguridad del sistema.

Todos los usuarios del sistema han de tener en cuenta las siguientes recomendaciones con sus claves:

- No usar palabras comunes o números asociados a la persona.
- No repetir las claves en distintas máquinas.
- Usar claves de 8 caracteres como mínimo, con al menos 2 caracteres no alfabéticos.
- No usar secuencias de teclado.
- Cambiar la clave periódicamente y no repetir claves anteriores.
- No dejar ni anotar la clave.
- Evitar que otra persona vea teclear la clave.

## 2.5. Permisos.

Uno de los elementos fundamentales de la seguridad en Unix es el buen uso de los permisos para acceder a ficheros y directorios. Todo usuario –no sólo el administrador– debe tener claros los conceptos más básicos para evitar que otro usuario lea, modifique o incluso borre datos de interés <sup>[12]</sup>.

El usuario administrador –al tener el control completo del sistema– también puede realizar estas operaciones sobre cualquier fichero o directorio de cualquier usuario (esta es una de las maneras de evitar que un usuario pueda entrar en su directorio personal).

Este hecho hace imprescindible que los responsables de la máquina tengan especial cuidado cuando utilicen la cuenta del usuario **root**.

Los permisos de acceso se dividen principalmente en dos categorías: permisos normales y especiales.

Por otro lado, los permisos también se subdividen en tres grupos: permisos para el propietario, para su grupo y para el resto de usuarios del sistema.

### 2.5.1. Permisos normales.

Cada usuario tiene un nombre de conexión único en el ordenador y pertenecerá a uno o varios grupos de usuarios. El propietario de un fichero o directorio puede seleccionar qué permisos desea activar y cuales deshabilitar.

Para comprobarlo de manera más clara, tómese el primer grupo de valores obtenidos con el mandato `ls -l`, que permitirá observar los permisos. Estos 10 caracteres indican:

- 1 carácter mostrando el tipo: fichero (-), directorio (d), enlace (l), tubería (p), etc.
- 3 caracteres para los permisos del propietario.
- 3 caracteres para los permisos de otros usuarios del grupo.
- 3 caracteres para los permisos del resto de usuario.

Según el tipo de entrada del directorio, los caracteres de permisos normales pueden variar de significado:

<b>Ficheros:</b>	<b>Lectura (r):</b> el usuario puede leer el fichero. <b>Escritura (w):</b> el usuario puede escribir en el fichero. <b>Ejecución (x):</b> el usuario puede ejecutar el fichero (siempre que sea un ejecutable o un guión de intérprete de mandatos).
<b>Directorios:</b>	<b>Lectura (r):</b> el usuario puede leer el contenido del directorio. <b>Escritura (w):</b> el usuario puede crear, modificar y borrar entradas del directorio. <b>Acceso (x):</b> el usuario puede acceder al directorio y puede usarlo como directorio actual (utilizando el mandato <code>cd</code> ). Este tipo de permiso posibilita proteger cierta información de un directorio padre y, sin embargo, acceder a la información de los directorios hijos.

La siguiente tabla muestra los permisos necesarios para poder ejecutar algunos mandatos <sup>[12]</sup>.

Mandato	Permisos directorio origen	Permisos fichero	Permisos directorio destino
<code>cd</code>	X	No aplicable	No aplicable
<code>ls</code>	R	No aplicable	No aplicable
<code>mkdir</code>	W, X	No aplicable	No aplicable
<code>rmdir</code>	W, X	No aplicable	No aplicable
<code>cat</code>	X	R	No aplicable
<code>rm</code>	W, X	-	No aplicable

<b>cp</b>	<b>X</b>	<b>R</b>	<b>W, X</b>
<b>mv</b>	<b>W, X</b>	<b>-</b>	<b>W, X</b>

### 2.5.2. Permisos especiales.

Los permisos especiales complementan, potencian la seguridad del sistema y se utilizan para soportar ciertas operaciones específicas.

Al igual que en el punto anterior, dependiendo del tipo de entrada del directorio, los caracteres de permisos especiales representados por **ls -l** son <sup>[12]</sup>:

<b>Ficheros:</b>	<p><b>Identificador de usuario activo</b> (<b>s</b> para el propietario): un programa ejecutable puede activar el identificador de usuario (<b>SUID</b>), esto permite que durante la ejecución del programa un usuario se convierta en el usuario propietario del fichero. Por ejemplo, el mandato <b>passwd</b> accede a ficheros que sólo puede modificar el usuario <b>root</b>. Dicho mandato tiene activo el SUID para que durante la ejecución del programa otro usuario sea por algún momento <b>root</b> y pueda cambiar su clave. Hay que tener especial cuidado con estos ejecutables, porque usuarios no autorizados pueden tomar privilegios.</p> <p><b>Identificador de grupo activo</b> (<b>s</b> para el grupo): al igual que en el caso anterior, un programa ejecutable puede activar el identificador de grupo (<b>SGID</b>) para que un usuario pueda realizar operaciones propias del grupo al que pertenece el fichero. Por ejemplo, el mandato <b>mail</b> activa el SGID para que cualquier usuario pueda acceder a su buzón de correo sin posibilidad de leer correo de cualquier otro usuario.</p>
<b>Directorios:</b>	<p><b>Directorio de intercambio</b> (<b>t</b> en el resto de usuarios): permite que en directorios compartidos los ficheros sólo puedan ser modificados por el propietario (suele usarse en directorios para ficheros temporales como <b>/tmp</b>).</p> <p><b>Identificador de grupo activo</b> (<b>s</b> para el grupo): los ficheros que se creen en dicho directorio tendrán el mismo grupo que el del propio directorio, en vez del grupo del propietario.</p>

El administrador debe catalogar todos los ficheros y directorios creados tras la instalación del sistema operativo o de cualquier aplicación, y que contengan permisos especiales. Periódicamente debe comprobar el estado de dichos archivos y verificar que no han sido modificados.

### 2.5.3. Notaciones simbólica y octal.

La orden **chmod** se utiliza para modificar los permisos de acceso descritos anteriormente y soporta dos tipos de notaciones: simbólica y numérica en formato octal.

La siguiente tabla muestra la forma de asignar permisos en ambas notaciones.

Permisos normales		Valor octal	Notación simbólica
Propietario:	Lectura	400	u+r
	Escritura	200	u+w
	Ejecución / Acceso	100	u+x
Grupo:	Lectura	40	g+r
	Escritura	20	g+w
	Ejecución / Acceso	10	g+x
Resto de usuarios:	Lectura	4	o+r
	Escritura	2	o+w
	Ejecución / Acceso	1	o+x
Permisos especiales		Valor octal	Notación simbólica
Propietario:	Usuario activo (SUID)	4000	u+s
Grupo:	Grupo activo (SGID)	2000	g+s
Resto de usuarios:	Directorio de intercambio	1000	+t

La notación simbólica se utiliza para añadir (+), quitar (-) o asignar (=) permisos agrupados según su tipo.

La notación numérica en formato octal sirve para asignar todos los permisos a la vez, aplicando una operación lógica O para obtener el resultado.

Véase un ejemplo. Si el usuario tiene permiso de modificación en el directorio y si es propietario de los archivos, se ejecutarán las siguientes modificaciones:

- A `fichero1` se le asignan los permisos de lectura y escritura para el propietario y el grupo asociado, y se le quitan (si existen) los de escritura y ejecución para otros usuarios.
- A `fichero2` se le asignan directamente los permisos de lectura y escritura para el propietario y de lectura para su grupo. El resto de usuarios no tiene ningún permiso.

```
> chmod ug=rw,o-wx fichero1
> chmod 640 fichero2
```



## 2.6. Configuración del entorno.

El intérprete de mandados de cada cuenta de usuario tiene un entorno de operación propio, en el que se incluyen una serie de variables de configuración.

El administrador del sistema asignará unas variables para el entorno de ejecución comunes a cada grupo de usuarios –o a todos ellos–; mientras que cada usuario puede personalizar algunas de estas características en su perfil de entrada, añadiendo o modificando las variables.

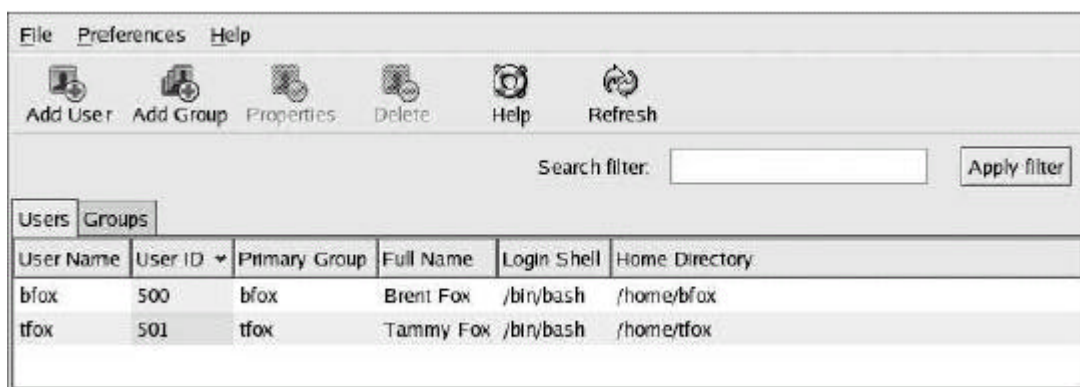
Para crear el entorno global, el administrador crea un perfil de entrada común para todos los usuarios (archivo `/etc/bashrc` en el caso de BASH), donde –entre otros cometidos– se definen las variables del sistema y se ejecutan los ficheros de configuración propios para cada aplicación.

Estos pequeños programas se sitúan en el subdirectorio `/etc/profile.d`; debiendo existir ficheros propios de los intérpretes de mandatos basados en el de Bourne (BSH, BASH, PDKSH, etc.), con extensión `.sh`, y otros para los basados en el intérprete C (CSH, TCSH, etc.), con extensión `.csh`.

El proceso de conexión del usuario se completa con la ejecución del perfil de entrada personal del usuario (archivo `~/.bash_profile` para BASH). Aunque el administrador debe suministrar un perfil válido, el usuario puede retocarlo a su conveniencia. En el siguiente capítulo se presentan las variables de entorno más importantes usadas por BASH.

## 2.7. Gestión de cuentas.

El siguiente gráfico presenta la herramienta gráfica **redhat-config-users**, usada para la gestión básica de usuarios y grupos. Con este programa pueden realizarse las operaciones más sencillas de revisión y control, pero resulta bastante pobre para una gestión automatizada y avanzada de las cuentas de los usuarios <sup>[3]</sup>.



El sistema operativo ofrece también algunos mandatos de gestión, que deben ser usados para personalizar y automatizar el proceso de creación, revisión y eliminación de usuarios y grupos. La siguiente tabla describe dichas funciones.

<b>Mandato</b>	<b>Descripción</b>
<b>useradd</b>	Crea una nueva cuenta de usuario.
<b>usermod</b>	Modifica los parámetros de una cuenta.
<b>userdel</b>	Borra una cuenta de usuario.
<b>passwd</b>	Modifica la clave de acceso a una cuenta.
<b>chpasswd</b>	Cambia la clave a varios usuarios usando un fichero de entrada de datos.
<b>chage</b>	Cambia las restricciones temporales de una cuenta.
<b>chfn</b>	Cambia la descripción del usuario.
<b>chsh</b>	Cambia el intérprete de mandatos de la cuenta.
<b>groupadd</b>	Crea un nuevo grupo de usuarios.
<b>groupmod</b>	Modifica los parámetros de un grupo de usuarios.
<b>groupdel</b>	Elimina un grupo de usuarios.
<b>gpasswd</b>	Cambia la clave a un grupo privado.

### 2.7.1. Planificación.

La gestión de las cuentas de los usuarios es uno de los aspectos más importantes dentro de las tareas administrativas, por ello deben planificarse detalladamente las características y las necesidades de los usuarios y de los grupos que vayan a darse de alta en el sistema.

Fundamentalmente, deben realizarse las siguientes operaciones previas antes de crear cualquier cuenta:

- Crear los distintos grupos de usuarios, uno para cada conjunto de tareas que vayan a ejecutar los usuarios, o uno por cada rol administrativo.
- Definir los parámetros globales del sistema, tales como: restricciones para la creación de claves, método principal de acceso, posibilidad de almacenamiento remoto de las cuentas, etc.
- Crear la estructura de directorios básica para las cuentas, separando los subdirectorios de cada grupo principal. Asignando los permisos adecuados, puede evitarse que usuarios con menor privilegio accedan a zonas reservadas de otros grupos.
- Definir listas privadas donde el administrador pueda comprobar la identidad de cada usuario, almacenando los datos básicos de cada persona y de su cuenta asociada.
- Crear los programas para la gestión de las cuentas, generando ficheros de configuración que automaticen los procesos de creación, modificación, revisión, caducidad y borrado de usuarios.

## 2.7.2. Ejemplo: servidor de prácticas universitarias.

Para ilustrar el proceso de gestión de cuentas, la siguiente tabla describe resumidamente una estructura que puede usarse en un servidor para prácticas universitarias, relativamente parecida a la existente en el Centro de Cálculo de la E.T.S. de Ingeniería Informática de la Universidad de Sevilla.

<b>Restricciones generales para claves:</b>	Las definidas anteriormente en este capítulo.
<b>Creación de grupos de usuarios:</b>	Crear grupos para administradores, alumnos normales, alumnos de proyectos fin de carrera y profesores. Un grupo para cada departamento. Puede ser necesario definir grupos para alumnos por curso.
<b>Estructura de directorios:</b>	Directorio privado para el grupo de administradores. Directorio privado para profesores con subdirectorios privados para cada departamento. Directorios para alumnos normales agrupados por cursos y para alumnos de proyectos. Directorio para apuntes, con permisos de escritura para profesores y de lectura para alumnos.
<b>Crear listas de usuarios:</b>	Generar una lista distinta para cada grupo de usuarios..
<b>Programas de gestión:</b>	Creación de perfiles de configuración para los programas, donde se almacena información por defecto para cada tipo de usuarios y para la generación de los menús de selección. Creación interactiva de cuentas usando dichos perfiles. Creación automática de varias cuentas usando un fichero de datos de entrada. Comprobación de datos de usuarios; mostrando el contenido de la lista correspondiente, la entrada del fichero <code>/etc/passwd</code> , el directorio de la cuenta, la fecha de caducidad y la cuota de disco. Comprobación de concordancia entre los datos de las listas de usuarios y las cuentas creadas. Registro de cuotas de disco y comprobación semanal de su estado. Comprobación de la caducidad de las cuentas. Renovación automática de cuentas. Eliminación automática de cuentas caducadas. Borrado interactivo de cuentas y sus directorios. Registro de incidencias sobre bloqueo y desbloqueo de cuentas. Cambio automático de claves.

## 2.8. Métodos de acceso (PAM)

El método típico de conexión a un servidor Linux es introduciendo el nombre de usuario y su clave asociada. Si los datos coinciden con los ficheros locales de la máquina, se continúa con el proceso de entrada en la cuenta.

La biblioteca **PAM** (*Pluggable Authentication Modules*) permite definir una arquitectura modular de métodos de conexión al sistema, como la autenticación mediante un servicio de directorios LDAP, a través de un dominio Samba o NT, conexión mediante páginas amarillas (NIS o NIS+), autenticación Kerberos, etc.

Cuando se usa correctamente, PAM presenta la siguientes ventajas <sup>[2]</sup>:

- Provee un esquema de autenticación típico usado por una gran variedad de aplicaciones.
- Tanto administradores como desarrolladores de aplicaciones obtienen una gran flexibilidad y control sobre el proceso de autenticación.
- Permite al desarrollador enfocar su código sobre los detalles del programa, sin tener que implementar un esquema de autenticación particular.

### 2.8.1. Servicios PAM.

El directorio `/etc/pam.d` contiene los ficheros de configuración para cada servicio que utiliza los módulos PAM. Estos ficheros constan de una línea para cada módulo con el siguiente formato:

```
TipoMódulo Control Módulo [ Argumentos ... ]
```

Cada servicio PAM puede constar de varios módulos, almacenados normalmente en el directorio del sistema `/lib/security`. La siguiente tabla presenta los 4 tipos de módulos definidos por la norma PAM.

Tipo de servicio		Descripción
Autenticación	<code>auth</code>	Autenticación del usuario (claves, credenciales, billetes, etc.).
Cuenta	<code>account</code>	Comprobación de acceso (caducidad, restricciones horarias, etc.).
Clave	<code>password</code>	Activación y modificación de claves.
Sesión	<code>session</code>	Gestión de la sesión del usuario (montar directorios, buzón, etc.).

Cada módulo PAM genera una salida que indica si se ha ejecutado correctamente o se ha producido algún error. Dependiendo de estos resultados, el fichero de configuración de cada servicio puede definir una serie de indicadores para cada módulo, descritos en la siguiente tabla.

Identificador de control		Descripción
Requerido	<code>required</code>	El módulo debe ejecutarse con éxito, se notifica el posible fallo de autenticación después de comprobar todos los módulos de este tipo.
Requisito	<code>requisite</code>	El módulo debe ejecutarse con éxito y se notifica el error justo al comprobarse el fallo de conexión.
Suficiente	<code>sufficient</code>	La comprobación de acceso se ignora si ésta falla, pero se permite la conexión si es correcta y no lo impide un módulo de tipo requerido.
Opcional	<code>optional</code>	Se determina el éxito de la conexión, sólo si no hay ningún otro módulo del mismo tipo que haya decidido previamente el estado de la autenticación.

## 2.9. Cuotas.

La cuota es una herramienta administrativa que permite monitorizar y limitar el acceso a disco en cada sistema de archivos por parte de determinados usuarios y grupos <sup>[7]</sup>. Los límites controlados por estas herramientas impiden que los usuarios saturen o acaparen el espacio de almacenamiento común.

El sistema de cuotas define 2 tipos de límites para cada sistema de archivos: número de `i` nodos creados (número de ficheros y directorios) y número de bloques de disco que pueden almacenarse. Para cada uno de estos límites existe un **valor tope** y un **valor máximo**. El usuario nunca podrá superar el valor máximo y sólo podrá superar el tope durante un determinado **periodo de gracia**.

Las cuotas se configuran editando el archivo `/etc/fstab`, añadiendo la propiedad deseada en el campo de opciones correspondiente a cada sistema de archivos donde se va a activar este servicio. Los tipos de cuota son:

- `usrquota`: cuota por usuario.
- `grpquota`: cuota por grupo de usuarios.

En el directorio raíz de cada sistema de archivos con cuota debe existir un fichero `quota.user` –si se activan las cuotas por usuario– y otro `quota.group` –si se activan las cuotas para grupos–, ambos con acceso restringido.

Los ficheros que contienen los datos de las cuotas deben estar muy protegidos, su propietario debe ser **root** y sus permisos han de tener un valor de **0600**.

Cada uno de estos ficheros guarda información codificada sobre los límites normal y máximo para el número de i-nodos y para el espacio de disco ocupado, el periodo de gracia y el estado actual de estos valores para cada usuario o grupo con limitación de cuotas de disco.

La siguiente tabla describe brevemente los mandatos que se usan para configurar y gestionar el sistema de cuotas.

Mandato	Descripción
<b>quotaon</b>	Activa el sistema de cuotas.
<b>quotaoff</b>	Para el sistema de cuotas.
<b>edquota</b>	Edita los límites de cuotas para usuarios o grupos y define el periodo de gracia.
<b>quota</b>	Muestra el estado de la cuota del usuario en cada sistema de archivos.
<b>repquota</b>	Presenta un informe completo del estado del sistema de cuotas.
<b>quotacheck</b>	Comprueba y actualiza las cuotas del sistema de archivos.

### 3. Intérpretes de mandatos.

El **intérprete de mandatos** o "*shell*" es la interfaz principal entre el usuario y el sistema, permitiéndole a aquél interactuar con los recursos de éste. El usuario introduce sus órdenes, el intérprete las procesa y genera la salida correspondiente.

La mayoría de estos intérpretes se utilizan como lenguajes de programación, donde el usuario puede crear **guiones** o "*shellscripts*" utilizando combinaciones de mandatos.

En Unix existen 2 familias principales de intérpretes de mandatos: los basados en el intérprete de Bourne (BSH) y los basados en el intérprete C (CSH).

En el resto del capítulo se hará referencia especialmente a **BASH** (*Bourne Again Shell*) – evolución de BSH–, ya que es el intérprete de mandatos más utilizado en Linux e incluye un completo lenguaje para programación estructurada y gran variedad de funciones internas.

#### 3.1. Redirecciones.

Unix hereda 3 ficheros especiales del lenguaje de programación C, que representan las funciones de entrada y salida de cada programa. Estos son:

- **Entrada estándar:** procede del teclado; abre el fichero descriptor 0 (`stdin`) para lectura.
- **Salida estándar:** se dirige a la pantalla; abre el fichero descriptor 1 (`stdout`) para escritura.
- **Salida de error:** se dirige a la pantalla; abre el fichero descriptor 2 (`stderr`) para escritura y control de mensajes de error.

Estos ficheros de E/S pueden ser redirigidos hacia o desde ficheros normales, realizando las funciones de lectura o escritura sobre dichos ficheros. La tabla describe las redirecciones soportadas por BASH.

Redirección	Descripción
<code>&lt; Fichero</code>	Lee la entrada de un fichero.
<code>&gt; Fichero</code>	Escribe la salida normal en un fichero.
<code>2&gt; Fichero</code>	Escribe la salida de error en un fichero.
<code>&gt;&amp; Fichero</code> <code>&amp;&gt; Fichero</code>	Escribe las salida normal y de error en un fichero.
<code>2&gt;&amp;1</code>	Redirige la salida de error hacia la salida normal.
<code>&gt;&amp;2</code>	Redirige la salida normal hacia la salida de error.

