

INTRODUCCIÓN A LINUX

Ramón M^ª Gómez Labrador

Mayo 1.998

Temario del curso.

Introducción al UNIX.

Instalación de Linux.

Cómo empezar.

Ficheros y directorios.

Procesos.

El entorno gráfico Xwindow.

Gestión de usuarios y grupos.

Referencias.

Introducción al UNIX.

El sistema operativo UNIX empezó a desarrollarse por universitarios estadounidenses en la década de los 70, escrito en lenguaje C. Desde sus orígenes ha sido dotado de una gran versatilidad y facilidad de utilización para el usuario, permitiéndole investigar y ampliar cómodamente el propio sistema, en detrimento de cierta seguridad.

Existe una gran variedad de versiones de UNIX para diferentes plataformas –tanto para ordenadores personales y estaciones de trabajo, como para servidores y miniordenadores–, tanto de libre distribución (Linux, Minix) como comerciales (Solaris, AIX, HP-UX, ...). Aunque todas ellas mantienen una filosofía común, no son exactamente equivalentes. En la actualidad se está haciendo un gran esfuerzo de normalización en esta materia (normas UNIX'95 y UNIX'98 o los entornos Xwindow y CDE).

Linux es un UNIX de libre distribución –desarrollado inicialmente por el programador finlandés Linus Torvalds–, que ha crecido rápidamente gracias a la ayuda de una gran cantidad de programadores comunicados por Internet, y que han desarrollado y probado muchos componentes para el sistema.

En la actualidad existen varias distribuciones de Linux: RedHat (es la que utilizaremos en este curso), Slackware, Debian, etc. Algunas de ellas han sido desarrolladas no sólo para ordenadores personales –basados en procesadores Intel o compatibles–, sino también para máquinas con procesadores Sparc o PowerPC.

Antes de comenzar a centrarnos con el Linux, conviene conocer algunas de las características fundamentales del UNIX (y, por extensión, del propio Linux):

- Sistema operativo **multiusuario y multitarea**.
- Sistema operativo **basado en capas**.
- El almacenamiento en disco se organiza en **sistemas de archivos**.
- Los **espacios de paginación** permiten incrementar la memoria disponible.
- Trabaja con el protocolo de red **TCP/IP**.
- Cada usuario tiene una **cuenta** cuyas características y permisos vienen definidos por el administrador (**root**).
- El usuario se comunica con el sistema mediante el **intérprete de mandatos**.
- Cada programa en ejecución consta de uno o más **procesos**, con identificador único y con una relación de **parentesco**.

Instalación de Linux.

Preinstalación.

Antes de empezar con la instalación de Linux –o de cualquier otro sistema operativo– resulta necesario, y en algunos casos imprescindible, conocer las características fundamentales de nuestro ordenador. Éstas son:

- Procesador.
- Memoria.
- Discos duros y disqueteras.
- Tarjeta de vídeo.
- Monitor, teclado y ratón.
- Unidad de CD-ROM.
- Tarjeta de red (si existe).
- Otros dispositivos (impresoras, tarjeta de sonido, unidades de cinta, ...).

Para este curso utilizaremos la versión 5 del Linux de RedHat. El paquete viene con 2 disquetes de arranque y 2 CD-ROM.

Los disquetes pueden utilizarse para arrancar el ordenador con un núcleo reducido de Linux. Sin embargo, nuestra instalación se realizará desde una pequeña partición MS-DOS (aunque Linux también puede “convivir” sin problemas con Windows, OS/2 u otra versión de Linux). Es interesante hacer constar que Linux puede instalarse también desde un servidor FTP anónimo o desde un servidor NFS (siempre que se tenga acceso a Internet o a una red local, respectivamente).

En principio, se instalará MS-DOS. Si los dos sistemas operativos van a estar en el mismo disco duro, no es necesario preocuparse por el tamaño de las particiones, ya que Linux permite modificarlas en caliente sin reformatearlas. El paso siguiente es instalar los controladores de la unidad de CD-ROM y terminar de configurar el resto de dispositivos que vayan a utilizarse (tarjeta de red, de vídeo, impresora, etc.). Colocar el disco de Linux en la unidad de CD-ROM y rearrancar el ordenador.

Primeros pasos en la instalación.

Durante el proceso de instalación se tiene a disposición una serie de consolas virtuales que permitirán comprobar la evolución de este proceso. La siguiente tabla muestra el contenido de cada consola y la combinación de tecla para cambiar a ella. En general, sólo debe usarse la consola virtual 1.

Consola	Teclas	Contenidos
1	[Alt]-[F1]	Diálogos de la instalación
2	[Alt]-[F2]	Intérprete de mandatos
3	[Alt]-[F3]	Histórico de instalación (mensajes programa de instalación)
4	[Alt]-[F4]	Histórico del sistema (mensajes del núcleo, etc.)
5	[Alt]-[F5]	Otros mensajes

Para comenzar la instalación desde MS-DOS:

```
D:                (o la letra de la unidad de CD-ROM)

CD \DOSUTILS

AUTOBOOT
```

Si se usa el disquete de arranque de Linux, una vez que aparezca el mensaje `boot`, pulsar la tecla [Intro] y ejecutará automáticamente el proceso de instalación.

La siguiente tabla muestra los primeros pasos del programa de instalación, en la primera columna, y la elección que debemos efectuar, en la segunda. Conviene resaltar que los datos mostrados son para un tipo específico de instalación y que en otros casos deberán estudiarse los valores más necesarios (referirse al manual de Linux de RedHat ^[1] para más información.

