TEMA 9: GESTIÓN DE ARCHIVOS Y DIRECTORIOS EN LINUX

Índice

1. Introducción

Después de estudiar en temas anteriores la administración y gestión del sistema operativo Windows en sus versiones de escritorio y servidor, en esta evaluación vamos a tratar de ver algunos de esos mismos contenidos pero aplicados al sistema operativo libre por excelencia, Linux.

Linux, al igual que Windows, dispone de una interfaz gráfica (la cual es distinta dependiendo de la distribución con la que estemos trabajando) y una interfaz tipo texto desde la que podremos hacer absolutamente todas las tareas de administración y gestión del sistema. En entornos Linux se suele hacer generalmente toda la administración desde la línea de comandos, así, nosotros vamos a estudiar ambas interfaces, pero haciendo mayor hincapié en la interfaz tipo texto.

La distribución Linux con la que vamos a trabajar es la Ubuntu, en su versión 16.04 LTS[1]. Las distintas distribuciones se diferencian unas de otras en los programas que llevan incorporados, tanto los programas propios del sistema operativo como los añadidos (es normal que al instalar Linux se nos instalen también aplicaciones de ofimática, como el LibreOffice, clientes de servicios de Internet, aplicaciones de imagen y sonido, etc.).

Algunos ejemplos de programas propios del sistema operativo que son diferentes entre distribuciones pueden ser el gestor de arranque (GRUB, LILO), el entorno de escritorio gráfico (KDE, Gnome, Unity, XFCE…), exploradores de archivos (Konqueror, Nautilus, Thunar…), pantallas y gestores de acceso al sistema (GDM, KDM, …), y así con casi todos los elementos (excepto el kernel) que forman el sistema.

Veremos que la ventaja de trabajar desde la línea de comandos es que todos los comandos son prácticamente similares entre las distintas distribuciones.

[1] En el momento de hacer estos apuntes la 16.04 es la última distribución LTS de Ubuntu. Está a punto de salir (en abril de 2018) la siguiente LTS.

2. Sistema de archivos de Linux

Ya hemos conocido los distintos sistemas de ficheros de disco que existen para los sistemas operativos. En Windows trabajamos con NTFS, ReFS, o FAT32. En el caso de Linux, el sistema de ficheros que más se utiliza es el denominado **EXT4**. EXT4 es una evolución del original EXT2 y EXT3. Existen otros sistemas de ficheros de Linux. Estudiaremos con más detalle el EXT4 y otros en temas posteriores.

Nomenclatura de ficheros y directorios

En Linux, las normas para la asignación de nombres son las mismas para ficheros y directorios. Los nombres de archivos y directorios pueden tener hasta 255 caracteres (incluyendo la ruta completa para llegar), y los únicos caracteres que no permite utilizar son :

• Barra : /

• Nulo (carácter especial)

En Linux , las barras se utilizan como separadores de niveles de directorios. Asimismo, se aconseja no usar interrogantes (?), Asteriscos (*), ampersand (&) y signos de dólar ($) en los nombres, ya que son caracteres con usos específicos en el terminal que podrían complicar la ejecución de órdenes.

A diferencia de los sistemas Windows, la extensión en Linux no se usa para determinar el tipo de archivo, sino que se convierte en una convención recomendada para que los usuarios puedan reconocerlo, sin que el sistema las requiera. Para reconocer los tipos de archivos, Linux utiliza el concepto de *magic numbers*.

Los *magic numbers* son unos caracteres alfanuméricos, típicamente dos bytes (octetos) situados en el inicio de los ficheros, que sirven para identificar el tipo de ficheros. Como se puede intuir, los *magic numbers* son un mecanismo mucho más robusto para reconocer tipos de archivo que la utilización de extensiones de archivos.

Otra de las particularidades más peculiares de Linux, relacionada con el uso de extensiones, es que los ficheros pueden empezar por punto (es decir, no tener la parte de

nombre, propiamente dicha). Por convención, se establece que **los ficheros que comienzan por punto son ficheros ocultos**.

Linux, a diferencia de Windows, es un sistema *case sensitive*. Esto implica que *os.jpg*, *Os.jpg* y *OS.jpg* son tres ficheros completamente diferentes para este sistema.