<code>&lt;&gt; Fichero</code>	Abre el fichero para leer y escribir la salida estándar.
<code>&gt;&gt; Fichero</code>	Añade la salida estándar al final del fichero.
<code>2&gt;&gt; Fichero</code>	Añade la salida de error al final del fichero.
<code>&lt;&lt;[-] Delimitador</code> <code>    Texto</code> <code>    ...</code> <code>Delimitador</code>	<p>Se usa el propio <i>shellscript</i> como entrada estándar, hasta la línea donde se encuentra el delimitador. No se realizan sustituciones de parámetros, comandos o expresiones.</p> <p>El texto debe ir siempre tabulado para tener una lectura más comprensible. Los tabuladores se eliminan de la entrada en el caso de usar la redirección <code>&lt;&lt;-</code> y se mantienen con <code>&lt;&lt;</code>.</p>

Las redirecciones afectan normalmente a la E/S de un mandato o de una lista de mandatos, y pueden combinarse, teniendo en cuenta que siempre se procesan de izquierda a derecha. El formato genérico es:

```
[ Mandato ] Redirección ...
```

### 3.2. Tuberías.

La tubería es una herramienta que permite utilizar la salida de un programa como entrada de otro, por lo que suele usarse en el filtrado y depuración de la información, siendo una de las herramientas más potentes de la programación con intérpretes Unix.

Pueden combinarse más de una tubería en la misma línea de órdenes, usando el siguiente formato:

```
Mandato1 | Mandato2 ...
```

Todos los dialectos Unix incluyen gran variedad de filtros de información. La siguiente tabla muestra algunos de los más utilizados.

Mandato	Descripción
<b>head</b>	Corta las primeras líneas de un fichero.
<b>tail</b>	Extrae las últimas líneas de un fichero.
<b>grep</b>	Muestra las líneas que contienen una determinada cadena de caracteres o cumplen un cierto patrón.
<b>cut</b>	Corta columnas agrupadas por campos o caracteres.



<b>uniq</b>	Muestra o quita las líneas repetidas.
<b>sort</b>	Lista el contenido del fichero ordenado alfabética o numéricamente.
<b>wc</b>	Cuenta líneas, palabras y caracteres de ficheros.
<b>find</b>	Busca ficheros que cumplan ciertas condiciones y posibilita ejecutar operaciones con los archivos localizados
<b>sed</b>	Edita automáticamente un fichero.
<b>awk</b>	Procesa el fichero de entrada según las reglas de dicho lenguaje.

### 3.3. Variables.

Al contrario que en otros lenguajes de programación, BASH no hace distinción en los tipos de datos de las variables; son esencialmente cadenas de caracteres, aunque –según el contexto– también pueden usarse con operadores de números enteros y condicionales. Esta filosofía de trabajo permite una mayor flexibilidad en la programación de guiones, pero también puede provocar errores difíciles de depurar <sup>[5]</sup>.

Una variable BASH se define o actualiza mediante una operaciones de asignación, mientras que se hace referencia a su valor utilizando el símbolo del dólar delante de su nombre.

#### 3.3.1. Tipos de variables.

Las variables del intérprete BASH pueden considerarse desde los siguientes puntos de vista:

- Las **variables locales** son definidas por el usuario y se utilizan únicamente dentro de un bloque de código o de una función determinada.
- Las **variables de entorno** son las que afectan al comportamiento del intérprete y al de la interfaz del usuario. Al igual que cualquier proceso, la *shell* mantiene un conjunto de variables que informan sobre el contexto de su propio proceso. El usuario –o un guión– puede actualizar y añadir variables exportando sus valores al entorno del intérprete (mandato **export**), lo que afectará también a todos los procesos hijos generados por ella. El administrador puede definir variables de entorno estáticas para los usuarios del sistema (como, por ejemplo, en el caso de la variable `IFS`).
- Los **parámetros de posición** son los recibidos en la ejecución de cualquier programa o función, y hacen referencia a su orden ocupado en la línea de mandatos; así, el primer parámetro se denota por la variable `$1` y el noveno por `$9`. El mandato interno **shift** permite desplazar la lista de parámetros hacia la izquierda para procesar los parámetros con un orden superior a 9. El nombre del programa se denota por la variable `$0`.

- Las **variables especiales** son aquellas que tienen una sintaxis especial y que hacen referencia a valores internos del proceso. Los parámetros de posición pueden incluirse en esta categoría.

La siguiente tabla presenta las principales variables especiales y de entorno.

Variable de entorno	Descripción	Valor por omisión
<b>SHELL</b>	Camino del programa intérprete de mandatos	La propia <i>shell</i> .
<b>PWD</b>	Directorio de trabajo actual	Lo establece la <i>shell</i> .
<b>PPID</b>	Identificador del proceso padre (PPID)	Lo establece la <i>shell</i>
<b>IFS</b>	Separador de campos de entrada (debe ser de sólo lectura)	ESP, TAB, NL
<b>HOME</b>	Directorio personal de la cuenta	Lo define <b>root</b> .
<b>LOGNAME</b>	Nombre de usuario que ejecuta la <i>shell</i>	Activado por <b>login</b>
<b>PATH</b>	Camino de búsqueda de mandatos	Según el sistema
<b>LANG</b>	Idioma para los mensajes	
<b>EDITOR</b>	Editor usado por defecto	
<b>TERM</b>	Tipo de terminal	
<b>PS1 ... PS4</b>	Puntos indicativos primario, secundario, selectivo y de errores	
Variable especial	Descripción	
<b>\$\$</b>	Identificador del proceso (PID)	
<b>\$0</b>	Nombre del programa	
<b>\$1 ... \$9</b>	Parámetros posicionales recibidos por el programa	
<b>\$*</b>	Contenido completo de los parámetros recibidos por el programa	
<b>\$#</b>	Número de parámetros	
<b>\$?</b>	Código de retorno del último mandato (0=normal, >0=error)	
<b>\$!</b>	Último identificador de proceso ejecutado en segundo plano	
<b>\$_</b>	Valor de la última variable utilizada.	

### 3.4. Expresiones.

El intérprete BASH permite utilizar una gran variedad de expresiones en el desarrollo de programas y en la línea de mandatos.

La siguiente tabla presenta los operadores utilizadas en los distintos tipos de expresiones BASH (aritméticas, comparativas, de cadenas y de ficheros).

Operadores aritméticos:	+ - * / % ++ --
Operadores de comparación:	== != < <= > >= -eq -nt -lt -le -gt -ge
Operadores lógicos:	! &&
Operadores binarios:	&   ^ << >>
Operadores de asignación:	= *= /= %= += -= <<= >>= &= ^=  =
Operadores de tipos de ficheros:	-e -b -c -d -f -h -L -p -s -t
Operadores de permisos:	-r -w -x -g -u -k -O -G -N
Operadores de fechas:	-nt -ot -et
Operadores de cadenas:	-o -z -n

### 3.5. Entrecomillado.

BASH utiliza algunos caracteres especiales para representar comodines (\* ? [ ]), metacaracteres (; \ |) y operadores.

El entrecomillado se utiliza para modificar el tratamiento normal, tanto de los caracteres especiales, como de palabras reservadas y parámetros. Los tipos de entrecomillado se describen en la siguiente tabla.

Entrecomillado	Descripción
\Carácter	Carácter de escape o carácter literal.
'Cadena'	Cadena literal de caracteres (a excepción de \).
"Cadena"	Cadena literal con sustitución de variables (a excepción de ` ` " \).
`Mandato`	Permite asignar o evaluar la salida normal del mandato ejecutado.

### 3.6. Programación estructurada.

Las estructuras de programación se utilizan para generar un código más legible y fiable. Son válidas para englobar bloques de código que cumplen un cometido común, para realizar comparaciones, selecciones, bucles repetitivos y llamadas a subprogramas.

No se pretende dar una guía completa sobre programación estructurada, si no una breve descripción de las posibilidades ofrecidos por BASH.

#### 3.6.1. Listas de mandatos.

Los mandatos BASH pueden agruparse en bloques de código que mantienen un mismo ámbito de ejecución. La siguiente tabla describe los aspectos fundamentales de cada lista de órdenes.

Lista de órdenes	Descripción
<i>Mandato &amp;</i>	Ejecuta el mandato en 2º plano. El proceso tendrá menor prioridad y no debe ser interactivo..
<i>Man1   Man2</i>	Tubería. Redirige la salida de la primera orden a la entrada de la segundo. Cada mandato es un proceso separado.
<i>Man1 ; Man2</i>	Ejecuta varios mandatos en la misma línea de código.
<i>Man1 &amp;&amp; Man2</i>	Ejecuta la 2ª orden si la 1ª lo hace con éxito (su estado de salida es 0).
<i>Man1    Man2</i>	Ejecuta la 2ª orden si falla la ejecución de la anterior (su código de salida no es 0).
<i>(Lista)</i>	Ejecuta la lista de órdenes en un subproceso con un entorno común.
<i>( Lista; )</i>	Bloque de código ejecutado en el propio intérprete.
<i>Submand1 \</i> <i>Submand2</i>	Posibilita escribir listas de órdenes en más de una línea de pantalla. Se utiliza para ejecutar mandatos largos.

#### 3.6.2. Estructuras condicionales y selectivas.

La estructura condicional sirve para comprobar si se ejecuta un bloque de código cuando se cumple una cierta condición. Pueden anidarse varias estructuras dentro del mismo bloques de código.

La estructura selectiva evalúa la condición de control y, dependiendo del resultado, ejecuta un bloque de código determinado.

La tabla muestra cómo se representan ambas estructuras en BASH.

Estructura de programación	Descripción
<code>if Expresión; then Bloque; fi</code>	<b>Estructura condicional simple:</b> se ejecuta la lista de órdenes si se cumple la expresión. En caso contrario, este código se ignora.
<code>if Expresión1; then Bloque1; [ elif Expresión2;   then Bloque2;   ... ] [else Bloque3; ] fi</code>	<b>Estructura condicional compleja:</b> el formato completo condicional permite anidar varias órdenes, además de poder ejecutar distintos bloques de código, tanto si la condición de la expresión es cierta, como si es falsa.
<code>case Variable in   Patrón1) Bloque1 ;;   ... esac</code>	<b>Estructura selectiva múltiple:</b> si la variable cumple un determinado patrón, se ejecuta el bloque de código correspondiente.

### 3.6.3. Bucles.

Los bucles son estructuras de código que se ejecutan repetitivamente. Un bucle debe tener siempre una condición de salida para evitar errores provocados por bucles infinitos.

La tabla describe cada uno de los bucles interpretados por BASH.

Bucle	Descripción
<code>for Var in Lista; do Bloque done</code>	<b>Bucle iterativo:</b> se ejecuta el bloque de mandatos del bucle sustituyendo la variable de evaluación por cada una de las palabras incluidas en la lista.
<code>while Expresión; do Bloque done</code>	<b>Bucle iterativo "mientras":</b> se ejecuta el bloque de órdenes mientras que la condición sea cierta. La expresión de evaluación debe ser modificada en algún momento del bucle para poder salir.
<code>until Expresión; do Bloque done</code>	<b>Bucle iterativo "hasta":</b> se ejecuta el bloque de código hasta que la condición sea cierta. La expresión de evaluación debe ser modificada en algún momento del bucle para poder salir.

<b>select</b> <i>Var in Lista;</i> <b>do</b> <i>Bloque1</i> ... <b>done</b>	<b>Bucle de selección interactiva:</b> se presenta un menú de selección y se ejecuta el bloque de código correspondiente a la opción elegida. El bucle se termina cuando se ejecuta una orden <b>break</b> .  La variable <b>PS3</b> se usa como punto indicativo.
<b>break</b>	<b>Ruptura inmediata de un bucle</b> (debe evitarse en programación estructurada para impedir errores de lectura del código).
<b>continue</b>	<b>Salto a la condición del bucle</b> (debe evitarse en programación estructurada para impedir errores de lectura del código).

### 3.6.4. Funciones.

Una función en BASH es una porción de código declarada al principio del programa, que puede recoger parámetro de entrada y que puede ser llamada desde cualquier punto del programa principal o de otra función.

Los parámetros recibidos por la función se tratan dentro de ella del mismo modo que los del programa principal, o sea con las variables internas \$0 a \$9.

El siguiente cuadro muestra los formatos de declaración y de llamada de una función.

```
[function] NombreFunción ( )
{
    Bloque
    ...
    [ return [Valor] ]
    ...
}
...
NombreFunción [ Parámetro ... ]
```

## 4. Sistemas de archivos.

La gestión adecuada del acceso a disco es otro de los aspectos importantes en el proceso de administración de sistemas operativos multiusuario y multitarea y es imprescindible mantener una estructura básica con un cierto nivel organizativo. El sistema operativo interactúa con los usuarios y las aplicaciones, y se hace necesario un modelo de seguridad dependiente de la forma en que se almacenan los ficheros en los dispositivos.

Un sistema de archivos puede verse desde dos categorías lógicas de ficheros <sup>[2]</sup>:

- Archivos compartidos con otras máquinas o privados.
- Archivos variables o estáticos.

Por lo tanto, un **sistema de archivos** es un subárbol de directorios con un directorio raíz –que debe tener unos permisos acorde con las necesidades de acceso a sus archivos–, una estructura lógica de almacenamiento y un punto de montaje adecuado en el árbol de directorios global del servidor.

### 4.1. Normas para la Jerarquía de Sistemas de Archivos (FHS).

Las **Normas para la Jerarquía de Sistemas de Archivos (FHS)** describen un conjunto de reglas que definen nombre y localizaciones para la mayoría de ficheros y directorios en sistemas operativos Linux.

La siguiente tabla describe brevemente los subdirectorios de la jerarquía principal, ordenados alfabéticamente.

Subdirectorio	Descripción
/bin	Binarios básicos para todos los usuarios del sistema.
/boot	Ficheros estáticos del cargador de arranque.
/dev	Sus entradas representan los dispositivos del sistema (conviene recordar que <i>"en Unix todo es un archivo"</i> ).
/etc	Configuración local de la máquina.
/home	Cuentas de usuarios (debe definirse como sistema de archivos independiente).
/lib	Bibliotecas compartidas del sistema y módulos fundamentales del núcleo.
/mnt	Puntos de montaje para sistemas de archivos temporales (disquete, CD-ROM, etc.).
/opt	Área compartida para paquetes de grandes aplicaciones (puede ser un sistema de archivos independiente con una jerarquía propia).
/proc	Sistema de archivos virtual con información sobre procesos y el núcleo.

<code>/root</code>	Cuenta del usuario administrador <code>root</code> .
<code>/sbin</code>	Binarios del sistema.
<code>/tmp</code>	Zona compartida para ficheros temporales.
<code>/usr</code>	Jerarquía secundaria con información que puede ser compartida por otros ordenadores, con acceso de sólo lectura (debe ser un sistema de archivos independiente en servidores y tiene una jerarquía básica similar a la principal).
<code>/usr/local</code>	Jerarquía para programas locales (debe ser un sistema de archivos independiente).
<code>/var</code>	Información variable, incluyendo ficheros históricos, de estado, de bloqueos, de recuperación, de colas de trabajos, etc.

## 4.2. Discos y particiones.

Red Hat Linux –como el resto de sistemas Unix– utiliza ficheros de dispositivos para acceder a los recursos de la máquina. Sin embargo, cada dialecto Unix tiene una notación diferente para identificar cada dispositivo de almacenamiento. Red Hat almacena dichos ficheros en el directorio `/dev`, con el siguiente formato:

- Tipo de dispositivo (`sd` para SCSI, `hd` para IDE).
- Unidad (`a` para el dispositivo 1, `b` para el 2, etc.).
- Número de partición.

Una **partición** es cada una de las subdivisiones que el gestor del sistema define en una unidad de disco del sistema, donde se almacena un determinado sistema de archivos o un espacio de paginación.

Siguiendo las normas descritas en el apartado anterior, el administrador debe definir los distintos sistemas de archivos de su sistema, creando particiones en cada disco, teniendo en cuenta los recursos disponibles y la utilización principal que los usuarios harán de ellos.

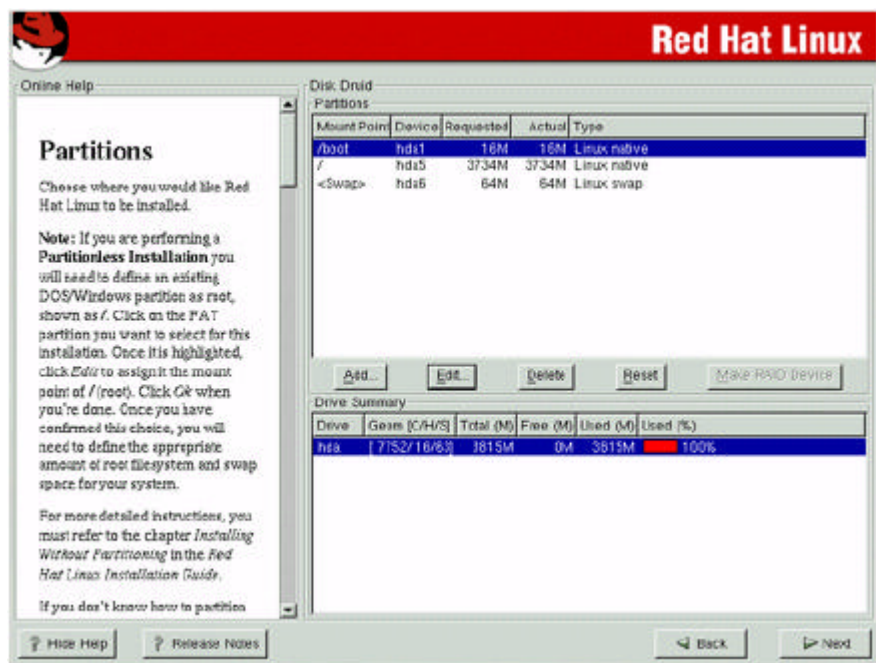
El proceso de crear los sistemas de archivos básicos suele realizarse durante la instalación de la máquina, aunque pueden añadirse y ampliarse posteriormente. La siguiente tabla define una distribución típica e indica algunas recomendaciones.

Sistema de archivos	Recomendaciones
<code>/</code>	Es necesario para trabajar, no debe ser de gran tamaño.
<code>/proc</code>	Es necesario para trabajar y debe ocupar el doble de la memoria física, al menos.
<code>/usr</code>	Contiene el sistema operativo, su tamaño depende de los paquetes que deban instalarse y de las previsiones de ampliación.



<code>/tmp</code>	Espacio para ficheros temporales, depende del número de usuarios que se definirán y del espacio estimado para sus trabajos.
<code>/var</code>	Contará con ficheros que crecen, debe preverse un tamaño suficiente, pero sin desperdiciar espacio de disco.
<code>/home</code>	Cuentas de usuarios; puede ser recomendable usar un disco independiente, cuyo tamaño dependerá del número de usuarios y de la capacidad estimada de sus cuentas.
<code>/usr/local</code>	Debe tener un tamaño suficientemente grande para almacenar las utilidades y aplicaciones instaladas; es recomendable usar un disco independiente.

El siguiente grafico describe la utilidad de creación de sistemas de archivos `disk druid`, usada durante la instalación de un nuevo servidor.



### 4.3. Sistemas de archivos de Linux (ext2 y ext3).

Linux soporta el montaje de distintos sistemas de archivos, tanto locales como remotos, ya que se ha programado una interfaz entre ellos y el núcleo, conocida como **Sistema de Archivos Virtual (VFS)**.

Las siguientes tablas describen el formato del fichero de configuración `/etc/fstab` y los mandatos más habituales en la gestión de sistemas de archivos.

Formato	Descripción
<b>/etc/fstab</b>	
<i>Etiqu Montaje Tipo Opciones Volc NOrden</i> ...	Fichero principal de descripción de sistemas de archivos. Sus campos son: <ol style="list-style-type: none"> <li>1. Etiqueta de la partición o directorio remoto.</li> <li>2. Punto de montaje local.</li> <li>3. Tipo de sistema de archivos (<i>ext2</i>, <i>ext3</i>, <i>swap</i>, <i>vfat</i>, <i>nfs</i>, <i>cifs</i>, etc.).</li> <li>4. Opciones de montaje (dependen de cada tipo de sistema de archivos).</li> <li>5. Control de volcado automático de seguridad ante caídas del sistema.</li> <li>6. Orden de comprobación (1 para <i>/</i>, incrementar en sistemas de archivos de distintos discos).</li> </ol>

Mandato	Descripción
<b>fdisk</b>	Gestor de discos usado para definir particiones.
<b>mkfs</b>	Formatea una nueva partición.
<b>mount</b>	Monta un sistema de archivos en el árbol global de directorios.
<b>umount</b>	Desmonta un sistema de archivos.
<b>tune2fs</b>	Conversor entre sistemas de archivos <b>ext2</b> y <b>ext3</b> .

El sistema de archivos más utilizado hasta hace poco tiempo en Linux era el conocido como **Sistema de Archivos Extendido 2 (ext2)**, que mejoraba las prestaciones de la primera versión, pero que seguía presentando problemas ante una caída inesperada del sistema, ya que necesitaba un largo proceso de comprobación y corrección.