Tipo de monitor:	Monitor color.
Tipo de teclado:	Teclado español (es).
Tipo de instalación:	Instalación con CD-ROM local.
Instalar o actualizar Linux:	Instalar un sistema nuevo.
Herramienta para crear particiones:	Disk Druid.

Creación de particiones.

El sistema operativo debe conocer los puntos de montaje para todos los sistemas de archivos definidos en nuestro árbol de directorios y los espacios de paginación que van a utilizarse. El Linux debe tener al menos un sistema de archivos (/) y un espacio de paginación.

Un **sistema de archivos** es un árbol de directorios que tiene un punto de montaje en el árbol de directorios global. Un servidor potente, con muchos usuarios y una gran carga de trabajo, debe tener definida por seguridad una serie de sistemas de archivos, donde cada uno de ellos tiene una función bien definida. La siguiente tabla muestra un ejemplo con los sistemas de archivos típicos

de una estación de trabajo.

/	Sistema de archivos raíz.
/usr	Contiene los programas y utilidades del sistema operativo.
/usr/local	Programas y utilidades adicionales.
/tmp	Almacena ficheros temporales.
/var	Ficheros de administración y contabilidad.
/home	Cuentas de los usuarios.

Las características fundamentales de un sistema de archivos son:

- Punto de montaje: directorio del árbol raíz en el que incluimos el sistema de archivos.
- Dispositivo: partición del disco utilizada.
- Tipo: tipo de la partición usada (Linux, DOS, paginación, CD-ROM, ...).
- Tamaño: espacio reservado en el disco.

Resulta de gran importancia hacer una buena estimación previa del tamaño que va a tener cada uno de los sistemas de archivos definidos en la máquina.

En nuestro caso, para simplificar, sólo se definirá el sistema de archivos raíz y se mantendrá el de MS-DOS montado sobre el directorio /DOS.

La memoria del ordenador es limitada y Linux permite la ejecución de varios procesos a la vez que comparten dicha memoria. Cuando ésta se satura, el sistema operativo utiliza el espacio en disco para intercambiar información. Esto es lo que se define como **espacio de paginación** o **espacio de intercambio**.

Normalmente, el espacio del disco que se reserva para paginación dependerá de la carga que estimemos vaya a tener el ordenador. El valor recomendado por Linux es el doble de la capacidad de nuestra memoria. Otros sistemas operativos (como el AIX de IBM) recomiendan reservar el triple de la cantidad de la memoria. En todo caso, Linux permite aumentar temporalmente nuestro espacio de paginación.

Se definirá un espacio de paginación con el doble de la cantidad de memoria del ordenador. El propio Linux de RedHat monta dicho espacio sobre el directorio /proc.

Paquetes de programas.

La instalación del Linux de RedHat permite seleccionar los paquetes de programas que más interesen para la funciones que llevará a cabo el servidor y según el espacio libre en disco del que dispongamos. Dichos paquetes están agrupados en componentes de acuerdo con su función.

El programa de instalación permite seleccionar los paquetes individuales o los componentes que más nos interesen. Hay que tener en cuenta que algunos paquetes dependen de la existencia de otros y el propio programa de instalación se encarga de comprobar dichas dependencias.

Las etiquetas de cada componente y de cada paquete dan una idea general de la función que realizan. Es conveniente tener conocimiento de la funcionalidad de cada paquete antes de añadirlo a la lista, pero en cualquier momento –tras el proceso de instalación– pueden incluirse o eliminarse con la utilidad `rpm`.

Finalizar la instalación.

Para terminar con el proceso de instalación, la siguiente tabla muestra los últimos pasos y los valores recomendados para el curso.

Configurar el ratón:	Seleccionar ratón por omisión.
Configurar Xwindow:	Seleccionar servidor XFree86 escogido por la utilidad <code>Xconfigurator</code> .
Configurar la red:	Ver el próximo apartado.
Configurar zona horaria:	Seleccionar Europa/Madrid.
Seleccionar servicios para el arranque:	Escoger aquellos servicios que más interesen para la configuración inicial del ordenador.
Configurar la impresora:	Selecciona modelo de la impresora (si hay).
Clave para usuario root :	Teclear la clave para el administrador.
LILO:	Instalar LILO en el sector de arranque para seleccionar arranque Linux o MS-DOS.

Es importante conocer la función de los servicios que se han seleccionado para el arranque del ordenador. Cada servicio supone uno o varios procesos ejecutándose en segundo plano. Si se añaden más servicios de los necesarios podemos disminuir el rendimiento del ordenador. En cualquier caso, tras el proceso de instalación puede modificarse dicha lista de servicios.

La siguiente tabla muestra las utilidades que ofrece el programa `/usr/sbin/setup` de Linux para la configuración del sistema.

Utilidad	Comentarios
<code>/usr/sbin/cabaret</code>	Configuración de sistemas de archivos.
<code>/usr/sbin/kbdconfig</code>	Configuración del teclado.
<code>/usr/sbin/mouseconfig</code>	Configuración del ratón.
<code>/usr/sbin/ntsysv</code>	Modifica la lista de servicios a cargar en el arranque.
<code>/usr/sbin/sndconfig</code>	Configuración de la tarjeta de sonido.
<code>/usr/sbin/timeconfig</code>	Configuración de la zona horaria.
<code>Xconfigurator</code>	Configuración de Xwindow.

Configuración de la red.

El primer paso para configurar la red es seleccionar el modelo de la tarjeta de comunicaciones del ordenador. El programa probará el sistema para intentar localizarla.