3. Estructura de directorios en Linux

La estructura de directorios de Linux es muy similar a la de Unix, e incluso a la de Windows. Se ha diseñado como una jerarquía en árbol. En el nivel superior del árbol se encuentra el directorio principal, al que suele llamarse **directorio raíz** (o **directorio raíz del sistema** o **directorio root**). Se le representa con / (barra). Es el directorio único que hay en ese nivel.

No hay que confundir el directorio raíz del nivel superior (/) con el directorio */root*. El directorio */root* es un subdirectorio del directorio /.

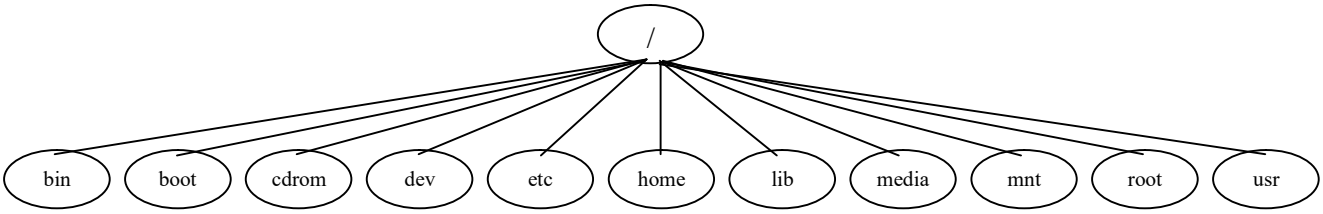
En Linux los nombres de los archivos y directorios son *case sensitive*, es decir, distinguen entre mayúsculas y minúsculas. Para Linux una letra en mayúscula es un carácter totalmente distinto a la minúscula.

En Linux, la barra inclinada o slash (/) se usa para indicar rutas a directorios específicos o para hacer referencia al directorio raíz. Microsoft,en sus SO usa la barra invertida o backslah (\).

Otro aspecto importante que hay que destacar es que el sistema de ficheros de Linux está incluido bajo un gran árbol que se extiende por una o más unidades de disco. Eso significa que para acceder a las diferentes unidades de disco no se hace a través de letras de unidad distintas como en DOS o Windows. En su lugar, todas las particiones se montan como subdirectorios que residen bajo el directorio raíz del sistema de archivos.

Del mismo modo, incluso a los dispositivos de HW se accede como archivos. Por supuesto no son archivos corrientes, pero facilita mucho la gestión de por ejemplo los permisos.

Al instalar Linux se crean unos directorios por defecto necesarios para el buen funcionamiento del sistema. Algunos de ellos son los siguientes:



/bin

El directorio */bin* contiene utilidades del sistema y de usuario. Se trata de utilidades que casi siempre son código compilado y a las que se les llama **binarios** o ejecutables. Algunas utilidades en */etc* son realmente scripts en la shell. Se puede determinar cuál es cuál usando el comando *file*.

/boot

Este directorio contiene componentes cruciales para el proceso de arranque, incluyendo un directorio llamado *grub*. Desde aquí se puede configurar el menú de arranque.

/cdrom

Antiguamente aquí se montaban los CD-ROM's. Actualmente es un enlace a */media/cdrom*, está ahí por compatibilidad con programas antiguos.

/dev

El directorio */dev* es el lugar donde residen todas las definiciones de dispositivos. Los controladores están en otro directorio pero la definición de un dispositivo está aquí. El directorio */dev/disk* contiene los archivos que representan las discos y sus particiones; */dev/psaux* representa al ratón; */dev/ram** representa la memoria RAM. La mayor parte del contenido de este directorio se genera durante el arranque.

/etc

Este directorio bien podría haberse llamado */config* (pero se llama */etc*) ya que contiene archivos de configuración, tanto del sistema como de programas o servicios.

/home

Es el directorio donde se almacenan los datos de usuario. Por ejemplo, cada usuario tiene un directorio personal dentro del directorio */home* cuando se activa su cuenta. Al iniciar sesión, el usuario queda ubicado en su directorio principal.

El directorio personal de cada usuario se representa por el símbolo ~.

Dentro existen subdirectorios que hacen referencia a Documentos (*~/Documentos*) o al escritorio (*~/Escritorio*).