A partir de la versión 7.2 de Red Hat se está utilizando el **Sistema de Archivos Extendido 3 (ext3)**, que incluye las siguientes mejoras:

- El diario de registros es la característica más importante, que mejora los procesos de revisión de integridad, ya que sólo se requiere la comprobación de dicho diario.
- Soporta mayores niveles de integridad de datos para evitar la corrupción del sistema de archivos, permitiendo elegir el tipo y el nivel de protección.
- Mayor flujo y mayor velocidad de accesos repetidos a datos.
- Fácil transición entre **ext2** y **ext3**, sin necesidad de volver a formatear las particiones.

## 4.4. Paginación y procesos.

### 4.4.1. Espacios de paginación.

Un sistema operativo multiusuario y multitarea como Linux necesita una gran cantidad de memoria física para poder ejecutar todos los procesos. Los **espacios de paginación** son particiones de disco que permiten ampliar virtualmente la memoria del sistema, guardando el estado de los procesos que en un determinado momento están a la espera de ser ejecutados, si la memoria física está agotada.

Los factores principales que deben determinar el tamaño del espacio total de paginación son:

- La cantidad de memoria y de disco del sistema.
- El número de usuarios que tendrán acceso a la máquina.
- El número previsto de procesos/usuario.
- El número de servicios activos en el sistema.
- El número estimado de clientes/servicio.

Como regla general suele usarse el doble de la memoria física instalada. Ante casos de necesidad, el administrador puede ampliar la cantidad de paginación usando ficheros de disco que pueden ser posteriormente eliminados <sup>[3]</sup>.

La siguiente tabla describe las órdenes Linux usadas para manipular los espacios de paginación.

Mandato	Descripción
<b>fdisk</b>	Gestor de discos usado para definir particiones.
<b>mkswap</b>	Crea particiones o ficheros de paginación.
<b>swapon</b>	Activa una partición o un fichero de paginación.
<b>swapoff</b>	Desactiva una partición o un fichero de paginación.

### 4.4.2. Sistema de archivos virtual /proc.

El árbol de directorios montado en `/proc` es un sistema de archivos virtual almacenado en memoria, que contiene una jerarquía de ficheros especiales que mantienen el estado actual del núcleo del sistema Linux, recopilando información sobre los dispositivos y los procesos en ejecución

La mayoría de los ficheros virtuales de `/proc` aparecen con longitud 0, aunque pueden ser revisados como si fueran archivos de texto con gran cantidad de información <sup>[2]</sup>.

En `/proc` hay una serie de subdirectorios especiales que representan el estado actual de cada proceso en ejecución –denotados por el identificador de cada proceso (PID)–, que incluyen datos como: la línea de la orden ejecutada, los directorio raíz y de trabajo del proceso, estados de la memoria, de ejecución y de uso de los procesadores, las variables de entorno, etc.

Otros ficheros y directorios de interés son los que informan sobre procesadores, memoria, controladores, dispositivos, interrupciones, particiones, puntos de montaje, módulos del núcleo, parámetros de la red, etc.

El directorio especial `/proc/sys` contiene ficheros que pueden ser modificados por el administrador para realizar cambios de configuración en el núcleo, habilitando o desactivando ciertas características operativas. **Debe tenerse gran precaución en la modificación de los archivos virtuales de `/proc/sys`.**

La siguiente tabla describe los subdirectorios de `/proc/sys`.

Directorio	Descripción
<code>/proc/sys/dev</code>	Información sobre dispositivos especiales (CD-ROM, discos RAID, etc.).
<code>/proc/sys/fs</code>	Parámetros de sistemas de archivos (límites de ficheros e inodos abiertos, cuotas, etc.).
<code>/proc/sys/kernel</code>	Configuración del núcleo (contabilidad de procesos, nombre del sistema, parada por consola, módulos, colas de mensajes, etc.).

La información completa sobre el sistema de archivos virtual `/proc` puede encontrarse en el paquete del código fuente del núcleo, en el subdirectorio `/usr/src/VersiónNúcleo/Documentation`.

## 4.5. Discos redundantes (RAID).

La **Matriz Redundante de Discos Independientes (RAID)** representa un conjunto de técnicas válidas para ahorrar costes y mejorar las prestaciones y la seguridad del acceso al almacenamiento masivo, combinando múltiples discos en un único dispositivo lógico<sup>[2]</sup>.

El concepto principal de RAID es dividir los datos en ciertos trozos y distribuirlos en los dispositivos de la matriz, según el nivel de necesidad. Durante el proceso de lectura se sigue un algoritmo inverso de reconstrucción.

Las principales características del uso de discos en RAID son:

- Aumentar la velocidad de acceso a los datos.
- Incrementar la capacidad de almacenamiento, combinando discos de menor capacidad en un único disco lógico mayor.
- Mejorar la tolerancia a fallos de los discos.

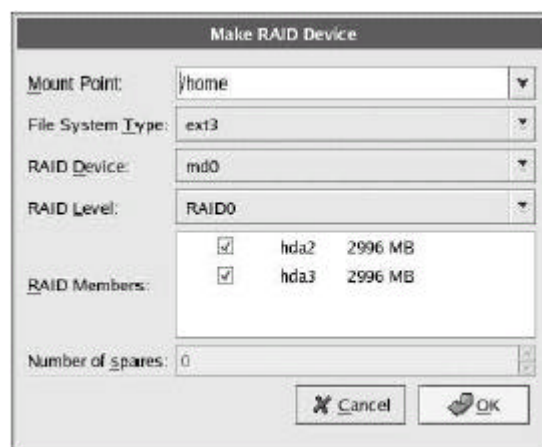
Existen dispositivos y controladores preparados para realizar técnicas RAID en su propio *hardware*, lo que aumenta las prestaciones y el precio final de la máquina. Los nuevos sistemas operativos son aptos para realizar este cometido bajo *software*.

La siguiente tabla describe los niveles RAID más usados.

<b>RAID 0:</b>	Los datos se dividen en bandas, escribiendo cada una de ellas en un disco. Se mejoran las prestaciones de acceso. La capacidad total es la suma de las capacidades de cada disco.
<b>RAID 1:</b>	Los datos se almacenan en espejo, repitiendo la misma escritura en cada disco. Se incrementa la seguridad y la tolerancia a fallos del sistema, porque puede sustituirse un disco defectuoso sin afectar al funcionamiento de la máquina. La capacidad total corresponde a la de cualquier disco (todos deben ser iguales).
<b>RAID 5:</b>	Se usan más de 2 discos para distribuir los trozos de datos y sus paridades. Cada disco contiene una banda de datos y la paridad de las bandas de otros datos. Se incrementan la seguridad, las prestaciones y los costes. La capacidad total es la suma total de la capacidad de los discos menos 1.
<b>RAID lineal:</b>	Los discos se agrupan secuencialmente para formar un disco lógico mayor. No se incrementan las prestaciones ni la seguridad, sólo la capacidad.

Las particiones que vayan a utilizarse en las matrices RAID deben definirse durante el proceso de instalación del sistema operativo. La utilidad `disk druid` permite definir particiones RAID, asociarles el nivel de redundancia y generar el disco lógico (**metadispositivo**).

El ejemplo del gráfico crea un metadispositivo `md0` de tipo RAID 0, formado por las particiones `hda2` y `hda3`, que contendrá un sistema de archivos de tipo Ext3 montado sobre el directorio `/home` <sup>[3]</sup>.



La definición de la matriz se encuentra en el fichero de configuración `/etc/raidtab`.

## 4.6. Volúmenes lógicos.

Los volúmenes lógicos son técnicas de gestión de almacenamiento desarrolladas a partir de la versión 8.0 de Red Hat –heredadas del sistema operativo AIX, el dialecto Unix de IBM– que permiten redimensionar las particiones y distribuirlas en varios discos.

La única restricción impuesta por el **Gestor de Volúmenes Lógicos (LVM)** de Red Hat 8 está en que el directorio `/boot` debe encontrarse en una partición real y no formar parte de ningún volumen lógico <sup>[3]</sup>.

Los volúmenes lógicos se definen en el proceso de instalación del sistema operativo y constan de 3 elementos fundamentales:

<b>Volumen físico:</b>	estructura que representa a un disco físico.
<b>Volumen lógico:</b>	estructura equivalente a un sistema de archivos Linux.
<b>Grupo de volúmenes:</b>	conjunto de varios volúmenes lógicos que pueden almacenarse en varios volúmenes físicos. Así, un disco puede contener varios sistemas de archivos y un sistema de archivos puede estar grabado en varios discos.

El instalador del sistema debe seguir los siguientes pasos:

- Crear una partición normal para el directorio `/boot`, ya sea incluido en el directorio raíz o en una partición propia.
- Definir un volumen físico en cada disco.
- Crear los grupos de volúmenes conjuntando adecuadamente los volúmenes físicos.
- Definir los volúmenes lógicos de cada grupo de volúmenes, asignando para cada uno de ellos su tamaño inicial y su punto de montaje.

Es recomendable dejar algún espacio sin asignar para ampliar las particiones que lo necesiten.

## 4.7. Sistemas de archivos remotos.

La conexión remota a otros ordenadores supone una gran ventaja en el proceso de compartir información. Los sistemas de archivos remotos permiten almacenar la información en un único nodo central y hacerla accesible a los distintos clientes, posibilitando la movilidad del usuario.

Para finalizar este capítulo van a describirse los sistemas de archivos remotos más utilizados actualmente.

### 4.7.1. NFS.

El **Sistema de Archivos en Red (NFS)** fue creado por Sun Microsystems para SunOS –su dialecto Unix–, usando las técnicas de **Llamadas a Procedimientos Remotos (RPC)**. NFS permite acceder a los archivos en nodos remotos exactamente en la misma manera que si fueran locales, de un modo completamente transparente al cliente e independientemente de la arquitecturas del servidor <sup>[6]</sup>.

La siguiente tabla describe los servicios que deben activarse en los ordenadores servidor y cliente NFS.

Servicio	Descripción
<code>portmap</code>	Servicio de control principal de RPC.
<code>rpc.mountd</code>	Control de montaje del cliente NFS.
<code>rpc.nfsd</code>	Servidor NFS.
<code>rpc.statd</code>	Monitor del Estado de la Red (NSM), que notifica el reinicio del servidor NFS.
<code>rpc.rquotad</code>	Provee información de cuotas para usuarios remotos.

El fichero `/etc/exports` se utiliza para configurar NFS en el servidor. La siguiente tabla describe el formato de las líneas de dicho fichero, una para cada directorio exportado.

Formato	Descripción
<b><code>/etc/exports</code></b>	
<i>Directorio Cliente(Opciones) ...</i> ...	Fichero principal que describe los directorios que pueden exportarse por NFS. Sus campos son: <ol style="list-style-type: none"><li>Directorio local a exportar.</li><li>Nombre o IP del cliente (soporta comodines en nombre y en dominios).</li><li>Opciones de exportación: sólo lectura (<code>ro</code>), lectura/escritura (<code>rw</code>), evitar acceso privilegiado para el <b>root</b> del cliente (<code>root_squash</code>), acceso privilegiado para <b>root</b> (<code>no_root_squash</code>), etc.</li></ol>

El cliente NFS puede configurar la importación de directorios en su fichero `/etc/fstab` o montarlo directamente con la orden **mount**.

```
mount -t nfs Servidor:Directorio PuntoMontaje [Opciones]
```

## 4.7.2. CIFS.

El **Sistema de Archivos Común para Internet (CIFS)** provee una serie de mecanismos abiertos e independientes de la plataforma utilizada, para que sistemas clientes soliciten servicios de ficheros a otras máquinas a través de la red. Este protocolo es la evolución del conocido como **Bloque de Mensajes del Servidor (SMB)**, usado principalmente por ordenadores con Windows <sup>[14]</sup>.

Las características principales de CIFS son:

- Acceso a ficheros, permitiendo compartir información en lectura y escritura.
- Acceso bloqueado y desbloqueado tanto a ficheros como a registros.
- Notificación de cambios en ficheros y directorios.
- Inclusión de atributos extendidos.
- Independencia del protocolo de resolución de nombres.

Las **Extensiones de CIFS para UNIX** son normas de reciente creación y sólo están implementadas en las últimas versiones de los servicios de ficheros. Los núcleos de Linux con versión 2.2 y 2.4 necesitan ser recompilados o generar un módulo propio para la gestión de clientes CIFS, aunque soportan el montaje de sistemas de archivos SMBFS.

El servidor de ficheros puede ser una máquina con sistema operativo Windows NT/2000/XP o con Linux y el servicio **Samba** activado. En ambos casos, deben ser configurados los recursos que van a ser exportados.

Los paquetes que deben instalarse en un ordenador Red Hat se describen a continuación.

Paquete	Descripción
samba-common	Aplicaciones básicas de Samba comunes a cliente y servidor.
samba-server	Servidor de ficheros.
samba-client	Clientes Samba para acceder a recursos remotos y montaje de sistemas de archivos SMBFS y CIFS.

El ordenador cliente debe instalar los paquetes `samba-common` y `samba-client` para acceder a los recursos del servidor a través del protocolo SMB. Si se desea usar el protocolo actualizado CIFS y montar sistemas de archivos de forma equivalente a NFS, el administrador debe acceder al paquete con el código fuente de Samba y compilar el módulo `cifs` correspondiente con su versión del núcleo e instalar el fichero del código objeto en el directorio correspondiente a los módulos para gestión de sistemas de archivos (normalmente `/lib/modules/VersiónNúcleo/kernel/fs`).

La próxima tabla muestra los mandatos usados por el cliente Samba.



Mandato	Descripción
<b>smbclient</b>	Cliente Samba con interfaz similar al cliente FTP.
<b>smbpasswd</b>	Permite cambiar la clave remota del usuario..
<b>smbsh</b>	Ejecuta un intérprete de mandatos Unix sobre un recurso de Windows NT.
<b>smbmount</b>	Montador de sistemas de archivos SMBFS.
<b>mount.cifs</b>	Montador de sistemas de archivos CIFS.
<b>umount</b>	Desmontador general de sistemas de archivos.

El montaje de un sistema de archivos CIFS requiere autenticación mediante usuario y clave. El método más seguro es indicar en la orden de montaje un archivo donde se incluyan las credenciales del usuario, con el siguiente el formato:

```
Username = UsuarioRemoto
Password = Clave
```

El formato para montar un sistema de archivos CIFS es el siguiente:

```
mount -t cifs //Servidor/Recurso PuntoMontaje \
-o credentials=FichCredenciales[,Opción=Valor,...]
```

## 5. Arranque y servicios.

### 5.1. Proceso de arranque.

Durante el proceso de arranque de la máquina se realizan las comprobaciones necesarias para configurar y activar todos los servicios definidos por el administrador del sistema. Es fundamental conocer este procedimiento y preparar los cambios necesarios según la planificación realizada para dicho ordenador.

El proceso de arranque de un servidor Linux basado en arquitectura x86 (Intel, AMD) comprende los siguientes pasos <sup>[2]</sup>:

- Tras comprobar los dispositivos de la máquina, el BIOS ejecuta el primer paso del cargador del sistema, situado en el sector de arranque del primer disco duro. Los cargadores más usados en Red Hat son LILO y GRUB.
- El cargador de arranque ejecuta el segundo paso del proceso, situado en la partición `/boot`.
- El núcleo de Linux y sus módulos adicionales se cargan en memoria y se monta la partición raíz en modo de sólo lectura.
- El núcleo toma el control de la secuencia de arranque y ejecuta el proceso `/sbin/init`.
- Este programa de iniciación carga todos los servicios definidos, ejecuta los programas de iniciación y configuración y monta las particiones definidas. `/sbin/init` lee la información del fichero de configuración `/etc/inittab`, carga en primer lugar los servicios básicos y luego los asociados al nivel de arranque elegido por el administrador.
- Se finaliza el arranque del sistema presentando al usuario el proceso de conexión (`login`) o el entorno gráfico (normalmente GNOME o KDE).

Evidentemente, por motivos de seguridad, todos los ficheros y mandatos de configuración tienen que estar completamente vetados para los usuarios normales del servidor y sólo pueden ser ejecutados o modificados por el usuario `root`.

### 5.2. Cargadores de arranque.

Como se ha descrito en el apartado anterior, un programa cargador de arranque es el encargado de iniciar un sistema operativo.

Los principales cargadores usados por Linux soportan la definición de un menú de ejecución para los distintos sistemas operativos instalados en el ordenador e iniciarlos en determinados niveles de ejecución.

Aunque cada arquitectura de computadores utiliza cargadores diferentes, este capítulo se centra en los usados por Red Hat con procesadores de tipo x86.

Para evitar sorpresas desagradables, el responsable del sistema tiene que crear un disquete de arranque, que permita reiniciar el equipo en caso de producirse fallos en el cargador. Para ello, debe ejecutar la siguiente orden (se utiliza por defecto el fichero de dispositivo para la primera disquetera, `/dev/fd0`):

```
/sbin/mkbootdisk [--device=Disquetera] VersionNúcleo
```

### 5.2.1. GRUB.

El **Cargador de Arranque Unificado de GNU (GRUB)** permite al usuario elegir el sistema operativo y el núcleo con que desea trabajar.

GRUB tiene 2 modos de operación <sup>[2]</sup>:

- El **modo directo** se usa para cargar el núcleo de Red Hat sin ningún tipo de intermediarios.
- El **modo encadenado** se utiliza para cargar otros sistemas operativos y apunta al primer sector de arranque de la partición, donde se encuentran los ficheros de iniciación del sistema.

GRUB ofrece al usuario un entorno de operación válido para realizar configuraciones previas al inicio del sistema operativo, pudiendo acceder directamente a su fichero de control `/boot/grub/grub.conf` (actualmente GRUB sólo permite almacenar el directorio `/boot` en una partición de tipo **ext2**).

Las órdenes para la configuración de GRUB definen las características del menú de arranque, indicando los distintas opciones de carga de sistemas, la opción por omisión, límites temporales, entorno gráfico, etc.

El siguiente ejemplo presenta la configuración de un menú para arrancar Red Hat con un núcleo versión 2.4.18 en un disco SCSI, y Windows 2000 <sup>[2]</sup>.

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
# section to load linux
title Red Hat Linux (2.4.18-5.47)
    root (hd0,0)
    kernel /vmlinuz-2.4.18-5.47 ro root=/dev/sda2
    initrd /initrd-2.4.18-5.47.img
# section to load Windows 2000
title windows
    rootnoverify (hd0,0)
    chainloader +1
```

### 5.2.2. LILO.

El **Cargador de Linux (LILO)** sigue utilizándose como gestor de arranque para Red Hat, aunque GRUB es el que se está utilizando en la configuración por omisión.

Tanto LILO como GRUB tienen un comportamiento básico equivalente, aunque existen algunas diferencias <sup>[2]</sup>:

- LILO no cuenta con interfaz de mandatos interactivo.
- Guarda en el sector de arranque la información de las localizaciones de los núcleos de los sistemas operativos disponibles, lo que obliga a reinstalar LILO cuando se realiza alguna modificación en el menú de selección.
- LILO no puede leer a particiones **ext2**.

El fichero de configuración `/etc/lilo.conf` almacena la información que será instalada en el sector de arranque del disco principal.

El siguiente ejemplo describe una configuración de LILO que despliega un menú que permite arrancar Red Hat desde una partición extendida con núcleo 2.4.0 y MS-DOS desde la primera partición del disco.

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
lba32
default=linux
image=/boot/vmlinuz-2.4.0-0.43.6
    label=linux
    initrd=/boot/initrd-2.4.0-0.43.6.img
    read-only
    root=/dev/hda5
other=/dev/hda1
    label=dos
```

### 5.3. El Núcleo.

Los sistemas operativos Unix se basan en una estructura de capas, donde las capas internas prestan servicios básicos a las externas. El **Núcleo (kernel)** es la parte principal del sistema operativo, realiza las funciones básicas de control y presta los servicios esenciales para gestionar el sistema operativo.

El núcleo de Linux consta principalmente de los componentes descritos en la siguiente tabla <sup>[13]</sup>.

<b>Gestor de memoria:</b>	Encargado de asignar áreas de memoria y de espacios de paginación a los procesos, a los módulos del núcleo y al área de <i>cache</i> .
<b>Gestor de procesos:</b>	Parte esencial que crea, activa y termina los procesos; implementa las reglas de la multitarea.
<b>Controladores de dispositivos:</b>	Gestionan la comunicación del sistema con cada uno de los dispositivos conectados. El núcleo se configura en el proceso de arranque para cargar los módulos necesarios para controlar los dispositivos específicos de cada máquina.
<b>Gestor del sistema de archivos virtual:</b>	Capa intermedia que permite acceder uniformemente a los sistemas de archivos, manteniendo un árbol de directorios homogéneo.
<b>Gestor de redes:</b>	Capa abstracta para el acceso general a la red informática, independientemente del tipo de dispositivos usado y de la arquitectura de la red.

Siendo Linux un sistema operativo basado en el código abierto, el administrador puede rehacer el núcleo, incluyendo o eliminando características operativas, según sus necesidades. El proceso para recompilar el núcleo de Linux es cada vez más sencillo de realizar, ya que se configura mediante menús con una gran cantidad de opciones.

Es conveniente contar con un núcleo pequeño, pero que pueda tratar todas las funciones básicas del sistema. La gestión de los dispositivos y de las funciones adicionales puede ser controlada por los módulos del núcleo.

### 5.3.1. Módulos.

Los módulos del núcleo de Linux son objetos compilados en lenguaje C que controlan elementos o funciones específicas.