El resto de la configuración incluye los siguientes parámetros:

- Dirección IP del ordenador.
- Máscara de red.
- Dirección IP del encaminador por omisión.
- Dirección IP del servidor de nombre primario.
- Nombre del dominio.
- Nombre del ordenador.
- Direcciones IP de los servidores de nombre secundario y terciario (opcionales).

Cómo empezar.

La primera sesión en Linux.

Desde la consola del propio ordenador –o entrando vía Telnet–, el usuario encontrará un aviso pidiéndole el nombre de conexión y la clave. Tras la instalación del sistema operativo, sólo está disponible el usuario administrador: **root**.

```
nombre_pc login: root
Passwd: introducir la clave
```

Es importante que el administrador conozca a fondo la estructura de los sistemas de archivos y la distribución de los ficheros en el árbol. La siguiente tabla muestra los directorios de mayor importancia en Linux.

Directorio	Comentario
/etc	Ficheros de configuración de la máquina.
/etc/rc.d	Ficheros de configuración del arranque.
/etc/skel	Plantillas para cuentas de usuarios.
/sbin	Ficheros esenciales del arranque, montaje del /usr y recuperación del sistema.

/bin	Ejecutables del sistema
/lib	Bibliotecas de los programas de /sbin y /bin.
/usr	Ficheros compartibles por todo el sistema. Tiene un árbol de subdirectorios con una estructura similar al raíz. Suele montarse sólo para lectura.
/usr/local	Programas instalados independientes del sistema operativo.
/var	Ficheros varios (administrativos, históricos, bloqueos, ...).
/tmp	Ficheros temporales.
/home	Datos de las cuentas de los usuarios.

Casi la totalidad de los directorios está gestionada por **root**, por lo que deben estar bien protegidos contra intrusiones de otros usuarios.

Una vez asimiladas las nociones previas, el usuario puede empezar a conocer algunos mandatos de interés para su primera sesión en Linux. La orden **ls** –muy conocido en todos los dialectos de UNIX– permite listar el contenido de los directorios. Veremos unos ejemplos:

- Mostrar el contenido del directorio raíz. Nótese que el símbolo \$ es el punto indicativo del intérprete de mandatos.

```
$ ls /
DOS  dev  lib      mnt  sbin  var
bin  etc  lost+found  proc  tmp
boot home  misc      root  usr
```

- Ver el contenido del directorio /usr incluyendo los archivos ocultos. Un archivo se considera oculto cuando su nombre comienza por punto. El directorio actual se denota por el fichero . (punto) y el directorio padre por .. (punto punto).

```
$ ls -a /usr
.      dict  include  local  src
..     doc   info     man    tmp
X11    etc   lib      mnt    var
bin    games libexec  sbin
```

- Examinar el contenido completo del directorio actual. La salida se divide en varias columnas, mostrando: permisos, número de enlaces, propietario, grupo, tamaño, fecha y nombre.

```
$ ls -al
drwx----- 1 root  root   1024 May 25 09:10 .
drwxr-xr-x 19 root  root   1024 May 03 17:27 ..
-rw----- 1 root  root   1126 Aug 23  1995 .Xdefaults
-rw-r--r-- 1 root  root    174 May 25 09:18 .bashrc
```

```
-rw-r--r-- 1 root root 25580 May 22 13:01 prueba.txt
```

Pero, ¿cuál es el directorio actual? En muchos casos, el usuario suele configurarse su perfil de entrada (fichero `.profile`) para que aparezca el directorio actual en el punto indicativo. En cualquier caso, podemos utilizar el mandato **pwd**.

```
$ pwd
/root
```

La orden **man** permite leer las páginas de manuales del sistema operativo y obtener una ayuda completa de cualquier mandato Linux, de ficheros de configuración, de llamadas al sistema para programación, etc.

```
$ man ls
Ayuda completa sobre el mandato ls.
```

La mayoría de las órdenes en Linux incluyen la opción `--help` para obtener una ayuda abreviada.

Revisando las páginas de manual de varios mandatos, el usuario podrá advertir que los mandatos UNIX suelen tener un formato bien definido.

```
mandato opciones argumentos
```

- Las opciones de un mandato permiten modificar o calificar el comportamiento del mandato. Suelen ir precedidas por uno o dos signos menos (`-` o `--`).
- Los argumentos son aquellos ficheros, directorios o elementos a los que se le aplicará dicho mandato.

ls	-al	/usr
Mandato;	Opciones:	Argumentos:
<ul style="list-style-type: none">• Listar directorios	<ul style="list-style-type: none">• <code>-a</code>: listar ocultos• <code>-l</code>: lista ampliada	<ul style="list-style-type: none">• <code>/usr</code>: contenido del directorio

Aunque la mayoría de los mandatos de los distintos dialectos de UNIX siguen este formato, existen algunas excepciones. Por otro lado, el UNIX es un sistema que distingue entre mayúsculas y minúsculas, por lo que no es lo mismo escribir `ls` que `Ls` o `LS`.

Para finalizar esta primera sesión Linux, falta conocer el mandato para la desconexión del sistema. Normalmente, basta con pulsar desde el punto indicativo las teclas **[Ctrl]-[D]**. Esta combinación de teclas ejecuta el mandato **logout**.

Los mandatos más comunes en Linux.