/lib

Contiene las bibliotecas necesarias para que se ejecuten los programas que tenemos en */bin* y */sbin*. En los sistemas de 64bits hay un enlace lib64 que apunta a */lib*.

/lost+found

Las utilidades del sistema de archivos para corregir errores del sistema almacenan datos (inodos perdidos) en este directorio cuando no se pueden recuperar.

/media

Aquí encontramos todas las unidades físicas que tenemos montadas. Discos duros, unidades de dvd, pendrive, ...

/mnt

Este ha sido el lugar tradicional para montar unidades, ha perdido gran parte de su función en favor de */media* pero sigue siendo útil para el montaje puntual de algunas cosas.

/proc

Este directorio es realmente un sistema de archivos especial. Contiene información detallada sobre procesos que están en ejecución, así como información específica sobre configuraciones de HW.

/root

Este directorio es el directorio principal (carpeta personal) del usuario **root**. Por omisión

el usuario *root* inicia sesión entrando en este directorio. Aquí sólo se guardan archivos específicos del usuario *root*. Equivaldría al directorio (que no existe) */home/root*.

/sbin

Similar a */bin* que contiene binarios esenciales del sistema que típicamente sólo son ejecutados por el superusuario.

/srv

Datos de los servicios proporcionados por el sistema (es más habitual en servidores).

/tmp

Archivos temporales. Es habitual planificar la limpieza periódica y automática de este directorio (muchas veces, al reiniciar el sistema o la sesión).

/usr

Este es el directorio más extenso del sistema. Contiene archivos de los programas no esenciales. Por ejemplo, la mayor parte de la documentación está en */usr/doc*. El árbol completo de fuentes del Kernel está en */usr/src*.

/var

Este directorio se utiliza para mantener datos variables, tales como trabajos de impresión, configuración y datos de servidores, archivos administrativos, etc.

No se recomienda hacer modificaciones sobre la estructura de estos directorios, ya que son básicos para el buen funcionamiento del sistema. Sin embargo, no hay ninguna restricción en cuanto a la creación de nuevos directorios en la raíz del sistema.