Los módulos básicos para el control de los dispositivos conectados se cargan en el proceso de arranque del sistema operativo. El resto de módulos que el gestor del sistema considera necesarios deben enumerarse en el fichero de configuración `/etc/modules.conf`, así como los parámetros opcionales que éstos requieran para obtener un funcionamiento óptimo del servidor

En `/etc/modules.conf` se especifican las asociaciones (alias) entre módulos reales y virtuales. Los módulos virtuales son las interfaces virtuales entre el sistema y los módulos que controlan dispositivos reales; son los encargados, por ejemplo, de gestionar la red, los sistemas de archivos especiales, las capacidades exclusivas de tarjetas de sonido, etc.

La siguiente tabla describe las órdenes del sistema implicadas en la gestión de los módulos del núcleo.

Mandato	Descripción
<b>lsmod</b>	Lista los módulos cargados.
<b>modprobe</b>	Prueba un determinado módulo del sistema e indica si puede ser instalado.
<b>insmod</b>	Instala un nuevo módulo.
<b>rmmod</b>	Desinstala un módulo cargado.

### 5.3.2. Parámetros de operación.

Las nuevas versiones de los núcleos de Linux soportan la configuración y personalización de sus parámetros de operación, que afectan a la conducta de sus componentes, tales como: incrementar el máximo número de ficheros abiertos (`fs.file-max`) o activar la capacidad de reenviar paquetes para crear cortafuegos (`net.ipv4.ip_forward`).

Las modificaciones en el entorno predefinido para el núcleo se verán reflejadas en el fichero correspondiente a la característica modificada, situado en el sistema de archivos virtual `/proc/sys`.

El fichero de configuración de los parámetros operativos del núcleo es `/etc/sysctl.conf` y puede ser editado por el responsable del sistema para incluir los cambios en el próximo arranque del sistema. Asimismo, las modificaciones se activan automáticamente ejecutando la orden `sysctl -p`.

La siguiente tabla muestra el formato de las líneas del fichero `/etc/sysctl.conf`.

Formato	Descripción
<b><code>/etc/sysctl.conf</code></b>	
<i>Componente[.Subcomp].Parámetro = Valor</i> ...	Configuración de los parámetros del núcleo. Los campos son: <ol style="list-style-type: none"> <li>1. Componente principal del núcleo: núcleo (<code>kernel</code>), memoria virtual (<code>vm</code>), sistema de archivos (<code>fs</code>) o red (<code>net</code>). Algunos de éstos puede contar con grupos (subdirectorios) de parámetros.</li> <li>2. Parámetro de operación.</li> <li>3. Valor asignado al parámetro.</li> </ol>

## 5.4. Niveles de arranque.

Los niveles de arranque sirven para que el sistema pueda operar de distintas maneras según las necesidades del administrador. Simplemente cambiando el nivel de arranque, el sistema operativo puede entrar en modo mantenimiento y posteriormente volver al modo multiusuario.

El Linux de Red Hat utiliza un proceso de arranque heredado de Unix System V, conocido como *SysV Init*. La siguiente tabla describe los niveles de ejecución soportados por estos sistemas.

Nivel	Descripción
0	Parada del sistema.
1 o S	Nivel de mantenimiento para usuario privilegiado (monousuario).
2	Definido por el administrador en Linux, multiusuario sin NFS en Solaris y AIX.
3	Nivel de multiusuario.
4	Definido por el administrador o no usado.
5	Nivel de multiusuario con entorno gráfico.
6	Rearranque del sistema.

El nivel de arranque usado por defecto en el sistema se define en el fichero de configuración `/etc/inittab`. Sin embargo, la orden `init` permite modificar el nivel de ejecución de la máquina en cualquier momento.

El mandato `init` lee los guiones de configuración almacenados en el subdirectorio `/etc/rc.d/rcN.d`, correspondiente al nivel de ejecución seleccionado. El modo de operación es el siguiente:

- Parar en orden de secuencia los procesos correspondientes a los ficheros `KNNservicio`, siendo `NN` el orden de la secuencia (2 dígitos) para dicho `servicio` (se ejecuta: `/etc/rc.d/rcN.d/KNNservicio stop`).
- Arrancar en orden de secuencia los procesos correspondientes a los ficheros `SNNservicio` (se ejecuta: `/etc/rc.d/rcN.d/SNNservicio start`).

Los guiones de ejecución de servicios se encuentran normalmente en el directorio `/etc/init.d` y están enlazados simbólicamente a los *scripts* de cada nivel de ejecución. De esta manera, el administrador puede arrancar o parar servicios independientemente, ejecutando:

```
/sbin/service Servicio { start|stop|restart }
```

Los mandatos `ntsysv` (modo texto) y `redhat-config-services` (modo gráfico) definen los servicios que se iniciarán en los niveles de ejecución para multiusuario, que serán descritos al final de este capítulo.

## 5.5. Configuración del sistema.

El directorio `/etc/sysconfig` contiene los ficheros y guiones de configuración de los dispositivos del sistema. En la siguiente tabla se describen los elementos más importantes de este directorio, ordenados alfabéticamente <sup>[2]</sup>.

Elemento	Descripción
<code>apmd</code>	Configuración del control de ahorro de energía.
<code>authconfig</code>	Configuración del programa de autenticación, indicando si se usa codificación MD5, acceso Kerberos, LDAP, NIS, etc.
<code>clock</code>	Control del reloj y huso horario.
<code>hwconf</code>	Lista los dispositivos detectados en el arranque de la máquina.
<code>i18n</code>	Define el idioma.
<code>keyboard</code>	Especifica el tipo de teclado.
<code>mouse</code>	Indica el tipo de ratón.
<code>network</code>	Configuración básica de la red.
<code>network-scripts</code>	Directorio con los guiones de arranque y parada de los controladores de acceso a la red.
<code>soundcard</code>	Indica los parámetros de la tarjeta de sonido.

## 5.6. Servicios.

Los servicios son programas cargados en un determinado nivel de ejecución, que suministran al usuario ciertas utilidades o beneficios. Cada servicio supone una posibilidad de conexión con la máquina, lo que implica la posibilidad de sufrir ataques contra la seguridad del sistema.

**El administrador sólo debe activar los servicios estrictamente necesarios para su máquina.**

La siguiente tabla describe los más usados en el Linux de Red Hat.



Servicio	Descripción
<b>apmd</b>	Control de ahorro de energía.
<b>crond</b>	Ejecución cronológica de programas.
<b>cups</b>	Servidor de impresión mediante protocolo IPP.
<b>dhcp</b>	Servicio DHCP para la asignación remota de parámetros de la red.
<b>httpd</b>	Servidor Apache para suministrar acceso a páginas <i>web</i> .
<b>ldap</b>	Servicio de directorios LDAP.
<b>nfs</b>	Acceso remoto a directorios mediante NFS.
<b>smb</b>	Servicio para compartir ficheros y recursos, compatible con la red de Windows.
<b>ssh</b>	Conexión segura.
<b>syslogd</b>	Registro de anotaciones e incidencias.
<b>wu-ftp</b>	Servicio FTP para transferencia de ficheros.
<b>xinetd</b>	Metaservicio de red.

El gráfico describe **redhat-config-services**, la herramienta usada para la configuración de servicios en Red Hat 8.0.



## 6. Instalación de programas.

Otra tarea importante en la administración de sistemas es la instalación de nuevas aplicaciones y la actualización de las que existen en el servidor.

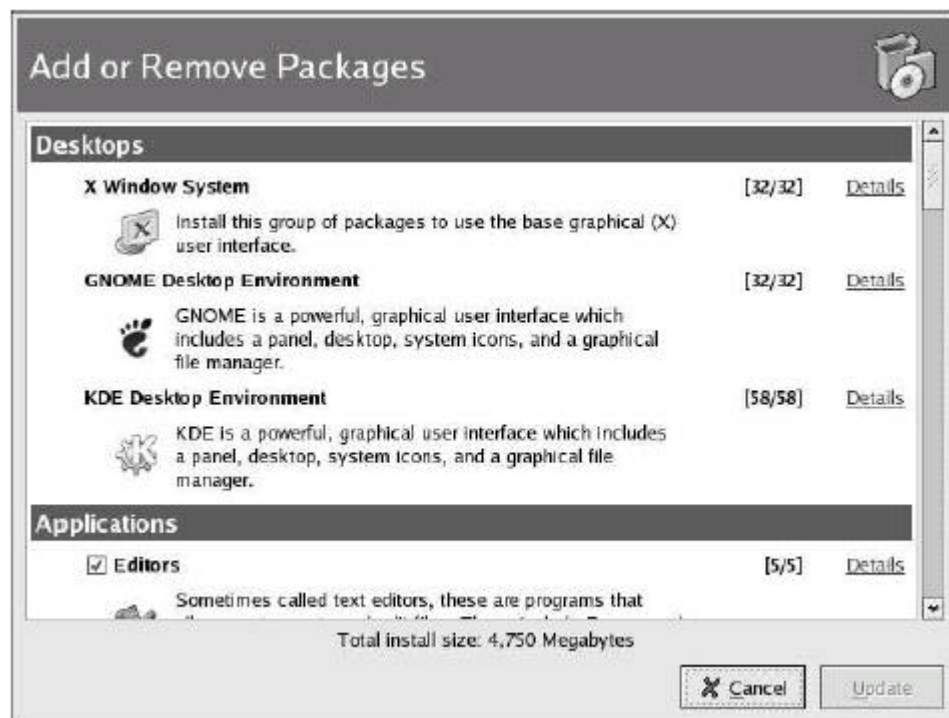
Aunque Linux es un sistema operativo muy completo, con gran cantidad de utilidades, los usuarios de la máquina suelen tener necesidad de completarlo con otras aplicaciones, bien creados por los programadores de la empresa, bien adquiridos o compilados.

Puede encontrarse una gran variedad de utilidades gratuitas y de libre distribución preparadas para ser instaladas o compiladas en Linux. El analista debe sopesar los pros y los contras de cada una de ellas y –si lo cree indispensable– realizar su instalación.

### 6.1. RPM.

Los sistemas Linux modernos se instalan fácilmente mediante paquetes de programas. Hay 3 tipos básicos de paquetes Linux: RPM (usado por las distribuciones Red Hat o Mandrake), DEB (Debian) y TGZ (Slackware).

El **Gestor de Paquetes de Red Hat (RPM)** es la herramienta usada para instalar, actualizar o eliminar los paquetes de programas en el Linux de Red Hat. La mayoría de los paquetes de programas se instalan junto con el sistema operativo, aunque la herramienta para gestión de paquetes `redhat-config-packages` –presentada en el siguiente gráfico <sup>[3]</sup>– y el mandato `rpm` permiten al administrador una gestión posterior.



Cada paquete RPM contiene la descripción y los ficheros de instalación de una herramienta o servicio. El usuario `root` debe instalar y activar el menor número de paquetes necesario –por motivos de espacio y de seguridad–, pero teniendo en cuenta siempre las necesidades de sus usuarios.

El siguiente ejemplo muestra la ejecución del mandato `rpm` para comprobar los ficheros que forman parte del paquete de gestión de claves.

```
> rpm -ql passwd
/etc/pam.d/passwd
/usr/bin/passwd
/usr/share/man/man1/passwd.1.gz
```

## 6.2. Archivadores y compresores.

Un programa **archivador** es el encargado de empaquetar ficheros y directorios en un único fichero conocido también como archivador. Los programas de archivado más usados en Red Hat son RPM (`rpm`) y Tar de GNU (`tar` o `gtar`).

El programa compresor utiliza un algoritmo especial para compactar los ficheros, almacenando la misma información en menor espacio. Para recuperar los datos debe usarse un algoritmo de descompresión. Los compresores típicos de Red Hat son Zip de GNU (`gzip`) y BZip2 (`bzip2`).

Los programas gratuitos suelen distribuirse en formato empaquetado y comprimido. La siguiente tabla describe los formatos más usados y las extensiones de estos ficheros.

Extensión	Descripción del formato
.rpm	Paquete de Red Hat, que contiene la descripción y los datos comprimidos de los ficheros del paquete.
.tar	Archivador no comprimido, almacena el nombre y el contenido de cada fichero o directorio. Esta orden también se usa para hacer copias de seguridad.
.gz	Fichero comprimido; suele combinarse con <code>tar</code> para comprimir un archivador (extensión <code>.tar.gz</code> ).
.bz2	Compresión por algoritmo de ordenación de bloques, más eficiente y más lento que GZip; suele combinarse con <code>tar</code> para comprimir un archivador (extensión <code>.tar.bz2</code> ).

## 6.3. Programas GNU.

Para finalizar el capítulo se va a describir a grandes rasgos la forma de compilar una aplicación a partir de su código fuente.

Una vez descargado el fichero con el código fuente de la aplicación, el administrador de la máquina debe leer detenidamente las instrucciones de configuración e instalación. En algunos casos será obligatorio realizar la instalación previa de otros programas o de bibliotecas de código. También puede ser necesario actualizar o crear algunas variables de entorno antes de comenzar con la instalación.

### 6.3.1. Compilar un paquete RPM.

A continuación se describen los pasos que deben seguirse para compilar un programa con código fuente almacenado en un paquete RPM (extensión `.src.rpm`) e instalar la aplicación en el sistema.

<code>rpm -ihv Prog-Vers.src.rpm</code>	Instala el código fuente de la aplicación en un subdirectorio de <code>/usr/src/redhat</code> ( <code>/usr/src/VersiónNúcleo</code> en el caso de los paquetes del núcleo).
<code>cd /usr/src/redhat/SPECS</code> <code>vi Programa.spec</code>	Editar el fichero de especificaciones del paquete para realizar las modificaciones necesarias. Este fichero contiene los parámetros de configuración, el entorno de compilación y los directorios de instalación.
<code>rpm -ba Programa.spec</code>	Configura, compila y genera el paquete RPM de la aplicación para la arquitectura de procesadores del sistema, siguiendo las instrucciones indicadas en el fichero de especificaciones.
<code>cd /usr/src/redhat/RPMS/Arquit</code> <code>rpm -Uhv Prog-Vers.Arquit.rpm</code>	Instala el paquete de la aplicación.

### 6.3.2. Compilar un paquete comprimido.

El código fuente de los programas GNU <sup>[viii]</sup> suele encontrarse comprimido en formato `.tar.gz` (aunque cada vez abunda más el formato `.tar.bz2`) y normalmente está preparado para instalarse en la estructura de directorios `/usr/local`.

Siguiendo esta norma, a continuación se describen los pasos para instalar una aplicación de libre distribución, que utilice el método de configuración de GNU.

<code>cd /usr/local/src</code>	Extrae el contenido del código fuente en un subdirectorio.
<code>tar xvzf Prog-Vers.tar.gz</code>	Conviene revisar previamente el paquete de la aplicación porque en algunos casos es necesario crear previamente el subdirectorio antes de extraer los datos.
<code>configure [Opciones ...]</code>	Indicar las opciones de configuración del programa, tales como: parámetros, bibliotecas, directorios, programas auxiliares, etc.  Cada aplicación soporta opciones de configuración diferentes, por lo que se recomienda leer la documentación y ejecutar antes de este paso <code>configure --help</code> .
<code>make</code>	Proceso de compilación del código fuente.
<code>make install</code>	Instalación de la aplicación.
<code>make clean</code>	Opcionalmente, puede eliminarse los ficheros intermedios que se han generado durante la compilación (código objeto, ficheros temporales, etc.).

Resulta de especial interés conservar un registro completo de las operaciones ejecutadas. Para ello puede utilizarse un fichero de texto que contenga la siguiente información para cada mandato ejecutado.

```
Mandato [Opciones]
-----
salida completa de la ejecución del mandato
...
```

Para obtener la salida de un mandato, ejecutar:

```
Mandato [Opciones] 2>&1 | tee -a FicheroRegistro
```

## 7. Configuración de la red.

La red informática es el medio por el cual el servidor puede comunicarse con los usuarios y con otras máquinas, tanto servidores como clientes, permitiendo el intercambio masivo de información entre ordenadores.

De acuerdo con la planificación efectuada, la empresa debe contar con una infraestructura adecuada para el intercambio de datos. Asimismo, los dispositivos de los servidores deben cumplir las necesidades previstas, ofreciendo un ancho de banda y una capacidad de procesamiento adecuados.

Existe una gran variedad de tipos de redes y protocolos de comunicaciones, sin embargo, este capítulo se centra en redes Ethernet con protocolos TCP/IP, los más usados en la conexión a Internet y en redes privadas.

### 7.1. Interfaces de red.

El ordenador necesita un dispositivo –conocido como **tarjeta de red**– que le permita conectarse a cada una de las subredes que tenga directamente a su disposición.

El sistema operativo Linux puede trabajar con una gran variedad de tipos de máquinas y periféricos. Para normalizar el acceso a la red, el sistema dispone de una serie de funciones básicas. El conjunto de estas funciones usadas en una arquitectura de comunicaciones determinada, se conoce como **interfaz de red**.

Por último, la interfaz de red dialoga con el dispositivo físico mediante un módulo específico del núcleo denominado **controlador de red**.

Las modernas versiones de Linux detectan automáticamente las tarjetas de red, cargan los módulos adecuados del núcleo y asignan los interfaces de red por defecto. El administrador puede establecer los parámetros de conexión durante el proceso de instalación del sistema.

Red Hat establece una nomenclatura para cada tipo de interfaz de red, añadiendo un número de orden para cada conector del mismo tipo (empezando por el número 0). La siguiente tabla describe la nomenclatura usada por Red Hat para los principales interfaces de red <sup>[3]</sup>.

Interfaz	Descripción
lo	Interfaz virtual para pruebas (tiene asignada la dirección IP 127.0.0.1).
eth	Dispositivos Ethernet (también se usa en dispositivos ADSL y Ethernet inalámbrica).
tr	Redes en anillo de tipo Token Ring.
ppp	Conexión mediante módem o RDSI.

Cada dispositivo de red cuenta con una dirección física de acceso al medio (**dirección MAC**) única y diferente, asignada por el fabricante. Sin embargo, durante el proceso de activado del interfaz de red deben asignarse sus parámetros de conexión.

La dirección MAC de una tarjeta Ethernet está formada por 48 bits representados en 6 campos con 2 dígitos exadecimales cada uno.

## 7.2. TCP/IP.

El protocolo de comunicaciones **TCP/IP** (*Transmission Control Protocol/Internet Protocol*) permite la localización y comunicación de todo tipo de máquinas conectadas a Internet. TCP/IP está constituido por un conjunto de protocolos basado en capas <sup>[12]</sup>:

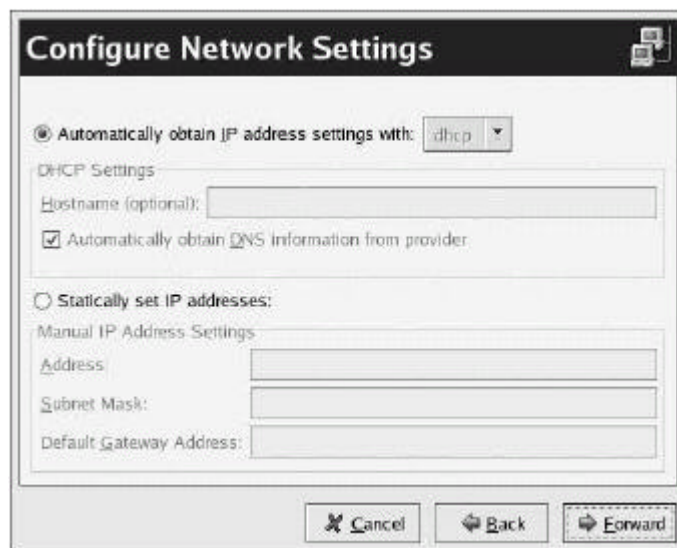
- La capa de red –equivalente al nivel 3 de la norma OSI–, que establece el camino óptimo que deben seguir los paquetes de información que comunican varias máquinas. Utiliza el **Protocolo de Internet (IP)**.
- La capa de transporte –equivalente al nivel 4 de la pila de protocolos OSI–, que permite establecer una conexión entre nodos de la red. Existen 2 protocolos de transporte: el **Protocolo para el Control de la Transmisión (TCP)** –que realiza una comunicación síncrona y segura con recuperación de datos en caso de error– y el **Protocolo de Datagramas del Usuario (UDP)** –que permite una comunicación asíncrona basada en paquetes denominados **datagramas**.

El conjunto de protocolos TCP/IP establece un mecanismo basado en direcciones y nombres que permite localizar inequívocamente cada máquina conectada a Internet. Las equivalencias entre direcciones IP y nombres de máquinas son realizadas por ordenadores especiales que atienden las consultas mediante el protocolo conocido como **Servicio de Nombres de Dominios (DNS)**.

El administrador del sistema tiene que establecer los parámetros para cada interfaz de red del sistema, bien mediante ficheros de configuración locales, bien generados por un servidor DHCP remoto, que puede asignar los valores estática o dinámicamente. En ambos casos, deben especificarse los aspectos descritos en la siguiente tabla.

<b>Dirección IP del interfaz:</b>	Dirección única y diferenciada en toda Internet o en la red privada, formada por 32 bits en IPv4 o por 128 bits en IPv6.
<b>Máscara de red:</b>	Especifica mediante una operación lógica Y la porción de bits de la dirección IP común a todas las máquinas de la misma subred.
<b>Dirección de difusión de la red:</b>	Usada para enviar paquetes de información a todos los dispositivos de la misma subred.
<b>Nombre del nodo y nombre del dominio de red:</b>	Ambos valores en conjunto describen fácil y unívocamente una determinada máquina en toda Internet o en la red privada.
<b>Direcciones de los servidores de nombres:</b>	Servidores encargados de la resolución de nombres en Internet mediante el protocolo DNS. No suele usarse en redes privadas.