Este apartado pretende ser una pequeña referencia para algunos de los mandatos de uso frecuente en Linux ordenados alfabéticamente.

bash	Uno de los intérpretes de mandatos más utilizados en Linux.
-------------	-------------------------------------------------------------

cat	Imprime el contenido de un fichero.
cd	Cambia de directorio.
chmod	Cambia los permisos de ficheros y directorios.
cp	Copia ficheros y directorios.
cut	Filtro para seleccionar columnas en ficheros de texto.
date	Muestra o cambia la fecha y la hora.
diff	Muestra las diferencias entre ficheros.
find	Búsqueda de ficheros y directorios que cumplan ciertas condiciones.
grep	Filtro para buscar expresiones regulares en ficheros de texto.
git	Entorno para realizar cómodamente las operaciones más rutinarias sobre ficheros (propio de Linux).
gzip	Utilidad para la compresión y descompresión de datos.
joe	Editor más cómodo, muy usado en Linux.
kill	Elimina o manda señales a procesos.
ls	Lista el contenido de directorios.
man	Muestra páginas de manual.
mkdir	Crea un nuevo directorio.
more	Imprime ficheros pantalla a pantalla.
mv	Mueve o renombra ficheros y directorios.
pwd	Imprime el camino del directorio actual.
passwd	Cambia la clave del usuario.
ps	Muestra información sobre procesos.
rm	Borra ficheros.
rmdir	Borra un directorio vacío.
sort	Ordenación de ficheros.
top	Entorno que muestra los procesos con mayor carga sobre la máquina (Linux).
vi	Editor más normal de UNIX.
wc	Cuenta caracteres, palabras y líneas de ficheros.
who	Lista los usuarios conectados.

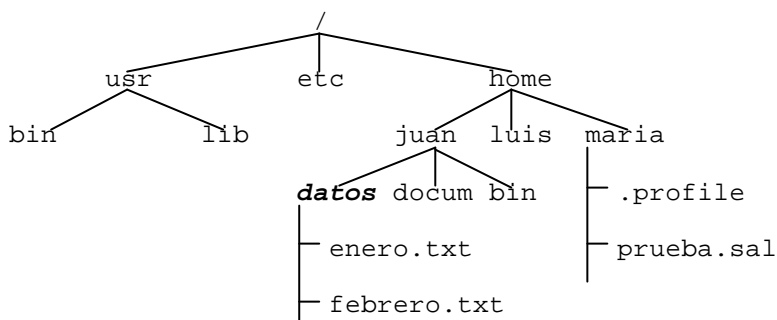
xinit

Inicia el entorno Xwindow.

Ficheros y directorios.

Camino absoluto y camino relativo.

Como se ha comentado previamente, los ficheros y directorios se agrupan en forma de árbol, cuyo directorio raíz se denota por /. El mismo carácter barra inclinada se utiliza como separador de subdirectorios. Obsérvese el siguiente ejemplo.



Suponiendo que el directorio actual es el directorio `datos` de `jaun`:

```
$ pwd
/home/juan/datos
$ ls
enero.txt febrero.txt
```

Para hacer referencia a un fichero en el directorio actual, basta con dar su nombre. Sin embargo, el camino completo de dicho fichero se obtiene precediendo al nombre del fichero el camino completo del directorio que lo contiene.

En el ejemplo, el camino completo del fichero `febrero.txt` es:

```
/home/juan/datos/febrero.txt
```

De igual modo, el camino completo del fichero `prueba.sal` del directorio `maria` es:

```
/home/maria/prueba.sal
```

Pero también podemos referirnos al mismo fichero de forma relativa respecto al directorio actual. Teniendo en cuenta que el directorio padre de uno dado se denota por `..` (punto punto), el camino relativo sería:

```
../../maria/prueba.sal
```

Conviene resaltar que el camino completo comienza siempre por la barra inclinada, mientras que el camino relativo comienza por cualquier otro carácter.

El siguiente ejemplo ilustra algunos usos comunes del trabajo con caminos relativos y caminos completos.

```
$ cat /home/maria/prueba.sal
Esto es una prueba de texto.

$ cat ../../maria/prueba.sal
Esto es una prueba de texto.

$ cd ../docum

$ pwd

/home/juan/docum

$ cat ../datos/enero.txt
Este fichero contiene los datos del mes de Enero.
```

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.

Cada usuario tiene un nombre de conexión único en el ordenador y pertenecerá a uno o varios grupos de usuarios. Teniendo esto en cuenta, un fichero o directorio tiene 3 conjuntos de permisos: permisos para el propietario, para el grupo y para el resto de usuarios.

El propietario de dicho fichero puede seleccionar qué permisos desea activar y cuales deshabilitar.

Para comprobarlo de manera más clara, tómesese 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), ...
- 3 caracteres para los permisos del propietario.
- 3 caracteres para los permisos para otros usuarios del grupo.
- 3 caracteres para los permisos para el resto de usuario.

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

- Ficheros:
 - Lectura (r): el usuario puede leer el fichero.
 - Escritura (w): el usuario puede escribir en el fichero.
 - Ejecución (x): el usuario puede ejecutar el fichero (siempre que sea un ejecutable o un guión de intérprete de mandatos).