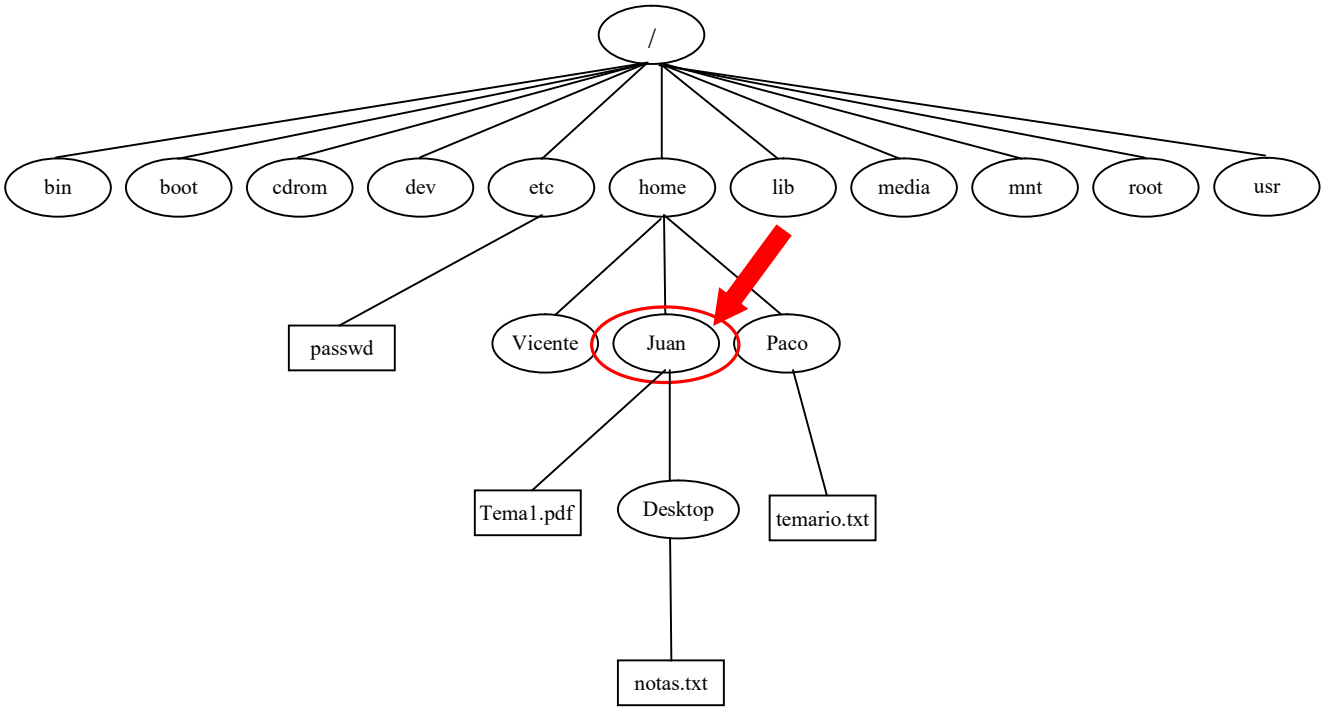
Rutas

Al igual que en Windows, Linux utiliza rutas absolutas y relativas para hacer referencia al nombre y ubicación de los ficheros y carpetas.

Como se ha señalado, la principal diferencia entre los dos sistemas es que el sistema de ficheros de Linux está incluido bajo un gran árbol que se extiende por una o más unidades de disco. Eso significa que para acceder a las diferentes unidades de disco no

se hace a través de letras de unidad distintas como en Windows. En su lugar, todas las particiones se montan como subdirectorios que residen bajo el directorio raíz del sistema de archivos.

La otra diferencia es que se utilizan las barras (/) como separadores, en lugar de las barras invertidas (\) de Windows, y los nombres son *case sensitive*.



Supongamos que dada esa estructura de directorios en nuestro sistema, nos encontramos situados en el directorio *Juan*. Nuestro directorio actual es pues *Juan*, cuya ruta absoluta es:
/home/Juan

Estando en ese directorio vamos a ver cómo se nombran algunos ficheros utilizando rutas absolutas y relativas:

- *Tema1.pdf* utilizando una Ruta Absoluta:
  /home/Juan/Tema1.pdf
- *Tema1.pdf* utilizando una Ruta Relativa:
  Tema1.pdf
  ó
  ./Tema1.pdf
- *notas.txt* utilizando una Ruta Absoluta:

/home/Juan/Desktop/notas.txt

- *notas.txt* utilizando una Ruta Relativa:
  Desktop/notas.txt
  ó
  ./ Desktop/notas.txt
- *temario.txt* utilizando una Ruta Absoluta:
  /home/Paco/temario.txt
- *temario.txt* utilizando una Ruta Relativa:
  ../Paco/temario.txt
- *passwd* utilizando una Ruta Absoluta:
  /etc/passwd
- *passwd* utilizando una Ruta Relativa:
  ../../etc/passwd