La siguiente figura muestra la pantalla principal de la herramienta **redhat-config-network**, suministrada por Red Hat para la configuración básica de las interfaces de red <sup>[3]</sup>.



### 7.3. Configuración de la red.

Para terminar la instalación básica de la red, el responsable del sistema debe revisar –y en algunos casos modificar– los ficheros de configuración de los servicios esenciales del sistema. La siguiente tabla describe los formatos de estos ficheros <sup>[12]</sup>.

<b>/etc/sysconfig/network</b>	
<b>Descripción:</b>	Establece los valores de las variables básicas para el servicio de red (nombre, dominio, dirección del <i>encaminador</i> , etc.
<b>Formato:</b>	<i>Variable=Valor</i> ...
<b>/etc/sysconfig/network-scripts/ifup-Interfaz</b>	
<b>Descripción:</b>	Establece los valores de las variables de red específicas para cada interfaz de red (recogida de valores de red mediante DHCP, BOOTP o local), dirección IP, máscara de red, dirección de difusión, etc.
<b>Formato:</b>	<i>Variable=Valor</i> ...
<b>/etc/hosts</b>	
<b>Descripción:</b>	Almacena la asociación entre dirección IP, nombre y alias de ordenadores conocidos. Siempre debe estar presente la dirección 127.0.0.1.
<b>Formato:</b>	<i>DirecciónIP Nombre [Alias ...]</i> ...



<i>/etc/resolv.conf</i>	
<b>Descripción:</b>	Establece las bases para la resolución de nombres, indicando dominio del ordenador, dirección de los servidores de nombres y otros dominios de interés.
<b>Formato:</b>	<pre>domain Dominio nameserver IPServidorDNS ... [search DominioBúsqueda ...]</pre>
<i>/etc/nsswitch.conf</i>	
<b>Descripción:</b>	Indica el orden de búsqueda para ficheros de red.
<b>Formato:</b>	<pre>TipoFichero TipoBúsqueda ... ...</pre>
<b>Tipos de búsqueda:</b>	<pre>files: archivos locales. nis: NIS. nisplus: NIS+. ldap: servicio de directorios. dns: servicio de nombres.</pre>
<i>/etc/services</i>	
<b>Descripción:</b>	Indica el protocolo y el puerto utilizado por cada servicio de comunicaciones (este fichero no debe modificarse, ya que suele estar bien configurado).
<b>Formato:</b>	<pre>Servicio Puerto/Protocolo [ Alias ... ] ...</pre>

## 7.4. Servicios de red.

Los protocolos definidos para controlar cada servicio de comunicaciones utilizan una especie de punto de anclaje a los protocolos TCP o UDP. Este mecanismo se conocido como **puerto**.

Si una aplicación quiere ofrecer un cierto servicio, se engancha ella misma a un puerto y espera las peticiones de los clientes (escuchar en el puerto). Un cliente que quiera usar este servicio se asigna un puerto libre en su nodo local y se conecta al puerto del servidor en el nodo remoto. El puerto del servidor podrá ser abierto por diferentes máquinas, pero nunca podrán usarlo por más de una al mismo tiempo <sup>[6]</sup>.

### 7.4.1. Breve descripción de los principales servicios de red.

Para finalizar el capítulo, la siguiente tabla presenta una sencilla descripción de los servicios de red más utilizados en Linux.

<b>dhcp</b>	
<b>Descripción:</b>	Servicio de asignación remota de parámetros de la red; utiliza el protocolo DHCP, aunque también puede usar BOOTP.
<b>Paquete RPM:</b>	dhcp (servidor), dhcpd (cliente)
<b>Fichero de configuración:</b>	/etc/dhcpd.conf
<b>ldap</b>	
<b>Descripción:</b>	Servicio de acceso a directorios mediante protocolo LDAP. Un directorio es un árbol donde se incluye todo tipo de recursos agrupados lógicamente.
<b>Paquete RPM:</b>	openldap, openldap-servers, openldap-clients
<b>Fichero de configuración:</b>	/etc/openldap/slapd.conf
<b>Directorio de esquemas LDAP:</b>	/etc/openldap/schemas
<b>httpd</b>	
<b>Descripción:</b>	Servicio de acceso a la información mediante hipertexto, utilizando el protocolo HTTP.
<b>Paquete RPM:</b>	apache, apacheconf (herramienta de configuración)
<b>Fichero de configuración:</b>	/etc/httpd/conf/httpd.conf
<b>squid</b>	
<b>Descripción:</b>	Servicio de acceso a la información mediante hipertexto, utilizando el protocolo HTTP.
<b>Paquete RPM:</b>	squid
<b>Fichero de configuración:</b>	/etc/squid/squid.conf
<b>samba</b>	
<b>Descripción:</b>	Servicio que permite compartir recursos (ficheros e impresoras) mediante los protocolos CISS o SMB.
<b>Paquete RPM:</b>	samba-common, samba-servers, samba-clients
<b>Fichero de configuración:</b>	/etc/samba/smb.conf

## 8. Utilidades de administración.

El administrador de un sistema operativo debe contar con una serie de utilidades de gestión que le ayuden a realizar su trabajo de un modo más eficiente y que complementen las herramientas básicas del sistema operativo.

Este capítulo describe algunas de las utilidades más comunes en la gestión del Linux de Red Hat y que deben ser instaladas en la máquina.

Las herramientas administrativas deben ser ejecutadas únicamente por usuarios privilegiados, salvo en aquellas excepciones previstas por el gestor del sistema.

### 8.1. Ejecución cronológica.

Las herramientas cronológicas sirven para ejecutar programas en un momento determinado. Aunque Red Hat cuenta con algunas aplicaciones propias, los siguientes apartados describen las herramientas cronológicas más usadas en Unix.

#### 8.1.1. cron.

El servicio cronológico (**cron**) utiliza un fichero de configuración para la ejecución periódica de tareas administrativas. La tabla siguiente describe las características básicas de este servicio.

Paquete RPM:	vixie-cron
Servicio:	<b>crond</b>
Fichero de configuración:	/etc/crontab
Ficheros de control de acceso:	/etc/cron.allow, /etc/cron.deny

El mandato **crontab** es el utilizado para gestionar la ejecución cronológica. El sistema realiza las siguientes comprobaciones cada minuto:

- la tabla cronológica principal /etc/crontab,
- las tablas cronológicas de los usuarios con permiso de acceso, almacenadas en /var/spool/cron/Usuario.

La siguiente tabla presenta el formato de los ficheros de configuración y de control de acceso a este servicio.

Formato	Descripción
<b>/etc/crontab, /var/spool/cron/Usuario</b>	
<code>Variable=Valor</code> ...	Asignación de las variables de entorno <code>SHELL</code> , <code>PATH</code> , <code>MALITO</code> y <code>HOME</code> .
<code>Min Hora Día Mes DíaSem Mandato</code> ...	Lista de configuración de la ejecución cronológica. Sus campos son: <ol style="list-style-type: none"> <li>1. Minuto.</li> <li>2. Hora.</li> <li>3. Días del mes.</li> <li>4. Mes</li> <li>5. Día de la semana (0=domingo, 1=lunes, etc.).</li> <li>6. Mandato a ejecutar; conviene redirigir las salidas a ficheros de control.</li> </ol> <p>Nota: Los valores numéricos pueden ser una cifra, un rango de valores separados por guión o un conjunto de valores y rangos separados por comas (el comodín <code>*</code> indica cualquier valor posible).</p>
<b>/etc/cron.allow</b>	
<code>Usuario</code> ...	Lista de usuarios que pueden ejecutar la orden <b><code>crontab</code></b> para realizar ejecuciones cronológicas. Si no existe este fichero, todos los usuario tienen permiso, excepto los listados en <code>cron.deny</code> .
<b>/etc/cron.deny</b>	
<code>Usuario</code> ...	Lista de usuarios que no pueden ejecutar la orden <b><code>crontab</code></b> . Si no existe este fichero, sólo tienen permiso los usuarios listados en <code>cron.allow</code> .

### 8.1.2. at, batch.

Los mandatos **`at`** y **`batch`** se usan para ejecutar un progrma de forma puntual. El primero de ellos planifica el lanzamiento de la aplicación a una hora determinada; mientras que el segundo lo realiza cuando el sistema está poco cargado. Las características del servicio son:

Paquete RPM:	<code>at</code>
Servicio:	<b><code>atd</code></b>
Ficheros de control de acceso:	<code>/etc/at.allow</code> , <code>/etc/at.deny</code> (formato como los de cron).

Los mandatos usados en este servicio son:

Mandato	Descripción
<b>at</b>	Solicita por teclado el mandato a ejecutar a una hora determinada..
<b>batch</b>	Solicita por teclado el mandato que se ejecutará cuando el sistema no esté sobrecargado (ocupación menor a 0'8).
<b>atq</b>	Presenta la cola de trabajos pendientes.

## 8.2. Control de procesos.

El *superusuario* debe mantener el control del sistema en todo momento, realizando revisiones periódicas de los procesos que se están ejecutando en el servidor, lo que puede evitar problemas y abusos que afecten el funcionamiento normal de la máquina. La siguiente tabla describe los mandatos más usados para el control de procesos.

Mandato	Descripción
<b>ps</b>	Presenta una lista con los procesos activos en la máquina, indicando propietario, identificador del proceso (PID), identificador del proceso padre (PPID), mandato ejecutado, etc.
<b>kill</b>	Manda una señal de interrupción a uno o a varios procesos. Suele usarse para finalizar su ejecución.
<b>nice</b>	Cambia la prioridad de los procesos. Se usa normalmente para bajar el tiempo de ejecución de procesos que saturan al sistema.
<b>top</b>	Presenta una lista actualizada de los procesos que consumen más recursos. También permite mandar señales y modificar la prioridad de ejecución.
<b>gnome-system-monitor</b>	Monitor gráfico del sistema similar a <b>top</b> .
<b>lsof</b>	Lista los ficheros y las conexiones de red abiertos por cada proceso, indicando además el propietario, PID, prioridad, mandato, etc.

El mandato **lsof** –que debe instalarse específicamente en el sistema– es una potente herramienta administrativa, ya que muestra todos los ficheros, tuberías con nombre, dispositivos y conexiones de red abiertos por cada proceso. Suele usarse para comprobar aquellos procesos sospechosos de crear problemas y para revisar las conexiones de red de cada servicio.

## 8.3. Medidas de rendimiento.

### 8.3.1. Mandatos `systat`.

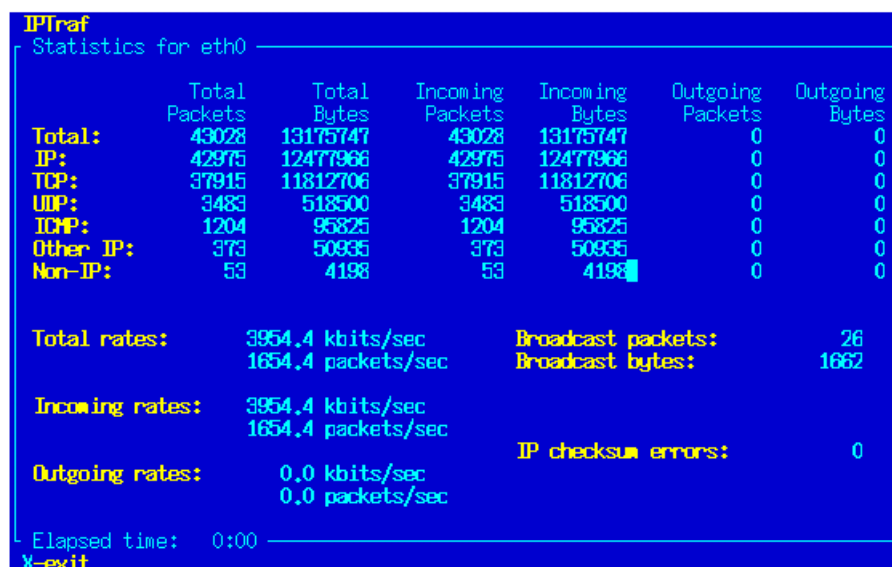
Los mandatos de tipo `systat` muestran información sobre la utilización de los recursos del sistema. Estos órdenes usan como parámetros un intervalo de ejecución y el número máximo de veces que debe listar sus datos. La siguiente tabla describe los mandatos `systat` más comunes.

Mandato	Descripción
<code>iostat</code>	Presenta estadísticas de utilización de la E/S de cada disco.
<code>mpstat</code>	Muestra estadísticas de los procesadores.
<code>vmstat</code>	Indica el estado de ocupación de la memoria virtual.

### 8.3.2. IPTraf.

Aunque IPTraf es un pequeño paquete que puede usarse para monitorizar y controlar la red, el uso fundamental de esta herramienta es el de recopilar estadísticas de uso de los interfaces de red.

El programa `iptraf` es una herramienta de consola a base de menús, que recoge estadísticas de los protocolos más usados en Internet (IP, TCP, UDP, ICMP, otros paquetes IP y tráfico no IP), como se muestra en el siguiente gráfico.



The screenshot shows the IPTraf terminal window with a blue background and yellow text. It displays statistics for the eth0 interface. The data is organized into several sections: a main table of statistics, broadcast statistics, rates, and errors. The main table has columns for Total Packets, Total Bytes, Incoming Packets, Incoming Bytes, Outgoing Packets, and Outgoing Bytes. The statistics are listed for Total, IP, TCP, UDP, ICMP, Other IP, and Non-IP. Below the main table, there are sections for Total rates, Incoming rates, Outgoing rates, Broadcast packets and bytes, and IP checksum errors. The elapsed time is shown as 0:00, and the window ends with a prompt to press X to exit.

IPTraf						
Statistics for eth0						
	Total Packets	Total Bytes	Incoming Packets	Incoming Bytes	Outgoing Packets	Outgoing Bytes
Total:	43028	13175747	43028	13175747	0	0
IP:	42975	12477966	42975	12477966	0	0
TCP:	37915	11812706	37915	11812706	0	0
UDP:	3483	518500	3483	518500	0	0
ICMP:	1204	95825	1204	95825	0	0
Other IP:	373	50935	373	50935	0	0
Non-IP:	53	4198	53	4198	0	0
Total rates: 3954.4 kbits/sec Broadcast packets: 26						
1654.4 packets/sec Broadcast bytes: 1662						
Incoming rates: 3954.4 kbits/sec						
1654.4 packets/sec						
Outgoing rates: 0.0 kbits/sec IP checksum errors: 0						
0.0 packets/sec						
Elapsed time: 0:00						
X-exit						

## 8.4. Revisión de ficheros históricos.

Los ficheros históricos del sistema mantienen un registro con información sobre incidencias de interés, generadas por el núcleo del sistema operativo, por los servicios y aplicaciones instaladas y por las alertas de seguridad.

Estos ficheros se almacenan en el directorio `/var/log` –o en alguno de sus subdirectorios– y deben ser revisados periódicamente por el administrador del sistema para comprobar el estado del servidor, ya que son una guía fundamental en la resolución de problemas de funcionamiento de los dispositivos y en el control de la seguridad de los datos <sup>[3]</sup>.

La mayoría de los ficheros históricos son generados por el servicio **syslogd**, que debe estar iniciado en todo momento en el sistema para recoger las incidencias de control. El fichero de configuración del servicio es `/etc/syslog.conf` y el formato más común de las líneas de este fichero se describe en la siguiente tabla.

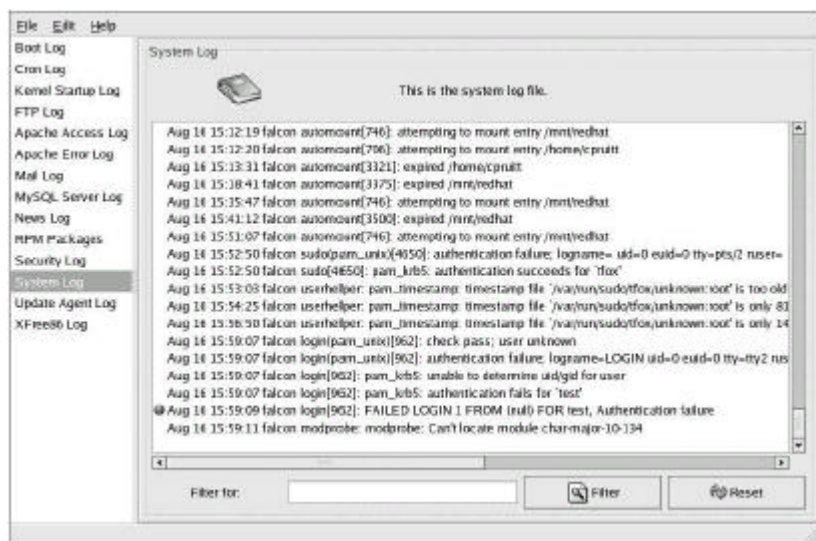
Formato	Descripción
<b>/etc/syslog.conf</b>	
<i>Servicio[,...].Nivel;... Fichero</i> ...	Configuración del servicio de recogida de incidencias de <b>syslogd</b> . Sus campos son: <ol style="list-style-type: none"><li>1. Servicios (separados por comas) generadores de la incidencia.</li><li>2. Nivel de registro (depuración información, error, crítico, emergencia, etc.). Un nivel con mayor descriptivo implica almacenar también la información de los niveles más restrictivos.</li><li>3. Fichero de registro.</li></ol> Nota: puede usarse el comodín <b>*</b> para indicar cualquier servicio o cualquier nivel de registro; es posible indicar varias combinaciones de <i>Servicio.Nivel</i> en la misma línea.

Para evitar que los ficheros históricos sean excesivamente grandes o muy antiguos y, por lo tanto, difíciles de tratar, debe incluirse la herramienta **logrotate** en el **cron** del sistema. Esta utilidad renombra los ficheros históricos añadiéndoles un número de orden, de tal forma que el número mayor es el fichero más antiguo y el más nuevo es el que no se le añade dicha extensión.

El programa **logrotate** se configura editando el fichero `/etc/logrotate.conf` y los ficheros del directorio `/etc/logrotate.d`.

Los archivos históricos son normalmente ficheros de texto –pueden ser revisado con cualquier mandato Unix, como **cat** o **more**–, sin embargo las últimas versiones del Linux de Red Hat ofrecen una herramienta gráfica que ayuda a una presentación más cómoda de estos ficheros.

La siguiente figura presenta un ejemplo de ejecución del programa `redhat-logviewer`, incluido con el entorno gráfico GNOME de Red Hat 8.0 <sup>[3]</sup>.





## 9. Impresoras.

Las impresoras son unos de los recursos esenciales de cualquier servidor, ya que permiten crear copiar en papel de documentos. Se utilizan tanto en oficinas como en ámbitos académicos y domésticos <sup>[1]</sup>.

Debe realizarse una previsión exhaustiva del tipo de documentación que los usuarios van a necesitar imprimir, ya que de ello depende –y de las posibilidades económicas de la empresa– la elección del tipo de dispositivo impresor que deba adquirirse, de cómo va a accederse a dicho periférico y del tipo de servicio que debe ser instalado y activado en el sistema.

La siguiente tabla presenta las características generales para los tipos de impresoras más utilizados <sup>[1]</sup>.

<b>Impresora matricial:</b>	bajo coste, baja calidad, lenta, puede usar folios y papel continuo.
<b>Impresora de líneas:</b>	coste alto, muy baja calidad, rápida, sólo utiliza papel continuo.
<b>Impresora de inyección:</b>	coste medio, puede alcanzar una buena calidad de impresión, la velocidad depende de la calidad, puede imprimir gráficos en color, sólo puede usar folios.
<b>Impresora LASER:</b>	coste medio/alto, alta calidad, gran velocidad, mayor coste en consumibles, usa folios.
<b>Impresora LASER color:</b>	alto coste, alta calidad, gran velocidad, alto coste en consumibles, puede imprimir en color sobre folios.
<b>Trazador:</b>	usado para impresión de trabajos de diseño gráfico.

Por último debe evaluarse si la impresora va a estar enchufada localmente al servidor, si va a usarse como un recurso compartido con otros clientes o si se conectará directamente a la red informática, en el caso de que ésta cuente con tarjeta de red.

### 9.1. Servicios de impresión.

El **servicio de impresión** de Unix recoge las solicitudes –locales o remotas– para obtener copias sobre papel y las dirige hacia una cola de trabajos. Como la impresora es un dispositivo más lento que los procesadores, el servicio selecciona el trabajo de impresión correspondiente según sus reglas de control y lo dirige hacia la impresora correspondiente cuando ésta queda libre para realojar la tarea.

Un servicio de impresión consta de los siguientes componentes:

- El proceso servidor que recoge los trabajos de los clientes.
- Una serie de utilidades de gestión, que permitan controlar los trabajos pendientes.

- Instalar un controlador de dispositivo para cada impresora definida, ya sea local o remota, donde se indiquen las características básicas del aparato (tipo de papel, calidad de impresión, etc.).
- Una cola de impresión para cada dispositivo definido, donde se dispongan los trabajos que están pendientes de impresión. Cada cola debe tener un nombre único en el sistema y ser designada mediante alias.