- Identificador de usuario activo (S para el propietario): un programa ejecutable puede activar el identificador de usuario (SUID), esto permite que durante la ejecución del programa un usuario se convierte en el usuario propietario del fichero. Por ejemplo, el mandato **passwd** accede a ficheros que sólo puede modificar el usuario **root**. Dicho mandato tiene activo el SUID para que durante la ejecución del programa otro usuario sea por algún momento **root** y pueda cambiar su clave. Hay que tener especial cuidado con estos ejecutables, porque usuarios no autorizados pueden tomar privilegios.
- Identificador de grupo activo (S para el grupo): al igual que en el caso anterior, un programa ejecutable puede activar el identificador de grupo (SGID) para que un usuario pueda realizar operaciones propias del grupo al que pertenece el fichero. Por ejemplo, el mandato **mail** activa el SGID para que cualquier usuario pueda acceder a su buzón de correo sin posibilidad de leer correo de cualquier otro usuario.
- $s = S + x$.
- Directorio (d):
 - Lectura (r): el usuario puede leer el contenido del directorio.
 - Escritura (w): el usuario puede crear, modificar y borrar entradas del directorio.
 - Acceso (x): el usuario puede acceder al directorio, permitiendo colocarse en él como directorio actual (utilizando el mandato **cd**). Este tipo de permiso permite proteger cierta información de un directorio padre y, sin embargo, acceder a la información de los directorios hijos.
 - Directorio de intercambio (T en el resto de usuarios): permite que en directorios compartidos los ficheros sólo puedan ser modificados por el propietario.
 - Identificador de grupo activo (S 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.
 - $t = T + x$.
 - $s = S + x$.

La siguiente tabla muestra los permisos necesarios para poder ejecutar algunos mandatos:

Mandato	Permiso directorio origen	Permisos fichero	Permisos directorio destino
cd	X	No aplicable	No aplicable
ls	R	No aplicable	No aplicable
mkdir	W, X	No aplicable	No aplicable
rmdir	W, X	No aplicable	No aplicable
cat	X	R	No aplicable
rm	W, X	-	No aplicable
cp	X	R	W, X

mv	W, X	-	W, X
-----------	------	---	------

Ampliando el ejemplo del apartado anterior:

- El usuario `juan` pertenece al grupo `ventas`.
- El usuario `luis` está incluido en los grupos `admin` y `general`.
- La usuaria `maria` pertenece a los grupos `ventas` y `general`.

```

luis$ ls -al /home/juan/datos
drwxrwxr-x 1 juan ventas _____ .
drwx--x--x 4 juan ventas _____ ..
-rw-rw-r-- 1 maria ventas _____ enero.txt
-rw----- 1 juan ventas _____ febrero.txt

```

Con esta información, ¿cuáles de los siguientes mandatos se realizarán con éxito y cuáles crearán algún error por violación de permisos?

<code>luis\$ cd /home/juan</code>	Error: falta permiso de acceso al directorio.
<code>luis\$ cd /home/juan/datos</code>	Correcto.
<code>luis\$ cp enero.txt /home/luis</code>	Correcto (si tiene permiso de escritura en su directorio).
<code>luis\$ cat febrero.txt</code>	Error: falta permiso de lectura del fichero.
<code>luis\$ vi enero.txt</code>	Falta permiso para escribir en el fichero.

Los permisos también se representan de forma numérica con valores en octal. La siguiente tabla muestra dicha equivalencia.

S (propietario)	4000
S (grupo)	2000
T (otros)	1000
R (propietario)	400
W (propietario)	200
X (propietario)	100
R (grupo)	40
W (grupo)	20
X (grupo)	10
R (otros)	4

W (otros)	2
X (otros)	1

La orden **chmod** es la utilizada para cambiar los permisos de ficheros y directorios, permitiendo tanto la descripción numérica como la simbólica.

```

juan$ ls -l datos/febrero.txt
-rw----- 1 juan  ventas  _____  febrero.txt
juan$ chmod g+rw febrero.txt
juan$ ls -l datos/febrero.txt
-rw-rw---- 1 juan  ventas  _____  febrero.txt
juan$ chmod 644 febrero.txt
juan$ ls -l datos/febrero.txt
-rw-r--r-- 1 juan  ventas  _____  febrero.txt

```

Procesos.

Conceptos básicos.

Puede entenderse por **proceso** todo programa o mandato en ejecución. Un proceso tiene las siguientes características:

- Consta de zona de código, de datos y de pila.
- Los procesos existen en una jerarquía de árbol (varios hijos, un sólo padre).
- El sistema asigna un identificador de proceso (**PID**) único al iniciar el proceso.
- El planificador de tareas asigna un tiempo compartido para el proceso según su prioridad (sólo **root** puede aumentar la prioridad de un proceso).
- Cada proceso almacena su identificador (**PID**) el de su proceso padre (**PPID**), el propietario y grupo del proceso y las variables de entorno.

Tipos de procesos:

- **Ejecución en 1^{er} plano:** proceso iniciado por el usuario o interactivo.
- **Ejecución en 2^o plano:** proceso no interactivo que no necesita ser iniciado por el usuario. Tiene una prioridad menor que un proceso interactivo.

Procesos especiales:

- **Proceso servidor o demonio:** proceso en 2º plano siempre disponible y que da servicio a varias tareas (debe ser propiedad del usuario **root**).
- **Proceso zombi:** proceso parado que queda en la tabla de procesos hasta que termine la ejecución de su padre. Este hecho se produce cuando el proceso padre no recoge el código de salida del proceso hijo.
- **Proceso huérfano:** proceso en ejecución cuyo padre ha finalizado. El nuevo identificador de proceso padre (PPID) coincide con el identificador del proceso **init** (1).

Manipulación de procesos.

El mandato **ps** permite ver los procesos que están activos en la máquina, pudiendo mostrar el identificador del proceso y del padre, el estado del proceso, su prioridad, desde qué terminal se ejecutó, el nombre del propietario o el mandato que se ha ejecutado.