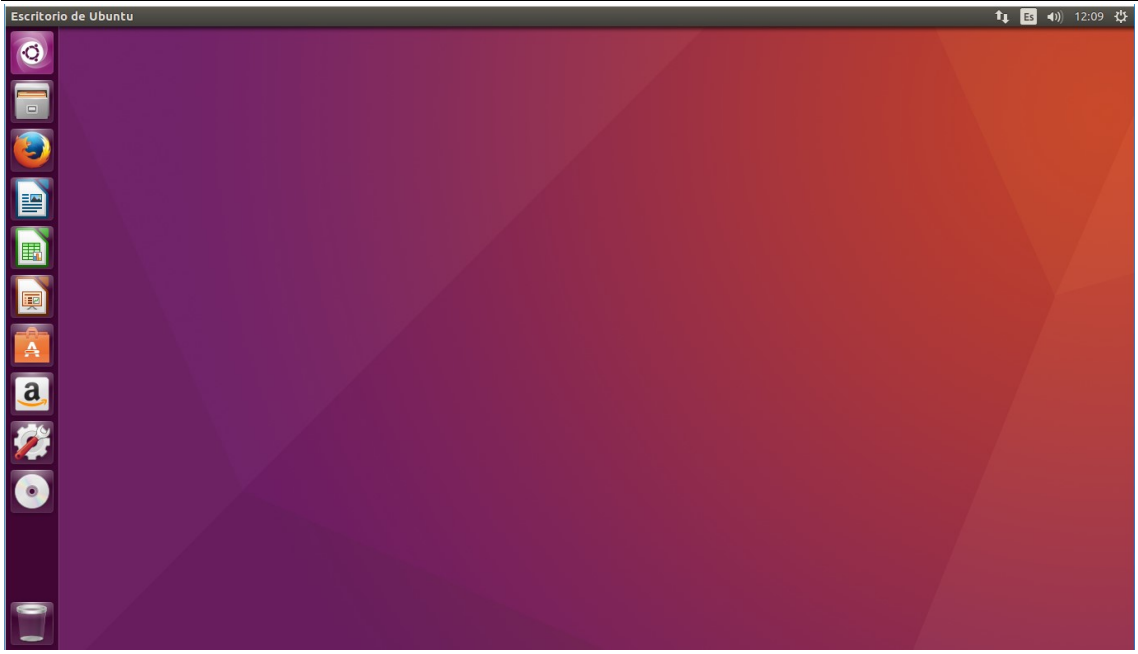
4. Entorno gráfico de Linux

Todos los sistemas operativos basados en interfaces gráficas (GUI) utilizan un entorno de escritorio. Los entornos de escritorio abarcan muchas cosas, incluyendo las siguientes (pero no únicamente):

- La apariencia del sistema
- La forma en que el escritorio está dispuesto
- Cómo navegar por el escritorio

En las distribuciones de Linux (como Ubuntu) existen varios entornos de escritorio disponibles. Ubuntu usa **Unity**[2] como entorno de escritorio predeterminado. Después de instalar e iniciar sesión en Ubuntu se ve el escritorio Unity. Esta vista inicial se compone del fondo de escritorio y de dos barras, una situada horizontalmente en la parte superior del escritorio llamada barra de menús, y la otra, orientada verticalmente en el lado izquierdo de la pantalla, denominada lanzador.

[2] La versión 16.04 LTS utiliza Unity como entorno predeterminado, pero la última versión no LTS, la 17.10, ha recuperado Gnome como entorno gráfico pedeterminado. La próxima LTS incorporará Gnome.

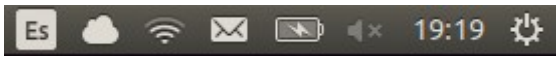


El fondo de escritorio

Bajo la barra de menús en la parte superior de la pantalla hay una imagen que cubre completamente el escritorio. Este es el fondo por defecto del escritorio, que pertenece al tema predeterminado de Ubuntu 16.04 conocido como **Ambiance**.

La barra de menús

La barra de menús incluye funciones usadas habitualmente en Ubuntu. El grupo de iconos situados en el extremo derecho de la barra de menú se llama área de indicadores, o área de notificaciones. Cada instalación de Ubuntu puede variar ligeramente en cuanto a los tipos y números de iconos en función de ciertos factores, incluyendo el tipo de hardware y los periféricos integrados disponibles en el sistema en el que está instalado Ubuntu. Algunos programas añaden un icono al área de indicadores durante la instalación (por ejemplo, Ubuntu One).

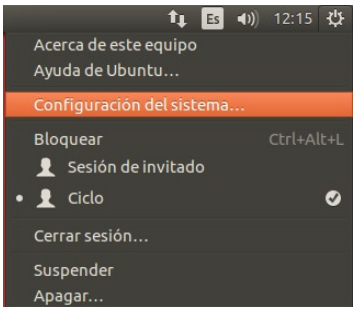


Los indicadores más comunes son:

- Indicador del teclado: Permite seleccionar una distribución de teclas, si se elige más de una de ellas. El menú del indicador del teclado contiene los siguientes

elementos: Mapa de caracteres, Distribución del teclado y Configuración de entrada de texto

- Indicador de red: gestiona las conexiones de red, permitiéndo conectarse fácil y rápidamente a una red cableada o inalámbrica.
- Indicador Ubuntu One: permite realizar copias de seguridad en línea y compartir archivos. Ya ha desaparecido en las últimas versiones.
- Indicador de mensajería: incorpora las aplicaciones sociales. Desde aquí se puede acceder, entre otros, a mensajería instantánea y a los clientes de correo electrónico.
- Indicador de sonido: proporciona una forma fácil de ajustar el volumen de sonido así como de acceder al reproductor de música y configuración de sonido.
- Reloj: muestra la hora actual y proporciona un enlace al calendario y a la configuración de fecha y hora.
- Indicador de sesión: es un enlace a la **configuración del sistema**, a la ayuda de Ubuntu y a las opciones de sesión (como el bloqueo del equipo, las sesiones de usuario o de invitado, el cierre de sesión, el reinicio del equipo o el apagado completo del mismo).