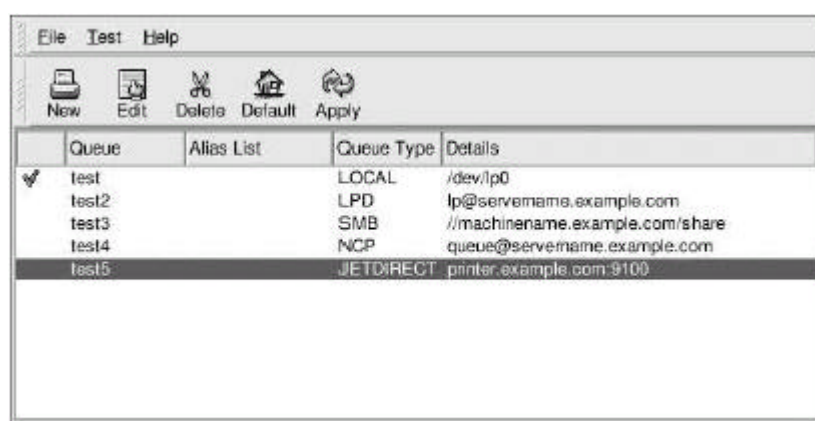
En el resto del capítulo se describen las características principales de los 2 tipos de servicios de impresión más utilizados en los sistemas Red Hat: LPRng y CUPS. Conviene destacar que, mediante un servidor Samba, pueden compartirse las impresoras definidas, independientemente del servicio que haya sido activado.

### 9.1.1. LPRng.

El servicio de impresión **LPR de siguiente generación (LPRng)** es una evolución de las clásicas utilidades **lpr** del Unix de Berkeley (BSD). A continuación se muestra la tabla de características del servicio.

Paquete RPM:	LPRng
Servicio:	lpd
Fichero de configuración:	/etc/printcap
Fichero de control de acceso:	/etc/hosts.lpd

El siguiente gráfico muestra la utilidad **redhat-config-printer**, que permite definir las colas de impresión y sus correspondientes controladores de dispositivos <sup>[3]</sup>.



Este programa permite definir 5 tipos de impresoras:

- Impresora local, donde puede definirse todas las características de la cola de impresión y las restricciones de seguridad.
- Impresora remota conectada a un servidor Unix con servidor **lpd**.

- Conexión remota a un recurso de impresión compartido por un servidor Samba o Windows NT/2000/XP.
- Conexión remota a un servidor Novell.
- Conexión remota a una impresora de red mediante el protocolo JetDirect de HP-Compaq, especificando máquina y puerto TCP.

Posteriormente debe seleccionarse el controlador correspondiente al modelo de la impresora y el tipo de filtro necesario para los datos que se desea enviar al dispositivo.

Por último, la siguiente tabla revela los mandatos que brinda el servicio LPRng.

Mandato	Descripción
<b>lpd</b>	Servidor de impresión..
<b>lpc</b>	Controla las funciones básicas del servicio de impresión.
<b>lpr</b>	Manda un trabajo de impresión a una cola.
<b>lpq</b>	Lista el contenido de los trabajos pendiente de una cola.
<b>lprm</b>	Elimina un trabajo de impresión.

### 9.1.2. CUPS.

El **Servicio de Impresión Común para Unix (CUPS)** brinda el soporte de una capa de impresión portátil para sistemas Unix, ofreciendo interfaces de comandos compatibles con los servicios de Unix System V y los de Unix BSD. Las características principales de CUPS son [vii].

- Utiliza el **Protocolo para Impresión en Internet (IPP)** como base para la gestión de los trabajos de impresión.
- Garantiza la compatibilidad con Unix System V (**lp**) y con Unix BSD (**lpr**).
- Soporta interfaz vía *web*.
- Ofrece contabilidad y cuotas de trabajos y de páginas impresas.
- Permite usar servicio de directorios para impresoras conectadas en red.
- Soporta clientes IPP y LPD.
- Brinda servicios de autenticación de usuarios y comunicación encriptada.
- Utiliza **Controladores de Impresión Portátiles (PPD)** para definir las características de las impresoras.
- Permite imprimir trabajos PostScript, PDF, HP/GL, texto y gráficos (BMP, GIF, JPEG, PNG, TIFF, etc.).
- Localización automática de impresoras conectadas y de los servidores CUPS de la misma subred.

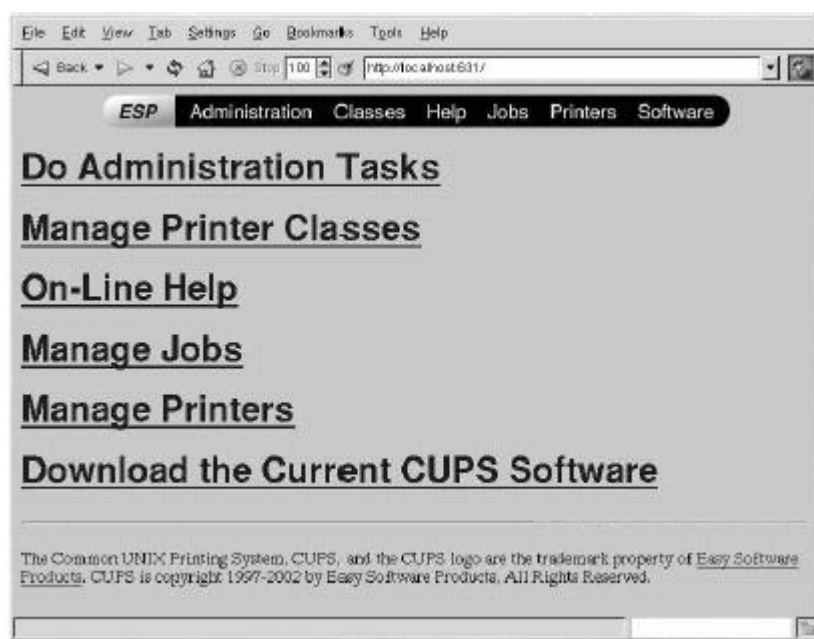
El protocolo IPP suministra métodos para la autenticación segura de los usuarios y la posibilidad de codificar la información mediante capas TLS o SSL. Estas características pueden ser configuradas tanto en las impresoras, como en el servidor CUPS.

La próxima tabla enseña las características básicas de configuración del servicio CUPS instalado en la versión 8.0 de Red Hat:

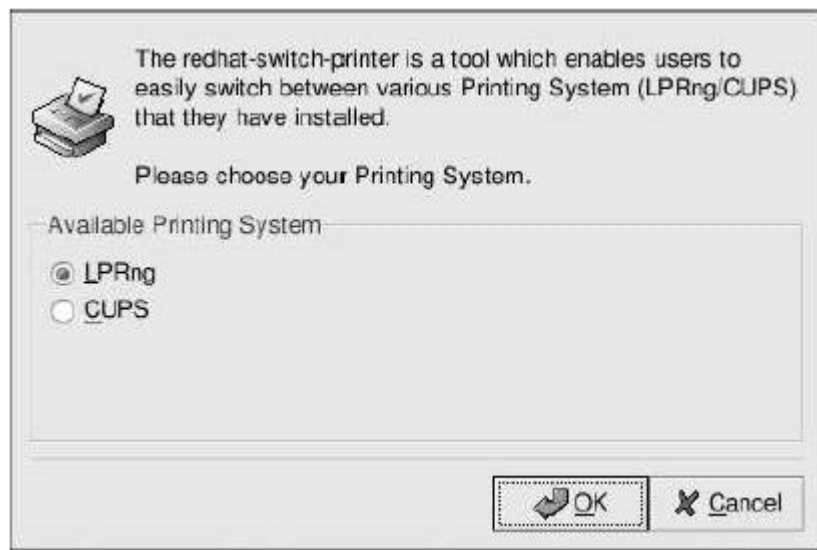
<b>Paquete RPM:</b>	<code>cups, cups-libs, cups-drivers</code>
<b>Servicio:</b>	<b>cups</b>
<b>Directorio de configuración:</b>	<code>/etc/cups</code>
<b>URL de gestión:</b>	<code>http://Servidor:631/</code>

La sintaxis del fichero principal de configuración del servicio CUPS es similar a la utilizada en los archivos correspondientes al servicio de hipertexto Apache, incluso se definen algunas órdenes comunes.

El siguiente gráfico describe el menú principal utilizado para la configuración del servicio CUPS mediante un cliente *web* <sup>[3]</sup>.



La herramienta gráfica **redhat-switch-printer** permite elegir cómodamente el tipo de servidor de impresión (LPRng o CUPS). El gráfico posterior presenta el programa distribuido por el sistema operativo Red Hat Linux <sup>[3]</sup>.



## 10. Copias de seguridad.

Todos los sistemas informáticos pueden tener problemas de funcionamiento que comprometan la integridad de sus datos, causados –como se ha descrito en el primer capítulo de este curso– por ataques, por fallos de funcionamiento la máquina, por errores humanos o por problemas ambientales.

Las copias de seguridad son el método más seguro para guardar los datos vitales del sistema en un medio de almacenamiento externo y así poder recuperarlos fácilmente ante cualquier imprevisto <sup>[7]</sup>.

Las copias deben hacerse sobre un medio independiente del sistema, fiable, relativamente rápido y barato. La siguiente tabla muestra las características principales de los medios de almacenamiento masivo más utilizados.

<b>Cintas:</b>	coste medio, sistema lento pero fiable, gran capacidad, soportes de coste medio que pueden reutilizarse un número limitado de veces. Es el medio más comúnmente utilizado.
<b>CD-R y CD-RW:</b>	bajo coste, buena velocidad de lectura, rápido acceso selectivo, poca capacidad de almacenamiento, los CD-R sólo pueden ser escritos una única vez.
<b>DVD:</b>	coste medio, rápido, con acceso selectivo, capacidad intermedia, existen varios formatos que pueden ser incompatibles entre sí.
<b>Almacenamiento en red (SAN):</b>	muy caro, rápido, fiable y con gran capacidad; se utiliza una red de banda ancha para crear un sistema de almacenamiento masivo.

### 10.1. Planificación.

El proceso de creación de copias de seguridad debe estar automatizado y planificado de antemano. Existen distintas categorías de datos –sistema operativo, aplicaciones, datos de programas y datos de usuarios– y cada una de ellas requiere un plan de actuación independiente.

El administrador del sistema debe crear una plantilla semanal o mensual para la realización de las copias de seguridad para cada categoría de datos de los sistemas a su cargo, indicando al menos: tipo de copia, técnica de grabación y soporte.

El operador tiene que etiquetar los medios fungibles utilizados en cada copia, indicando como mínimo: orden de restauración, máquina, tipo de datos y fecha.

La siguiente tabla detalla los 3 tipos de copias de seguridad.

<b>Completa:</b>	se copian todos los datos (del servidor, de un sistema de archivos o de una categoría de datos determinada), por lo tanto el proceso es más lento y puede necesitar más medios de almacenamiento.
<b>Incremental:</b>	sólo se copian los archivos que se hayan modificado desde la última copia, con lo que se ahorra en tiempo de ejecución del proceso, pero se necesita efectuar una restauración principal y todas las incrementales realizadas hasta la fecha.
<b>Diferencial:</b>	sólo se copian los archivos que se hayan modificado desde la última copia completa; el ahorro en tiempo de creación y en medios es intermedio, pero sólo requiere restaurar la última copia completa y la última copia diferencial.

Un plan de copias típico puede constar de hacer una copia completa semanalmente y realizar copias diferenciales o incrementales cada día.

En sistemas Linux no deben hacerse copias del sistema de archivos virtual `/proc`, de los medios extraíbles montados en `/mnt`, de los sistemas de archivos importados por la red, ni de aquellos datos que el administrador considere que no son necesarios guardar y que ocupan gran cantidad de espacio (como ficheros temporales).

## 10.2. Utilidades.

Para finalizar este capítulo se describen las utilidades gratuitas de copia de seguridad más usadas en Linux, incluidas en las distribuciones de Red Hat.

### 10.2.1. tar.

El mandato **tar** (**archivador de cintas**) sirve para copiar y restaurar información. Se utiliza tanto en la gestión de copias de seguridad, como para empaquetar datos (como se ha descrito en el capítulo 6).

Esta orden incluye funciones para crear, comprobar y restaurar copias completas o incrementales, en uno o en varios volúmenes. En el siguiente cuadro se muestran los formatos de creación, restauración y listado de archivadores mediante **tar** (si no se especifica el nombre del fichero archivador, esta orden utiliza el controlador de la primera unidad de cintas `/dev/st0`).

```
tar cv[M][p][z][f Archivador] Ficheros ...
tar xv[f Archivador] [Ficheros ...]
tar tv[f Archivador]
```

### 10.2.2. AMANDA.

**AMANDA** es una aplicación cliente/servidor creada en la Universidad de Maryland (EE.UU.), que permite hacer copias de seguridad remotas a través de la red. El programa requiere un servidor con gran capacidad de disco y que tenga conectado el dispositivo de almacenamiento, donde los clientes hacen las peticiones de gestión de las copias. Esta arquitectura permite centralizar el proceso de creación y restauración de la seguridad general del centro informático <sup>[1]</sup>. Con las últimas versiones de AMANDA puede utilizarse un servidor Samba para hacer copias de seguridad de ordenadores con Windows NT/2000/XP.

AMANDA se centra en el control de las comunicaciones entre los clientes y el servidor, y en la planificación de las copias de seguridad, ya que utiliza el mandato **tar** para realizar sus operaciones.

La siguiente tabla muestra los características técnicas de los servicios de AMANDA.

Paquetes RPM:	amanda, amanda-server, amanda-client
Servicios:	<b>amandaidx</b> (servidor), <b>amanda</b> (cliente)
Directorio de configuración:	/etc/amanda



## 11. Seguridad.

El último capítulo de este curso de formación se dedica a recopilar unas guías que ayuden al administrador de un sistema Linux a mejorar la seguridad del servidor y de los datos. Para obtener mayor información sobre el tema, el lector puede referirse a la bibliografía adjunta o a los cursos de formación específicos sobre seguridad ofertados por la Universidad de Sevilla.

Como se ha descrito en el capítulo 1, el analista de sistemas debe planificar una política de control que evite –o minimice– los fallos físicos de los dispositivos y los errores humanos. Sin embargo, también debe realizar una previsión en el control de la integridad de los datos, ante posibles ataques malintencionados.

### 11.1. Tipos de ataques.

Es muy importante que el administrador de la red comprenda la naturaleza de los posibles ataques a la seguridad informática, para así poder tomar las medidas más adecuadas. La siguiente tabla analiza brevemente los tipos de ataques más importantes y las medidas más comunes para intentar evitarlos <sup>[6]</sup>.

<b>Acceso no autorizado:</b>	<p>personas que no tienen permiso para utilizar los servicios del sistema son capaces de conectarse a ellos.</p> <p>El administrador debe especificar cuidadosamente los permisos de acceso a los recursos y los usuarios autorizados para cada uno de ellos.</p>
<b>Aprovechamiento de los fallos de programación:</b>	<p>algunos programas, servicios de red e incluso componentes del sistema operativo no han sido diseñados originalmente teniendo en cuenta una seguridad elevada; son inherentemente vulnerables a los ataques, como por ejemplo, los servicios remotos del tipo BSD (<b>rlogin</b>, <b>rexec</b>, etc.).</p> <p>La mejor manera de protegerse contra este tipo de ataques consiste en deshabilitar los servicios vulnerables, encontrar alternativas viables e instalar parches de seguridad.</p>
<b>Denegación de servicio:</b>	<p>estos ataques causan que el servicio o programa deje de funcionar o impide que los clientes puedan utilizarlo. Pueden ser realizados enviando datagramas cuidadosamente preparados para causar que las conexiones de red fallen, o bien tratar de enviar órdenes contra un programa para conseguir que se vuelva muy ocupado o que pare su funcionamiento.</p> <p>Se debe impedir que el tráfico sospechoso pueda alcanzar los servicios de la máquina, conociendo y aprendiendo los detalles de los métodos de ataque.</p>

<b>Suplantación de identidad:</b>	<p>una máquina o aplicación simula las acciones realizadas de otra, haciéndose pasar por una máquina conocida.</p> <p>Para protegerse contra este tipo de ataques, debe verificarse la autenticidad de los datagramas y órdenes recibidos, rechazar las conexiones de direcciones de origen no válidas o introducir datos impredecibles en los mecanismos de control de la conexión (números de secuencias TCP, asignación dinámica de puertos, etc.).</p>
<b>Indiscreción:</b>	<p>en redes de difusión como Ethernet, el atacante puede usar un programa para “escuchar” la red y capturar los datos no destinados a él., obteniendo información privilegiada (usuarios, claves, etc.).</p> <p>Para protegerse contra este tipo de amenazas, debe evitarse el uso de tecnologías de red con difusión e imponer el uso de encriptación de los datos.</p>

Planificar cuidadosamente el diseño de la red empresarial ayuda en gran medida a la protección de los datos contra ataques exteriores. Es conveniente crear una red privada con acceso restringido y controlado a Internet mediante un cortafuegos.

De igual modo, todos los servidores –y en especial el citado cortafuegos– tienen que mantener unos altos niveles de seguridad.

## **11.2. Utilidades de seguridad.**

Se describen a continuación las características de algunas de las utilidades gratuitas de seguridad más utilizadas en los sistemas operativos Linux.

### **11.2.1. TCP-Wrappers, Xinetd.**

**Xinetd** es la evolución del metaservicio de red clásico de Unix, conocido como **inetd**. Este servicio es el encargado de recoger peticiones de conexión y dirigirlas hacia el servicio adecuado, tales como FTP, Telnet, IMAP, POP, SWAT, etc.

La siguiente tabla muestra los requisitos del servicio.

<b>Paquetes RPM:</b>	xinetd
<b>Servicios:</b>	<b>xinetd</b>
<b>Fichero de configuración principal:</b>	/etc/xinetd.conf
<b>Directorio de configuración de servicios:</b>	/etc/xinetd.d

Por otro lado, **TCP-Wrappers** provee mecanismos de control de acceso al resto de servicios de red, permitiendo crear listas donde se indican qué máquinas con permiso de acceso a cada uno de los servicios soportados. La tabla posterior indica los requisitos de este programa.

Paquetes RPM:	<code>tcp_wrappers</code>
Servicios:	<code>tcpd</code>
Fichero de control de accesos permitidos :	<code>/etc/hosts.allow</code>
Fichero de control de accesos denegados:	<code>/etc/hosts.deny</code>

Conviene hacer notar que TCP-Wrappers no realiza ninguna previsión por defecto, si no que hay que indicar explícitamente las listas de control de accesos permitidos y de accesos denegados. Normalmente, se niega el acceso desde todos los ordenadores a todos los servicios –en el fichero `/etc/hosts.deny`– y se activan los servicios permitidos a los indicados en la lista de `hosts.allow`.

### 11.2.2. OpenSSH, OpenSSL.

El **Protocolo de Intérprete Seguro (SSH)** se utiliza para realizar conexiones seguras y codificadas, sustituyendo a los servicios clásicos de Unix (`telnet`, `ftp`, `rlogin`, `rsh`, etc). La codificación se realiza mediante una biblioteca de programación conocida como **Capa de Conexión Segura (SSL)**.

El protocolo SSH presenta los siguientes beneficios <sup>[2]</sup>:

- El cliente verifica periódicamente la conexión al mismo servidor.
- La información de autenticación se transmite codificada, usando métodos de claves pública y privada.
- Todos los datos se transmiten fuertemente codificados para evitar la escucha.
- El cliente puede lanzar aplicaciones gráficas de forma segura a través de la red.

**OpenSSL** es una implementación gratuita de las bibliotecas SSL, que soporta varios tipos de codificación de paquetes. De igual modo, **OpenSSH** es una versión de libre distribución del protocolo SSH. La siguiente tabla muestra los requisitos de instalación de OpenSSH.

Paquetes RPM:	<code>openssl, openssh, openssh-server, openssh-clients</code>
Servicios:	<code>sshd</code>
Directorio de configuración:	<code>/etc/ssh</code>
Configuración de usuarios:	<code>~/.ssh</code>

### 11.2.3. Tripwire.

**Tripwire** es un programa que –ejecutado en el **cron** del sistema– monitoriza la integridad de los ficheros y directorios críticos del sistema, registrando sus cambios y generando alertas de seguridad. Tripwire puede usarse como método local para la detección de intrusos o de fallos de funcionamiento. Sus requisitos se muestran en la siguiente tabla.

Paquetes RPM:	tripwire
Fichero de configuración:	/etc/tripwire/tw.cfg
Fichero de control:	/etc/tripwire/tw.pol

La base de datos de Tripwire almacena información para cada nodo indicando: nombre, tipo de nodo, número de i-nodo, permisos, UID del propietario, GID, fecha de modificación, etc.

### 11.2.4. Netfilter.

El filtrado de paquetes es un proceso que permite decidir los paquetes de datagramas que han de ser procesados por el ordenador y los que deben ser descartados y eliminados.

El componente principal de **Netfilter** es el servicio **iptables**, que constituye la evolución para núcleos 2.4 de los programas **ipchains** –usado en núcleos 2.2– e **ipfwadm**, y permite crear conjuntos de reglas según los siguientes criterios de filtrado <sup>[6]</sup>:

- Tipo de protocolo (TCP, UDP, ICMP, etc.).
- Número de anclaje (*socket*) para TCP y UDP.
- Tipo de datagrama (SYN/ACK, datos, eco ICMP, etc.).
- Direcciones originaria y destinataria de los paquetes de datos.

Las versiones 2.4 del núcleo de Linux contienen 3 tablas internas para el filtrado de paquetes con Netfilter. Cada una de estas tablas cuenta con unas cadenas internas de acciones que se ejecutan sobre cada paquete filtrado. Las tablas internas son:

Tabla	Descripción
filter	Tabla por defecto para la recogida de paquetes de la red. Sus cadenas predefinidos SON: INPUT, OUTPUT Y FORWARD
nat	Tabla usada para modificar los paquetes que crean una nueva conexión.
mangle	Tabla usada para la alteración específica de paquetes.