Para ver los procesos que tenemos en ejecución actualmente:

```
$ ps
  PID TTY STAT TIME COMMAND
 1273 1   S    0:00 /bin/login -- ps
 2721 1   S    0:00 -bash
 2738 1   R    0:00 ps
```

Para enviar un proceso al 2º plano, basta con colocar un ampersand (&) tras el mandato:

```
$ sleep 100 &
[1] 3064
$ ps fu
USER      PID _____ COMMAND
root     1273 _____ /bin/login -- ps
root     2721 _____ \_ -bash
root     3064 _____ \_ sleep 100
root     3080 _____ \_ ps fu
```

Como muestra la pantalla anterior, el mandato **sleep** ha sido enviado a 2º plano y no hace falta esperar para continuar trabajando. Por otro lado, puede observarse que el intérprete de mandatos (**bash**) es el padre del mandato de espera (**sleep**) y del propio mandato para informe de procesos (**ps**).

Linux incluye la herramienta **top**, que permite una gestión cómoda de los procesos en ejecución. Por otro lado, conviene resaltar que suele ser de los procesos que mantienen una ocupación mayor del sistema y puede ralentizarlo.

Las órdenes internas **fg** y **bg** se utilizan para enviar un proceso al 1º o al 2º plano, respectivamente.

Una **señal** es un evento que puede ser procesado por un proceso y que puede afectar a su ejecución normal.

La orden **kill** se utiliza para enviar señales a procesos. Una de las señales de mayor importancia es la de matar un proceso, o sea, hacer que finalice en el acto.

En el caso anterior, para terminar el proceso de espera:

```
$ kill 3064
[1]+ Terminated sleep 100
```

Hay situaciones que no es suficiente con la señal de parada y debe usarse la señal de matar:

```
$ kill -9 3064
```

Redirecciones y tuberías.

Para cada proceso en ejecución el sistema abre 3 ficheros especiales:

0. **Entrada normal** (stdin): datos de entrada por teclado.
1. **Salida normal** (stdout): datos de salida a la pantalla.
2. **Salida de error** (stderr): datos de errores a la pantalla.

Estos valores por omisión pueden modificarse utilizando las redirecciones. Es importante no confundir la redirección con los parámetros de un mandato.

Los operadores de redirección son:

<	Redirigir la entrada desde un fichero.
>	Redirigir la salida a un fichero.
>>	Añadir la salida a un fichero.
2>	Redirigir la salida de error a un fichero.
2>>	Añadir la salida de error a un fichero.

El fichero **/dev/null** se utiliza como papelera para hacer redirecciones nulas.

Un **filtro** es un programa que lee datos de la entrada normal, los procesa y escribe los resultados en la salida normal. Filtros típicos de UNIX son los mandatos **grep**, **sort**, **cut**, **sed**, ...

Una **asociación** es enviar una redirección al mismo fichero que se utiliza para otra redirección. Es muy importante saber que las sustituciones se realizan de izquierda a derecha y se representan de la siguiente manera:

0	Sustituye la entrada normal.
&1	Sustituye la salida normal.
&2	Sustituye la salida de error.

Veremos algunos ejemplos que ilustran los conceptos anteriores.

- Ordenar la entrada del fichero fich1.ent, añadir la salida a fich2.sal e ignorar los errores:

```
$ sort <fich1.ent >>fich2.sal 2>/dev/null
```

- Copiar datos.sal y los posibles errores de esta operación en datos.copia.

```
$ cat datos.sal >datos.copia 2>&1
```

- Listar todo el árbol de directorios, mandar los errores a la salida normal (pantalla) y redirigir la salida normal a todoelarbol.sal:

```
$ ls -alR / 2>&1 >todoelarbol.sal
```

- Buscar todos los directorios, mandar la salida a direc.sal, los errores a direc.err y hacer la ejecución del mandato en 2º plano:

```
$ find / -type d -ls >direc.sal 2>direc.err &
```

Una **tubería** es un mecanismo entre 2 procesos por el cual la salida normal de uno de ellos se sincroniza con la entrada normal del segundo. Para representar una tubería los procesos se separan por el carácter barra vertical (|).

Los filtros suelen utilizarse para procesar datos recogidos por otras órdenes utilizando este mecanismo.

Más ejemplos:

- Mostrar el número de entradas no ocultas del directorio actual:

```
$ ls | wc -w
```

- Contar el número de intérpretes **bash** que hay en ejecución en el ordenador:

```
$ ps x | grep bash | wc -l
```

- Mandar el contenido del directorio a ls.sal y mostrarlo en pantalla con orden inverso (la orden **tee** copia la salida a pantalla y a un fichero):

```
$ ls | tee ls.sal | sort -r
```

El entorno gráfico Xwindow.

El entorno Xwindow fue desarrollado en principio por el Instituto Tecnológico de Massachusetts (MIT) a partir del W de la Universidad de Stanford. Se ha convertido en una norma no sólo para sistemas UNIX, ya que su código fuente es público. El Linux de RedHat 5.0 incluye la versión X11R6 basada en la instrumentación Xfree86 –también soportada por otros UNIX para ordenadores personales. En la actualidad el **Open Group** distribuye el entorno Xwindow, el manejador de ventanas MOTIF, el pupitre CDE y otros programas y entornos que son

independientes del sistema operativo y del tipo de ordenador que utilizemos.

Durante el proceso de instalación se ha realizado una configuración preliminar del Xwindow. Si quiere modificarse algún parámetro o si existe algún problema con la tarjeta de vídeo hay que volver a ejecutar como **root** la orden **Xconfigurator**.

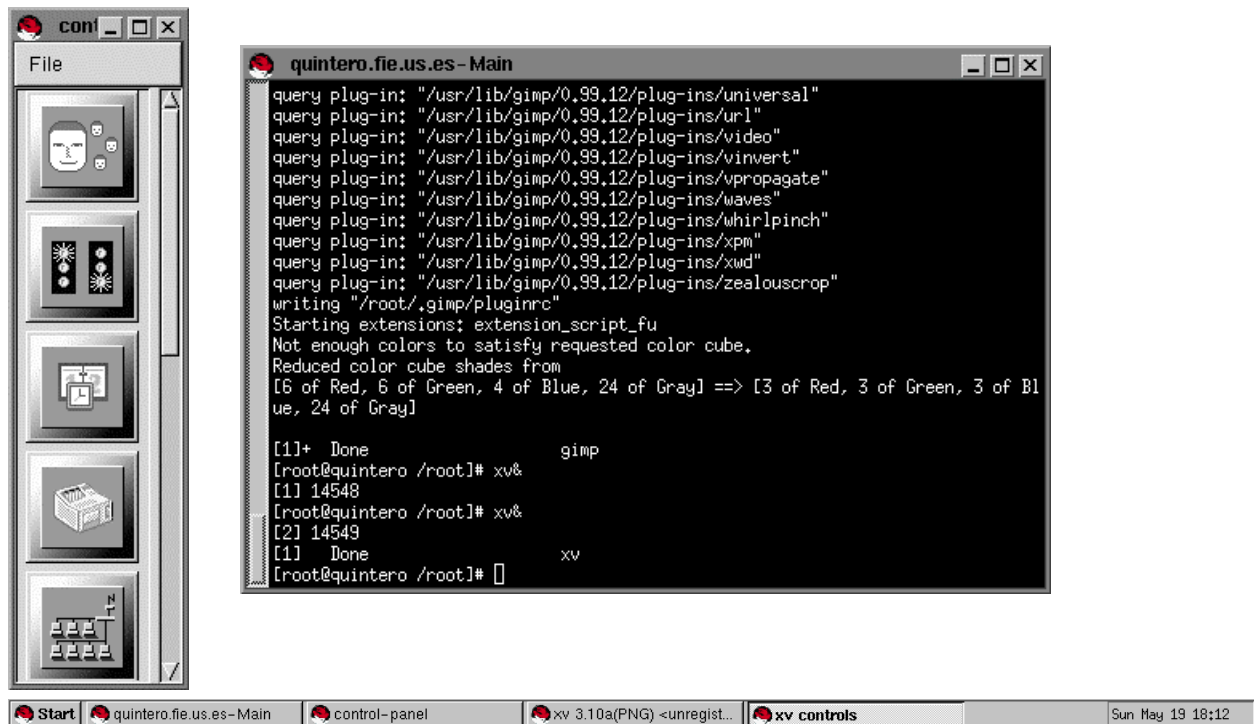
Para iniciar una sesión X, ejecutar **xinit** desde el punto indicativo. Esta orden también permite iniciar el manipulador de pantalla y los clientes (programas) que se ejecutarán en el arranque del entorno. Dicha configuración se lee del fichero **.xinitrc** en el directorio del usuario.

La configuración por omisión del X11R6 en Linux incluye el manejador de pantallas FVWM2 que incorpora las siguientes características:

- Entorno similar al del Windows 95.
- Permite usar un pupitre virtual más amplio o una serie de pupitres separados.
- Aceleradores de teclado para ejecutar la mayoría de sus funciones.
- Configuración de fondos, tipos, menús, combinaciones de teclas, etc.
- Basado en el manipulador de pantallas TWM.

El fichero de configuración carga la barra de tareas, el panel de control y un terminal para poder utilizar un intérprete de mandatos.

La mayoría de las aplicaciones y de las funciones propias de Xwindow pueden seleccionarse a partir de la barra de tareas o ejecutarse desde el intérprete de mandatos.



Fijándonos en los últimos mandatos ejecutados en la ventana de terminal, podemos ver que se han ejecutado –en 2º plano, como toda aplicación X– los programas **gimp** y **xv**.

El panel de control permite arrancar ciertas aplicaciones de configuración, como la gestión de usuarios, de la impresora, la fecha y la hora, los paquetes de programas, la red, etc.

Xwindow permite ver y operar en modo gráfico con programas que están ejecutándose en otro ordenador. Deben seguirse los siguientes pasos:

- En el ordenador local, indicar qué máquinas podrán enviar información X. Para ello se utiliza la orden **xhost**.
- Conectarse al ordenador remoto vía Telnet.
- En el ordenador remoto, exportar la variable **DISPLAY** para indicar el nombre o la dirección de nuestro ordenador local.
- Ejecutar la aplicación.

```

ramon:milinux$ xhost +
access control disabled, clients can connect from any host
ramon:milinux$ telnet miaix.us.es

_____

ramon:miaix> export DISPLAY=milinux.us.es:0.0
ramon:miaix> dtstyle &
    
```



Para finalizar una sesión X, debe utilizarse la opciones de salir del FVWM en el menú inicio (**Start**) en la barra de tareas. Si existe algún problema, Xwindow prevé la posibilidad de hacer una salida abrupta pulsando la combinación de teclas **[Ctrl][Alt][Borrar]**. Esta pulsación de teclas sólo debe usarse en casos estrictamente necesarios.