Netfilter se instala tanto en cortafuegos, para filtrar el tráfico que comunica la red privada con Internet, como en cualquier servidor, para bloquear y verificar sus conexiones. La tabla muestra los requisitos de instalación de esta herramienta.

Paquetes RPM:	<code>iptables</code>
Servicio:	<code>iptables</code>
Formato de ejecución:	<code>iptables [-t <i>Tabla</i>] Orden Cadena Parámetro1 Opción1 ...</code>

### 11.2.5. Sudo.

La cuenta del usuario privilegiado `root` tiene todos los permisos de acceso en un servidor Unix, por lo que debe tener una utilización restringida y mínima, para evitar posibles errores humanos.

Por motivos de seguridad, puede desactivarse la conexión directa del administrador. El mandato `su` se utiliza para ejecutar un intérprete como otro usuario (incluso como `root`), sin embargo la orden `sudo` puede incrementar las prestaciones generales de seguridad.

**Sudo** es un programa que permite al administrador del sistema asignar a ciertos usuarios o grupos la posibilidad de ejecutar mandatos como `root`, registrando las operaciones realizadas.

Las características de Sudo son:

- Ejecución por línea de mandatos.
- Restricción de los mandatos que pueden ser ejecutados por cada usuario o grupo.
- Registra de cada mandato ejecutado o fallido (debe usarse en conjunción con `syslogd`).
- El usuario debe autenticarse y el programa le asigna un billete de acceso temporal renovable, lo que evita tener que utilizar un intérprete de mandatos como usuario privilegiado.

Las características de configuración de Sudo se muestran en la siguiente tabla.

Paquetes RPM:	<code>sudo</code>
Fichero de configuración:	<code>/etc/sudoers</code>
Formato de ejecución:	<code>sudo {-p Mens} [-a Autentif] -u [Usuario UID] Mandato</code>

### 11.2.6. chkrootkit.

Una vez que un atacante ha encontrado una vulnerabilidad en un sistema, la utiliza para entrar como usuario, posteriormente intenta ganar privilegios hasta obtener acceso como usuario **root** y por último intenta instalar programas –denominados *rootkits*– que realizan las siguientes funciones <sup>[15]</sup>:

- Crear puertas traseras para entrar posteriormente en el sistema.
- Eliminar o modificar los registros históricos del sistema.
- Modificar o reemplazar las herramientas del sistemas para evitar su detección.
- Monitorizar el tráfico de la red o las pulsaciones de teclas.
- Lanzar ataques sobre otros sistemas desde el ordenador “capturado”.

El programa **chkrootkit** localiza actualmente 44 tipos de *rootkits* y puertas traseras en ordenadores Unix. Esta aplicación no viene incluida con las versiones de Red Hat, aunque puede ser descargada gratuitamente desde Internet <sup>[ix]</sup>.

Para obtener una mayor eficacia, **chkrootkit** debe ser lanzado periódicamente en el **cron** del sistema para comprobar su estado. Por otro lado, este programa ejecuta varios mandatos del sistema, que previamente deben ser salvados en un soporte externo (**awk**, **cut**, **echo**, **egrep**, **find**, **head**, **id**, **ls**, **netstat**, **ps**, **strings**, **sed** y **uname**).

### 11.2.7. SNORT.

**SNORT** es una herramienta de seguridad –no incluida en Red Hat– que permite realizar las siguientes funciones <sup>[x]</sup>:

- Escuchar la red y mostrar el contenido del flujo de datos.
- Registrar en disco de los paquetes capturados.
- Analizar el tráfico de la red para realizar procesos de detección de intrusos.

La función principal de **SNORT** es la de detectar intrusos en la red privada, presentando distintos tipos de alertas de seguridad, tanto al **syslogd** como a puertos específicos de otras máquinas o mediante un cliente Samba.

El siguiente cuadro muestra un ejemplo de la ejecución de **SNORT** como detector de intrusos en la red.

```
./snort -dev -l ./log -h 192.168.1.0/24 -c snort.conf
```

## 12. Referencias.

1. Red Hat Inc.: “*Red Hat 8.0: The Official Red Hat Linux System Administration Primer*”, 2.002.
  2. Red Hat Inc.: “*Red Hat 8.0: The Official Red Hat Linux Reference Guide*”, 2.002.
  3. Red Hat Inc.: “*Red Hat 8.0: The Official Red Hat Linux Customization Guide*”, 2.002.
  4. Mike G, trad. G. Rodríguez Alborich: “*Programación en BASH – COMO de Introducción*”. 2.000.
  5. M. Cooper: “*Advanced Bash-Scripting Guide, v1.7*”. Linux Documentation Project, 2.003.
  6. O. Kirch, T. Dawson: “*Guía de Administración de Redes en Linux*”. O’Reilly, 2.000. Trad. Proyecto LuCAS de HispaLiNux, 2.002.
  7. G. Mourani: “*Securing and Optimizing Linux: The Ultimate Solution, v2.0*”. Open Network Architecture Inc., 2.001.
  8. E. Parreño Gómez: “*Detección de Espías en una Red Ethernet, v1.0*”. 2.002.
  9. D. Barreña Molina y otros: “*Proyecto RHODAS: Migración a estaciones de trabajo Linux para usuario final en el MAP*”. Ministerio de Administraciones Públicas (España), 2.002.
  10. D. Quinlan, trad. I. Barrientos: “*Estructura del Sistema de Archivos de Linux*”. 1.996.
  11. R. M. Gómez Labrador: “*Servicios de Internet para Linux*”. Secretariado de Formación Permanente del PAS (Universidad de Sevilla), 1.999.
  12. R. M. Gómez Labrador: “*Sistemas Operativos en Red: Introducción a Linux*”. Secretariado de Formación Permanente del PAS (Universidad de Sevilla), 1.998.
  13. L. Virzenius, J. Oja, S. Stafford: “*The Linux System Administration Guide, v0.7*”. 2.001.
  14. SNIA: “*CIFS Technical Reference, v1.0*”, 2.002.
  15. C. Prosis, S. U. Shah: “*At the Root of Rootkits*”. C|Net Builder, 2.001.
- 
- i. Red Hat Inc.: <http://www.redhat.com/>
  - ii. Linux OnLine!: <http://www.linux.org/>
  - iii. The Linux Documentation Project (TLDP): <http://www.tldp.org/>
  - iv. Proyecto HispaLinux (LDP-ES): <http://www.hispalinux.es/>
  - v. Norma para la Jerarquía en Sistemas de Archivos (FHS): <http://www.pathname.com/fhs/>
  - vi. Samba: <http://www.samba.org/>
  - vii. CUPS: <http://www.cups.org/>
  - viii. AMANDA: <http://www.amanda.org/>
  - ix. chkrootkit: <http://www.chkrootkit.org/>
  - x. SNORT: <http://www.snort.org/>
  - xi. Proyecto GNU: <http://www.gnu.org/>
  - xii. Servicio de Seguridad IRIS-CERT: <http://www.rediris.es/cert/>
  - xiii. RPMFind.net: <http://www.rpmfind.net/>
  - xiv. SourceForge.net: <http://www.sourceforge.net/>
  - xv. Freshmeat: <http://www.freshmeat.net/>

## APÉNDICE A. Cuestionario del curso.

**1. ¿Qué fichero contiene las claves codificadas de los usuarios locales?**

- a) /etc/passwd
- b) /etc/security/passwd
- c) /etc/shadow
- d) /etc/login.d

**2. ¿Qué tipo de módulo PAM se usa para comprobar las credenciales del usuario?**

- a) auth
- b) account
- c) password
- d) session

**3. En BASH ¿cuál es el símbolo para redirigir la salida de error a un fichero?**

- a) <
- b) >
- c) >>
- d) 2>

**4. ¿Cuál es el protocolo usado para obtener remotamente la dirección de red de un cliente?**

- a) DHCP.
- b) NIS.
- c) LDAP.
- d) CIFS.

**5. ¿Cuál es el fichero de control para la primera partición del disco maestro conectado en la interfaz IDE primaria?**

- a) /dev/sda1
- b) /dev/hda1
- c) /dev/st1
- d) /dev/eth1

**6. ¿Qué valor se asigna a la variable `VAR` al ejecutar bajo BASH la siguiente orden? `VAR=`grep "^$LOGNAME:" /etc/passwd|cut -f3 -d:``**

- a) La cadena de caracteres: `"grep "^$LOGNAME:" /etc/passwd|cut -f3 -d:",` pero sustituyendo el valor de la variable `LOGNAME`.
- b) El UID del usuario.



- c) La lista de los usuarios del sistema que tienen activada la variable `LOGNAME`.
- d) Se produce un error de ejecución.

**7. ¿Cuál de los siguientes tipos de sistemas de archivos permite montar remotamente un directorio compartido por un servidor Samba?**

- a) NFS.
- b) VFS.
- c) CIFS.
- d) EXT3.

**8. ¿Qué técnica de copias de seguridad necesita restaurar un mayor número de archivadores?**

- a) Completa.
- b) Incremental.
- c) Diferencial.
- d) RAID.

**9. ¿Qué extensión tienen los paquetes de programas de Red Hat?**

- a) rpm.
- b) tgz.
- c) bz2.
- d) deb.

**10. ¿Qué mandato debe utilizarse para editar la lista de ejecución periódica de tareas?**

- a) `sysctl`
- b) `batch`
- c) `crontab`
- d) `at`

**11. ¿Cuál de las siguientes afirmaciones es correcta en relación con el servicio CUPS?**

- a) Sirve para realizar estadísticas remotas de los registros históricos a través de *web*.
- b) Es una biblioteca de programación que implementa una capa segura del tipo SSL.
- c) Utiliza el protocolo IPP para gestionar los trabajos de impresión.
- d) Forma parte de la utilidad de filtrado de paquetes NetFilter.

**12. ¿Para qué se utiliza el mandato `lsOf`?**

- a) Para ver paginadamente el contenido de un fichero.
- b) Para decodificar el contenido de los ficheros históricos del sistema.
- c) Para mostrar el contenido de un árbol de directorios.
- d) Para listar los ficheros abiertos y las conexiones de red establecidas.

**13. Si el usuario `u1` pertenece al mismo grupo que `u2` y los permisos de su directorio personal `~u1` son de 2710, ¿cuál de las siguientes operaciones realizadas por `u2` puede ser ejecutada sin errores?**

- a) Listar el contenido del directorio `~u1`.
- b) Listar el contenido del directorio `~u1/d1`, ya que éste tiene los permisos 750.
- c) Eliminar el contenido del fichero `~/u1/f1`, ya que éste tiene permisos 770.
- d) Copiar su fichero `f2` al directorio `~u1`.

**14. ¿Cuál de las siguientes herramientas utiliza un cortafuegos para filtrar la información?**

- a) `sudo`
- b) `openldap`
- c) `tripwire`
- d) `iptables`

**15. ¿Para qué se utiliza el fichero `/etc/sysctl.conf`?**

- a) Para configurar las listas de control de acceso de los ficheros del sistema.
- b) Para configurar los parámetros del núcleo.
- c) Para configurar las variables de entorno de las cuentas de los usuarios.
- d) Para configurar los servicios que se deben iniciar al arrancar la máquina.

## **APÉNDICE B. Respuestas al cuestionario.**

1. c)
2. a)
3. d)
4. a)
5. b)
6. b)
7. c)
8. b)
9. a)
10. c)
11. c)
12. d)
13. b)
14. d)
15. b)

## APÉNDICE C. Ejemplos de programas.

### C.1. variables

Este es un sencillo ejercicio que presenta el contenido de algunas de las variables más importantes utilizadas por el intérprete de mandatos BASH.

```
#!/bin/bash
# variables - listado de variables usadas por BASH.

echo "Camino de búsqueda:      $PATH"
echo "Directorio actual:      $PWD"
echo "Directorio del usuario:  $HOME"
echo "Intérprete de mandatos:  $SHELL"
echo "Tipo de terminal:       $TERM"
echo "Nombre del usuario:      $LOGNAME"
echo "Idioma:                  $LANG"
echo "Punto indicativo:       $PS1"
```

### C.2. especiales

Como complemento al programa anterior, se presentan algunas de las variables especiales de la BASH. El programa puede recibir varios parámetros en la línea de entrada y describe el uso de la orden interna **shift** para desplazarlos.

```
#!/bin/bash
# especiales - listado de variables especiales de BASH.

echo "PID del programa:          $$"
echo "Nombre del programa:       $0"
echo "Primer parámetro:          $1"
echo "Segundo parámetro:         $2"
echo "Todos los parámetros:       $"
echo "Número de parámetros:      $#"
```

shift

```
echo "Número de parámetros tras shift:  $"
echo "Todos los parámetros tras shift:  $"
```

### C.3. dircom

Programa comentado compatible con varios dialectos de Unix, que se utiliza para listar la información completa de un mandato. La última orden del programa es la que realiza la operación, el resto de líneas realizan las comprobaciones de ejecución.

```

# S.O.: AIX 3.x
# S.O.: Solaris 2.x
# S.O.: Linux 2.x
# dircom - Indica el camino completo de un mandato.
# Uso: dircom [?] | <mandato>
#           ?: ayuda.
# Ramón Gómez - Enero 1.994.
#           Versión 2: Octubre 1.996. Compatible Solaris y Linux.

if [ "x$" = "x?" ];
then
    echo "Uso:          `basename $0` [?] | mandato"
    echo "Versión:      2.0: R.G.L. - Octubre 1.996."
    echo "Propósito: Indica el camino completo de un mandato."
    exit 0
fi
if [ $# -ne 1 ];
then
    echo "`basename $0`: Parámetro no reconocido." >&2
    echo "Uso: `basename $0` [?] | mandato" >&2
    echo "  ?: ayuda" >&2
    exit 1
fi

if [ "`uname`" = "AIX" ]
then PARAMS="selbd"
else PARAMS="slbd"
fi
ls -$PARAMS `whereis $1` 2>/dev/null

```

## C.4. usuario

Lista información sobre un usuario del sistema, procesando directamente los ficheros del sistema. Se utilizan técnicas de entrecomillado, tuberías, listas de mandatos y estructuras selectivas.

```

#!/bin/bash
# usuairo - lista información de un usuario.

if [ $# != 1 ]
then
    echo "$0: Parámetro erróneo." >&2
    exit 1
fi
TEMPORAL=`grep "^$1:" /etc/passwd 2>/dev/null`
if [ $? != 0 ]
then
    echo "$0: El usuario \"$1\" no existe." >&2
    exit 2
fi
USUARIO=`echo $TEMPORAL | cut -f1 -d:`
echo "Nombre de usuario: $USUARIO"
echo -n "Identificador (UID): "
echo $TEMPORAL | cut -f3 -d:
echo -n "Nombre del grupo primario: "
GID=`echo $TEMPORAL | cut -f4 -d:`

```

```

grep ":$GID:" /etc/group | cut -f1 -d:
DIRECTORIO=`echo $TEMPORAL | cut -f6 -d:`
echo "Directorio personal: "
ls -ld $DIRECTORIO
echo -n "¿Es accesible? "
if [ -x "$DIRECTORIO" ]; then echo "sí"; else echo "no"; fi
echo "Conectado desde: "
( who|grep "^$USUARIO " ) || echo "no conectado"

```

## C.5. comprus

Realiza un listado completo de la información de usuarios del sistema, puede ser ejecutado en varios dialectos Unix y recoge información tanto de los ficheros del sistema, como remotamente mediante NIS o LDAP.

Se hace utilización de varias funciones que permiten estructurar el programa y conseguir una lectura más comprensible.

```

# S. O.: AIX 3.x, 4.x
# S. O.: Solaris 2.x
# S. O.: RedHat Linux 7.x
# comprus - comprueba la existencia de usuarios en las listas, en el
#           archivo de claves (normal y NIS), su directorio y cuotas.
#           Uso: comprus ? | cadena
#           ?: ayuda.
# Ramón Gómez - Marzo 1.992.
#           Revisión 1.1: Octubre 1.992.
#           Versión 2: Diciembre 1.992.
#           Revisión 2.1: Junio 1.993.
#           Revisión 2.2: Septiembre 1.995. Nuevos grupos.
#           Revisión 3: Septiembre 1.996. Solaris y nuevos grupos.
#           Revisión 3.1: Septiembre 1.997. Nuevos grupos.
#           Revisión 3.2: Noviembre 1.997. Cuotas.
#           Revisión 3.3: Septiembre 1.998. Comprueba el directorio de listas.
#           Versión 4: Septiembre 1.998. Búsqueda en NIS y versión Linux.
#           Versión 4.1: Septiembre 2.002. Búsqueda en LDAP para Linux.

# Rutina de impresión.
# Parámetros:
#           1 - cadena a buscar.
#           2 - texto de cabecera.
#           3 - archivo de búsqueda.
salida ()
{
    if egrep "$1" $3 >$TMPGREG 2>/dev/null; then
        echo "           $2:"
        cat $TMPGREG
    fi
}

# Rutina de búsqueda en NIS.
# Parámetros:
#           1 - cadena a buscar.
#           2 - texto de cabecera.
salidayp ()
{

```

```

        ypcat passwd | egrep "$1" \
        | awk 'BEGIN {FS=":";} \
                {print $1":!:"$3": "$4": "$5": "$6": "$7; }' \
                >$TMPGREG 2>/dev/null

    if [ -s $TMPGREG ]; then
        echo "        $2:"
        cat $TMPGREG
    fi
}

# Rutina de búsqueda en LDAP.
# Parámetros:
#     1 - cadena a buscar.
#     2 - texto de cabecera.
salidaldap ()
{
    smbldap-usershow.pl \* 2>/dev/null | egrep "$1" \
    | egrep "^(dn|gecos):" >$TMPLDAP

    if [ -s $TMPLDAP ]; then
        echo "        $2:"
        cat $TMPLDAP
    fi
    rm -f $TMPLDAP
}

TMPGREG=/tmp/grep$$
TMPLDAP=/tmp/ldap$$
DIRLISTAS=/home/cdc/listas

if [ "x$*" = "x?" ]
then
    echo "
Uso:      `basename $0` ? | cadena
Versión:   4.1: R.G.L. - Septiembre 2.002.
Propósito: `basename $0`: Búsqueda de usuarios (listas, claves, NIS, LDAP).
           cadena: expresión regular a buscar.

Ejemplos:
`basename $0` cadena    -> lista usuarios con 'cadena' en listas o en su
\"login\".
`basename $0` cadena:   -> lista usuarios cuyo \"login\" acaba en
'cadena'.
`basename $0` \"APELL1 APELL2\" -> lista usuarios con apellidos APELL1
APELL2.
"
        exit 0
    fi
    if [ $# -ne 1 ]
    then
        echo "`basename $0`: Falta parámetro o parámetros no reconocidos.
Uso: `basename $0` ? | cadena
     ?: ayuda" >&2
        exit 1
    fi
    echo

    for i in $DIRLISTAS/*.lista
    do
        salida "$1" "`basename $i | sed 's/.lista//`'" "$i"
    done
    if [ "x`ls /var/yp/binding`" = "x" ]
    then salida "$1" "passwd" "/etc/passwd"

```

```

else salidayp "$1" "NIS"
fi
salidaldap "$1" "LDAP"
if [ -s "$TMPGREG" ];
then echo "      Directorios:"
    ls -ld `cut -f6 -d: $TMPGREG`
    if [ "$LOGNAME" = "root" ];
    then echo "      Fecha de caducidad:"
        for US in `cut -f1 -d: $TMPGREG`
        do
            case "$SO" in
                AIX)    echo "$US:    `lsuser -ca expires $US|fgrep -v
"#"|cut -f2 -d:`" ;;
                SunOS) echo "$US:    `logins -aol $US|cut -f7 -d:`" ;;
                Linux) echo "$US:    `chage -l $US|grep Account|cut -f2 -
d:`" ;;
                *) ;;
            esac
        done
        echo "      Capacidad (en KB):"
        for US in `cut -f1 -d: $TMPGREG`
        do
            du -ks `grep "^$US:" $TMPGREG | cut -f6 -d:`
            quota -v $US | tail -1
        done
    fi
    rm -f $TMPGREG
    echo
fi

```

## C.6. crearus

Se presentan 2 ejemplos de programas para la creación interactiva de cuentas de usuarios, manteniendo también unas listas actualizadas.