También puede cambiarse rápidamente la resolución actual de la pantalla:

- **[Ctrl][Alt][+ numérico]**: incrementa la resolución.
- **[Ctrl][Alt][- numérico]**: decrementa la resolución.

Para finalizar este tema, la siguiente table muestra algunos de los clientes X más utiliados:

gimp	Programa para la manipulación de imágenes.
gvim	Editor vi para X.
netscape	Navegador para hipertexto.
xfm	Gestor de sistemas de archivos.
xftp	Ciente FTP.
xman	Lector de páginas de manual.
xterm	Terminal X.

Gestión de usuarios y grupos.

El usuario administrador está encargado de la apertura, modificación y cancelación de las cuentas de los usuarios y grupos de usuarios definidos en el sistema.

Una **cuenta** permite la conexión de un usuario al ordenador y habilita un espacio en disco para que dicho usuario almacene sus datos. En Linux, la gestión de la cuenta conlleva el control de los siguientes parámetros:

- Contraseña de entrada.
- Grupos a los que pertenecerá el usuario.
- Variables de entorno.
- Conexiones permitidas.
- Cuotas de espacio en disco (opcional).
- Auditoría de seguridad (opcional).

Un **grupo de usuarios** es un conjunto que engloba a usuarios con una función común. Un usuario puede pertenecer a más de un grupo, pero debe tener definido un **grupo primario o inicial**.

El mandato **useradd** es el encargado de crear una nueva cuenta. La siguiente tabla muestra las opciones y los parámetros más necesarios en la llamada a **esta orden**:

Opciones	Descripción
<code>-u <i>identif</i></code>	Identificador de usuario.
<code>-g <i>grupo</i></code>	Grupo primario o inicial (previamente definido).
<code>-G <i>grupos</i></code>	Otros grupos previamente definidos (opcional).
<code>-d <i>directorio</i></code>	Directorio del usuario.
<code>-m</code>	Crea el directorio del usuario.
<code>-c <i>nombre</i></code>	Nombre completo del usuario.
<code>-k <i>directorio</i></code>	Directorio con ficheros de configuración inicial (usar <code>/etc/skel</code>).
<code>-s <i>intérprete</i></code>	Intérprete de mandatos inicial.
<code>-e <i>fecha</i></code>	Fecha de caducidad de la cuenta (opcional).

<i>usuario</i>	Nombre de conexión para el usuario.
----------------	-------------------------------------

El **nombre de conexión** para el usuario debe ser una única palabra de hasta 8 caracteres, formada por letras o números y que debe comenzar por una letra.

El mandato **useradd** utiliza los ficheros `/etc/default/useradd` y `/etc/login.defs` para obtener los parámetros por omisión en la creación de nuevas cuentas. En el directorio `/etc/skel` se almacenan los perfiles de entrada y los ficheros de configuración inicial para cada usuario.

```
# useradd -u 527 -g proyecto -G ventas,admin -s /bin/bash \
-md /home/ramon -k /etc/skel -c "Ramón Gómez" -e 12/31/98 ramon
```

Los mandatos **usermod** y **userdel** se utilizan, respectivamente, para modificar los parámetros de una cuenta y para eliminar una cuenta existente.

```
# userdel -r ramon
```

El administrador podrá utilizar también los mandatos **groupadd**, **groupmod** y **groupdel** permite crear, modificar y eliminar grupos de usuarios.

```
# groupadd -g 502 ventas
```

La base de datos de usuario se almacena en los ficheros `/etc/passwd` y `/etc/shadow` y la de grupos, en `/etc/group`. Las claves encriptadas se guardan en `/etc/passwd`, pero dicho fichero puede ser leído por cualquier usuario, por ello es muy importante utilizar el mandato **pwconv** después de **useradd** para actualizar las claves en `/etc/shadow`, ya que este fichero sí está protegido.

El mandato **passwd**, permite generar o cambiar la clave de acceso para el usuario. Una vez conectado el usuario nuevo, puede utilizar esta orden para cambiarse su clave.

La siguiente tabla muestra los mandatos que puede utilizar un usuario para ver el estado de su cuenta, ordenados alfabéticamente.

chfn	Cambia el nombre completo (si el administrador lo permite).
chgrp	Cambia el grupo de un fichero (siempre que el usuario pertenezca a dicho grupo).
chsh	Cambia el intérprete de mandatos inicial.
groups	Muestra los grupos en los que se encuentra el usuario.
id	Muestra identificador de usuario e identificadores de grupos.
passwd	Cambia la contraseña de entrada.
who am i	Indica el nombre del usuario y desde donde se ha conectado.
whoami	Imprime el nombre del usuario.

Referencias.

1. “RedHat Linux 5.0: Installation Manual”.
2. “Linux Instalación y Primeros Pasos” del Proyecto LuCAS.
3. “Guía del Usuario de Linux” del Proyecto LuCAS.
4. RedHat: <http://www.redhat.com>
5. Slackware: <http://www.cdrom.com/titles/os/slackwar.htm>
6. Linux International: <http://www.li.org>
7. Proyecto LuCAS (Linux en Castellano): <http://lucas.hispalinux.es/>
8. Spanish Linux User Group: <http://slug.ctv.es>
9. “Programación de Sistemas Unix: Comunicación entre Procesos” por Ramón M^a Gómez Labrador: <http://www.informatica.us.es/~ramon/articulos/seminario-1.html>
10. Open Group: <http://www.opengroup.org>