El programa considera que las cuentas de los usuarios se almacenan en el directorio personal */home/Grupo/Usuario* y los datos básicos de la persona se guardan en listas independientes para cada grupo primario, en archivos del tipo */home/cdc/listas/Grupo.lista*.

```

#!/bin/bash
# crearus - programa de creación de cuentas de usuarios.
# Formato: crearus
# Versión 1.0 - Marzo 2.003
# Autor: Ramón Gómez

echo -n "Nombre completo: "
read NOMBRE
echo -n "D.N.I.: "
read DNI
echo -n "Grupo principal: "
read GRUPO
echo -n "Otros grupos: "
read OTROSGRUPOS
echo -n "Nombre de usuario:"
read USUARIO
echo -n "Fecha caducidad: "

```



```

read CADUC
DIRPERS="/home/$GRUPO/$USUARIO"
LISTA="/home/cdc/listas/$GRUPO.lista"

useradd -c "$NOMBRE" -g $GRUPO -G "$OTROSGRUPOS" -k /etc/skel -s /bin/bash
-md $DIRPERS -e $CADUC -p "$DNI" $USUARIO
if [ $? -ne 0 ]; then
    echo "$0: error al crear la cuenta de \"$USUARIO\"." >&2
    exit 1
fi
echo "$DNI:$USUARIO:$NOMBRE" >>$LISTA
chmod -R go-rwx $DIRPERS

```

La segunda revisión del programa realiza una serie de comprobaciones básicas y control de errores que deben cumplirse antes de realizar la orden principal de creación de la cuenta.

```

#!/bin/bash
# crearus - programa de creación de cuentas de usuarios.
# Formato: crearus
# Autor: Ramón Gómez
# Versión 1.0: Marzo 2.003.
# Versión 1.1: Marzo 2.003. Realiza comprobaciones básicas.

DIRLISTAS="/home/cdc/listas"

echo "Creación de cuentas de usuarios v1.1"
echo
echo -n "Nombre completo: "
read NOMBRE
echo -n "D.N.I.: "
read DNI
if egrep "^$DNI:" $DIRLISTAS/*.lista; then
    echo "$0: El DNI \"$DNI\" ya existe en las listas." >&2
    exit 2
fi
echo -n "Grupo principal: "
read GRUPO
if ! egrep "^$GRUPO:" /etc/group >/dev/null; then
    echo "$0: El grupo \"$GRUPO\" no existe." >&2
    exit 3
fi
echo -n "Otros grupos: "
read OTROSGRUPOS
for GR in `echo "$OTROSGRUPOS"|tr ',' ' '`; do
    if ! egrep "^$GR:" /etc/group >/dev/null; then
        echo "$0: El grupo \"$GR\" no existe." >&2
        exit 3
    fi
done
echo -n "Nombre de usuario:"
read USUARIO
if egrep "^$USUARIO:" /etc/passwd >/dev/null; then
    echo "$0: El usuario \"$USUARIO\" ya existe." >&2
    exit 4
fi
if [ ${#USUARIO} -lt 5 -o -n "${USUARIO//[a-z]}" ]; then
    echo "$0: Nombre de usuario \"$USUARIO\" erróneo." >&2
    exit 4

```

```

fi
echo -n "Fecha caducidad:  "
read CADUC
DIRPERS="/home/$GRUPO/$USUARIO"
LISTA="$DIRLISTAS/$GRUPO.lista"

useradd -c "$NOMBRE" -g $GRUPO -G "$OTROSGRUPOS" -k /etc/skel -s /bin/bash
-md $DIRPERS -e $CADUC -p "$DNI" $USUARIO
if [ $? -ne 0 ]; then
    echo "$0: error al crear la cuenta de \"$USUARIO\"." >&2
    exit 1
fi
echo "$USUARIO:$DNI:$NOMBRE" >>$LISTA
chmod -R go-rwx $DIRPERS

```

## C.7. borrar.cuentas

Permite la eliminación de las listas y el borrado del contenido de varias cuentas de usuarios del sistema. Es un programa compatible para IBM AIX, Sun Solares y Red Hat Linux.

```

# S. O.: AIX 4.x
# S. O.: Solaris 2.x
# S. O.: RedHat Linux 7.x
# borrar.cuentas - Borra cuentas y directorios completos de mltiples
usuarios.
#
#           No borra cuentas de usuarios administradores.
#       Uso: borrar.cuentas ? | [-n] usuario [usuario ...]
#           ?: ayuda.
#           -n: borra sin preguntar
#       NOTA: Este programa puede ejecutarlo 'root' y usuarios con
#           permiso de borrado de dichos directorios.
# Ramón Gómez - Enero 1.994.
#       Revisión 1.1: Octubre 1.997. Ejecutable en AIX y en Solaris.
#       Revisión 2: Septiembre 1.998. Compatible con "comprus" y rutina
"borrar_de_listas".
#       Revisión 2.1: Septiembre 1.998. Corrección fallo en borrado de
listas con nombre
#
#           de usuario en el 1er. campo.
#       Revisión 2.2: Septiembre 2.002. Versión para Linux.
#
#           Borrado de cuentas de LDAP en Linux.

HOME CDC="/home/cdc"

### Borra la linea del usuario de la lista correspondiente.
#       Parametros:
#           1 - Nombre de usuario
borrar_de_listas ()
{
    cd $HOME CDC/listas
    umask 117
    for i in `egrep "^$1:" *.lista | cut -f1 -d:`
    do
        TMPLISTA=$i.$$
        if egrep -v "^$1:" $i >$TMPLISTA;
        then
            echo "$1 borrado de $i."

```

```

        mv -f $TMPLISTA $i
    fi
done
}

### Imprime en pantalla sin salto de linea (echo -n).
# Parametros:
#          * - Cadena(s) a imprimir.
echon ()
{
    if [ "x`echo -n`" = "x-n" ]
    then echo "$*\c"
    else echo -n "$*"
    fi
}

### PROGRAMA PRINCIPAL

if [ "x$*" = "x?" ];
then
    echo "
Uso:      `basename $0` ? | [-n] usuario [usuario ...]
Versión:   2.2: Ramón Gómez - Septiembre 2.002
Propósito: Borrar cuentas y directorios de varios usuarios.
            -n: borra sin preguntar.
"
    exit 0
fi
if [ $# -lt 1 ];
then
    echo "`basename $0`: Falta(n) parámetro(s).
Uso: `basename $0` ? | [-n] usuario [usuario ...]
     ? : ayuda" >&2
    exit 1
fi
if [ "`whoami`" != "root" ];
then
    echo "`basename $0`: Sólo ejecutable por usuario privilegiado." >&2
    exit 2
fi

PREG="s"
if [ "x$1" = "x-n" ];
then
    PREG="n"
    shift
fi

while [ $# -gt 0 ]
do
    DATOS=`grep "^$1:" /etc/passwd | cut -f1,6 -d:`"
    if [ -z "$DATOS" ]
    then echo "$1 no tiene cuenta."
    else
        if [ ! -z "`groups $1 | fgrep personal`" ]
        then
            echo "$1 es usuario administrador. No se borrara la cuenta."
        else
            comprus "^$1:"
        fi
    fi
done

```

```

        if [ "$PREG" = "s" ];
        then
            echon "¿Eliminar la cuenta de $1? (s/n) ... [n]: "
            read RESP
        else
            RESP="s"
        fi
        if [ "$RESP" = "s" -o "$RESP" = "S" ];
        then
            DIR=`echo $DATOS | cut -f2 -d:`
            case "`uname`" in
                AIX)    if rmuser -p $1  && echon "$1  sin cuenta.      "
                        then rm -r $DIR  && echo "Directorio  $DIR
borrado."
                                fi ;;
                SunOS) userdel -r $1 && echo "$1  sin cuenta.
$DIR borrado." ;;
                Linux) userdel -r $1 && echo "$1  sin cuenta.
$DIR borrado."
                                smbldap-userdel.pl $1 2>/dev/null && echo "$1
borrado de LDAP."
                                ;;
                esac
            if [ -n "$DIRMAIL" ]; then
                rm -f $DIRMAIL/$1  &&  echo "$DIRMAIL/$1 borrado."
            fi
            borrar_de_listas "$1"
        fi
    fi
fi
shift
done

```

## C.8. borrar.cuentas

Este programa puede ser utilizado por el administrador para cambiar la fecha de caducidad de un conjunto de cuentas de usuarios. Asimismo, genera un fichero de salida con información sobre los cambios realizados.

```

# S.O.: AIX 3.x
# S.O.: Solaris 2.x
# S.O.: RedHat 7.x, 8.x
# renovar.cuentas - Cambia la fecha de caducidad de varias cuentas.
#                  Genera un fichero de salida con los datos de las
#                  cuentas renovadas.
#      Uso: renovar.cuentas ? | fecha_caduc usuario [usuario ...]
#      ?: ayuda.
#      Nombre del fichero de salida: renovar.<máquina><dd><mm>
#      Formato de las líneas del fichero de salida:
#              fecha_caducidad usuario
#      NOTA: Este programa sólo puede ser ejecutado por 'root'.
# R. G. L. - Marzo 1.999.

SO=`uname`
HOME CDC=/home/cdc

```

```

if [ "x$*" = "x?" ];
then
    echo "
Uso:      `basename $0` ? | fecha_caduc usuario [usuario ...]
Versión:   1.0: R.G.L. - Marzo 1.999
Propósito: Cambiar fechas de caducidad de varias cuentas."
    exit 0
fi
if [ $# -lt 2 ];
then
    echo "
`basename $0`: Falta(n) parámetro(s).
Uso: `basename $0` ? | fecha_caduc usuario [usuario ...]
      ? : ayuda" >&2
    exit 1
fi
if [ "`whoami`" != "root" ];
then
    echo "`basename $0`: Sólo ejecutable por usuario privilegiado." >&2
    exit 2
fi
FECHA=$1
shift
case "$SO" in
    AIX)      PROG="chuser expires="
            ;;
    SunOS)    PROG="usermod -e "
            ;;
    Linux)    PROG="usermod -e "
            ;;
esac
FICHSALE=$HOME/CDC/temp/renovar.`hostname`.`date +%d%m`
echo "      RENOVAR CUENTAS v1.0                      Ramón Gómez - Abril 1.999"
echo
while [ $# -gt 0 ]
do
    if id "$1" 2>/dev/null;
    then
        eval "${PROG}\`${FECHA}\` \`${1}\`" && echo "${FECHA} $1" >>FICHSALE
    else
        echo "$1 no tiene cuenta." >&2
    fi
    shift
done

```

## C.9. dnis.repes

Revisa las listas de usuarios y presenta aquellas que contengan números de DNI repetidos. Este pequeño programa puede usarse para comprobar la consistencia de las listas, ya que éste debe ser un valor único y distinto para cada persona.

```

#/bin/bash
# dnis.repes - Muestra los DNI que están repetidos en las listas.
# Formato:      dnis.repes [?] | [FichPerfiles]
# Ramón Gómez - Octubre 1.998.

```

```

if [ "x$" = "x?" ]
then
    echo "
Uso:      `basename $0` [?] | [FichPerfiles]
Versión:   1.0: Ramón Gómez - Octubre 1.998.
Propósito: Muestra los DNIs que están repetidos en las listas."
    exit 0
fi

DIRLISTAS="/home/cdc/listas"
LISTAS="*.lista"

cd $DIRLISTAS
for DNI in `cut -sf2 -d: $LISTAS | sort -n | uniq -d`; do
    egrep ":$DNI:" $LISTAS 2>/dev/null
done

```

## C.10. listar.expiracion

Se propone como ejercicio retocar el siguiente programa para adaptarlo a las versiones 7 y 8 del Linux de Red Hat. El *“script”* debe presentar una lista con todos los usuarios del sistema y el UID, el grupo primario, el directorio y la fecha de caducidad de cada uno de ellos.

```

# S. O.: AIX 3.2
# S. O.: Solaris 2.[46]
# listar.expiracion - Lista las fechas de caducidad de todas las cuentas,
#                     junto con sus directorios.
#     Uso: listar.expiracion [?]
#     ?: ayuda.
#     NOTA: Este programa sólo puede ejecutarlo 'root'.
# Ramón Gómez - Enero 1.996.
#     Versión 2.0: Marzo 1.998. Actualización Solaris.
#     Versión 2.1: Marzo 1.999. Compatibilidad entre Solaris y AIX.

HOME CDC="/home/cdc"
tmpus=$HOME CDC/temp/expir$$

if [ "$LOGNAME" != "root" ];
then
    echo "\n`basename $0`: $LOGNAME no tiene permiso de ejecución."
    exit 1
fi
if [ "x$" = "x?" ];
then
    echo "\nUso:      `basename $0` [?]"
    echo "Version:   2.1: R.G.L. - Marzo 1.999."
    echo "Propósito: Lista las fechas de caducidad de todas las cuentas.\n"
    exit 0
fi

echo "\n\tLISTAR FECHAS DE CADUCIDAD DE TODAS LAS CUENTAS v2.1      R.G.L.
(3/99)\n"
case "`uname`" in
    AIX)    lsuser -ca id pgrp home expires ALL | fgrep -v "#" >$tmpus
            ;;

```

```

SunOS) logins -xao | cut -f1-3,6,14 -d: >$tmpus
    ;;
*)      echo "\n`basename $0`: Variable de entorno incorrecta."
        exit 1
    ;;
esac

echo "USUARIO      (ID.)  CADUCIDAD    GR.PRIM.   DIRECTORIO"
echo "-----"
echo "-----"
awk -F: '{
    printf "%-8s (%4s)  %10s  %-8s  %s\n", $1,$2,$5,$3,$4;
}' $tmpus
rm -r $tmpus

```

## C.11. borrado.diario

Este programa administrativo elimina los ficheros temporales que se han quedado obsoletos y debe ejecutarse en el cronológico del sistema para realizar esta operación todas las noches.

```

# S.O.: AIX 3.x, 4.x
# S.O.: Solaris 2.x
# S.O.: RedHat 7.x, 8.x
# borrado.diario: Borrado diario de ficheros temporales.
#      Nota: Este programa debe incluirse en el "cron" con propiedad de
#      "root".
#      Uso: borrado.diario      (conviene redirigir las salidas).
# Versión 1:      - Adaptación del programa "skulker" de AIX.
# Ramón Gómez - Junio 1.994.
# Versión 2: Noviembre 1.996. Compatible Solaris.
# Versión 2.1: Marzo 2.000.  Borrar los ficheros .pine-debug* de las
# cuentas.

export LANG=${LANG:-"es"}
date=`date`
uname=`uname -nm`
case `uname` in
    SunOS) HOM=`/bin/df|cut -f1 -d"("|sed 's/ //g'|grep "home"|tr "\n" "
` ;;
    AIX)   HOM=`lsfs -c | cut -f1 -d: | grep "home"` ;;
    *)     HOM=/home ;;
esac

echo "`basename $0` empieza el  $date  en  $uname.\n"

if [ -d /var/spool/structmail ]
then
    echo "Borrar de structmail"
    find /var/spool/structmail -mtime +2 -atime +2 -type f -ls -exec /bin/rm
-f {} \;
fi
if [ -d /var/spool/qdaemon ]
then
    echo "Borrar salidas perdidas de qdaemon".
    find /var/spool/qdaemon -mtime +4 -type f -ls -exec /bin/rm -f {} \;

```

```

fi
if [ -d /usr/lib/lpd/qdir ]
then
    echo "Borrar entradas antiguas de qdir"
    find /usr/lib/lpd/qdir -mtime +4 -type f -ls -exec /bin/rm -f {} \;
fi
if [ -d /var/spool/mqueue ]
then
    echo "Borrar entradas que han quedado en /var/spool/mqueue"
    find /var/spool/mqueue -type f -mtime +2 -ls -exec /bin/rm -f {} \;
fi
if [ -d /tmp ]
then
    echo "Limpiar /tmp"
    find /tmp -mtime +1 -type f -ls -exec /bin/rm -f {} \;
fi
if [ -d /var/tmp ]
then
    echo "Limpiar /var/tmp"
    find /var/tmp -type f -atime +1 -mtime +1 -ls -exec /bin/rm -f {} \;
fi
if [ -d /var/news ]
then
    echo "Eliminar noticias antiguas de /var/news"
    find /var/news -mtime +45 -type f -ls -exec /bin/rm -f {} \;
fi
# Limpiar las cuentas de los usuarios. Se usa -xdev para evitar la búsqueda
# en otros sistemas de archivos montados bajo ${HOM}. En algunos casos
# puede resultar interesante especificar el resto de sistemas de archivos
# con cuentas de usuarios.
echo "Limpiar las cuentas de los usuarios en ${HOM}."
find / /usr /var /tmp ${HOM} \
    \( \
        \( \
            -name "*.bak" -o -name ".*.bak" -o -name core -o \
            -name a.out -o -name "...*" -o -name ed.hup -o \
            -name ".pine-debug*" \
            -atime +3 -mtime +3 -type f \
            -o \
            \( \
                -name proof -o -name galley \
                -atime +1 -mtime +1 -type f ! -perm -0200 \
            \) \
        \) -xdev -ls -exec /bin/rm -f {} \;
echo "Limpiar los .putdir"
for i in `find / /usr /var /tmp ${HOM} \
    -xdev -type d -name ".putdir" -print`
do
    find $i -mtime +1 -type f -ls -exec /bin/rm -f {} \;
done
if [ -d /var/preserve ]
then
    echo "Eliminar ficheros antiguos de /var/preserve"
    find /var/preserve -type f -size +1 -mtime +30 -ls -exec rm -f {} \;
fi
if [ -d /var/spool/mail ]
then CORREO=/var/spool/mail
elif [ -d /var/mail ]
then CORREO=/var/mail
else CORREO=""
fi
if [ ! -z ${CORREO} ]
then
    echo "Borrar correo antiguo de ${CORREO}"
    find ${CORREO} -type f \( -mtime +60 -o -nouser \) -ls -exec rm -f {} \;
fi

```



```
date=`date`  
echo "\n`basename $0` termina el $date en $uname.\n"
```

## C.12. comprobar.paquetes

Lista todos los paquetes RPM –situados en el directorio de trabajo actual– que contengan un cierto patrón en los caminos de los ficheros que lo forman.

```
#!/bin/bash  
# comprobar.paquetes - lista los paquetes que contienen un cierto patrón.  
  
for PAQUETE in *.rpm  
do  
    if [ -n "`rpm -qlp $PAQUETE|grep $1`" ]  
    then  
        echo "El fichero/patrón \"$1\" está en el paquete \"$PAQUETE\"."  
    fi  
done
```

## C.13. tararear

Utilidad interactiva para gestionar copias de seguridad de datos en cinta o en ficheros, ejecutando la orden **tar** en segundo plano. Se generan un fichero de salida normal y otro de errores.

```
# S. O.: AIX 3.x, 4.x  
# S. O.: Solaris 2.x  
# S. O.: Linux 2.x  
# tararear - Genera y extrae datos de ficheros "tar". Guiado por menús.  
#      Uso: tararear  
# Ramón Gómez - Octubre 1.996.  
# Versión 2.0: Junio 1.998. Compatibilidad con Solaris y uso de GNU Tar.  
# Versión 2.1: Marzo 1.999. Parámetros para usar varias cintas de longitud  
#      fija en Solaris (Rafa Sierra indica los valores adecuados).  
# Versión 3.0: Marzo 2.000. Compatibilidad con Linux.  
#      Añadir opción para comprobar el estado de una copia.  
#      Permitir acceso a servidor remoto.  
  
### Imprime en pantalla sin salto de línea.  
#      Parámetros:  
#      * - Cadena(s) a imprimir.  
echon ()  
{  
    if [ "x`echo -n`" = "x-n" ]  
    then echo "$*\c"  
    else echo -n "$*"  
    fi  
}
```

```

### Establece las variables adecuadas para el entorno.
entorno ()
{
    SO=`uname`
    case "$SO" in
        AIX)          DISP=/dev/rmt0
                      TARSAL=$HOMECD/doctype/tar.sal
                      TARERR=$HOMECD/doctype/tar.err
                      oAC="ç"; INTER="";
                      ;;
        SunOS)        DISP=/dev/rmt/0
                      TARSAL=$HOMECD/temp/tar.sal
                      TARERR=$HOMECD/temp/tar.err
                      oAC="ó"; INTER="¿"
                      ;;
        Linux)         DISP=/dev/st0
                      TARSAL=/tmp/tar.sal
                      TARERR=/tmp/tar.err
                      oAC="ó"; INTER="¿"
                      ;;
        *)            echo "`basename $0`: No disponible para el Sistema
Operativo: $SO." >&2
                      exit 1;;
    esac
}

### PROGRAMA PRINCIPAL.

entorno

TAR=`which gtar|grep "^/"`
TAR=${TAR:-`which tar|grep "^/"`}

echo "
Gesti${oAC}n de copias de seguridad v3.0.          Ramón Gómez - Marzo 2.000
"
ELEC=0
while [ "$ELEC" = 0 ]
do
    echo "1 - Crear una nueva copia de seguridad."
    echo "2 - Restaurar una cinta con copia de seguridad."
    echo "3 - Ver/comprobar el estado de la copia de seguridad."
    echo "0 - Salir del programa."
    echon "Elige una opci${oAC}n: "
    read ELEC
    case "$ELEC" in
        1) OPC=c
           MENS="Crear cinta";;
        2) OPC=x
           MENS="RESTAURAR cinta";;
        3) OPC=t
           MENS="Comprobar cinta";;
        0) exit;;
        *) print "Debes escoger una opci${oAC}n.\n"
           ELEC=0;;
    esac
done
echo
echon "Servidor con unidad de cintas [`hostname`]: "

```

```

read SERV
if [ ! -z "$SERV" ]
then
    SERV=${SERV}:
fi
echon "Dispositivo a utilizar [$DISP]: "
read FICH
DISP=${FICH:-$DISP}
if [ -z "$FICH" -a "`uname`" != "AIX" ]
then
    echo "0 - Varias cintas."
    echo "1 - Varias cintas de 90m."
    echo "2 - Varias cintas de 120m."
    echo "3 - Una sola cinta."
    echon "Elige una opci${oAC}n ..... [0]: "
    read ELEC
    case "$ELEC" in
        1) LONG="-ML 1900000" ;;
        2) LONG="-ML 3900000" ;;
        3) LONG="" ;;
        *) LONG="-M"
    esac
fi

echon "Opciones y directorios [.]:"
read DIRS
DIRS=${DIRS:-"."}

echon "${INTER}$MENS? (s/n) .. [n]: "
read RESP
if [ "$RESP" = "s" -o "$RESP" = "S" ]
then
    echo "Lanzado..."
    $STAR ${OPC}pvf ${SERV}${DISP} $LONG $DIRS 2>&1 >$TARSAL | tee $TARERR
fi

```