Cada aplicación tiene un sistema de menús donde se pueden ejecutar las diferentes acciones de una aplicación (como Archivo, Editar, Ver, etc.); el sistema de menús de una aplicación se llama, apropiadamente, menú de aplicación. En Unity, el menú de aplicación no está en la barra de título de la aplicación como es el caso habitual en otros entornos GUIs. En lugar de eso, está situado en la zona izquierda de la barra del menú. Para mostrar el menú de una aplicación, basta con que se mueva el ratón a la barra de menú del escritorio (en la parte superior de la pantalla). Mientras el ratón esté posicionado ahí, los menús de la aplicación activa se mostrarán sobre la barra de menú del escritorio, permitiendo acceder a las opciones del menú de la aplicación. Cuando se haga clic sobre el escritorio, la barra de menú del escritorio reaparecerá. Esta característica de Unity de mostrar el menú de la aplicación solo cuando se necesita es

especialmente útil para usuarios de netbooks y portátiles ya que proporciona mayor espacio libre en el área de trabajo.

El lanzador

La barra vertical de iconos en la parte izquierda de la pantalla es llamada el lanzador. El lanzador proporciona un fácil acceso a las aplicaciones, dispositivos montados y a la papelera. Todas las aplicaciones en ejecución en su sistema colocarán un icono en este lanzador mientras se encuentren ejecutándose.

Para cambiar el tamaño de los iconos del lanzador, hay que ir a **Indicador de sesión ‣ Configuración del sistema ‣ Apariencia**, pestaña **Aspecto**.

El primer icono en la parte superior del lanzador es el tablero, un componente de Unity. Por defecto aparecen otras aplicaciones en el lanzador, incluyendo el Administrador de Archivos, LibreOffice, Firefox, cualquier dispositivo montado y la Papelera, que contiene las carpetas y archivos eliminados, en la parte inferior del lanzador.

Si se mantiene pulsada la tecla Súper, también denominada tecla Windows (tecla Win), que está situada entre la tecla Ctr y la tecla Alt izquierdas, Ubuntu superpondrá un número en las diez primeras aplicaciones del lanzador y además mostrará una pantalla completa de atajos útiles. Se puede lanzar una aplicación con el número n sobreimpreso tecleando Súper+n.

Si se abren más aplicaciones de las que se pueden mostrar en el lanzador, este «apilará» los iconos de aplicación en su parte inferior. Basta con que se mueva el ratón sobre esa zona para que los iconos del lanzador se «deslicen» y los iconos apilados se desplieguen para permitir su acceso.

Ejecutar aplicaciones

Para ejecutar una aplicación desde el lanzador (o mostrar una aplicación que ya está en ejecución), basta con que se haga clic sobre el icono de la aplicación.

Las aplicaciones que se encuentran en ejecución tendrán uno o más triángulos en la parte izquierda del icono, indicando el número de ventanas abiertas que pertenecen a cada aplicación. Las aplicaciones en ejecución también mostrarán su icono en el lanzador iluminado.

La aplicación en primer plano (es decir, la aplicación que está sobre todas las demás ventanas de aplicación abiertas) se indica mediante un único triángulo blanco en la derecha de su icono.

También puede ejecutar una aplicación mediante el tablero.

Añadir y eliminar aplicaciones del lanzador

Hay dos maneras de añadir una aplicación al lanzador:

- Se abre el tablero, se busca la aplicación que se desea añadir al lanzador y se arrastra su icono sobre el lanzador.
- Ejecutamos la aplicación que se desea añadir al lanzador, hacemos clic derecho sobre el icono de la misma en el lanzador y seleccionamos *Mantener en el lanzador*.

Para eliminar una aplicación del lanzador, basta con hacer clic derecho sobre el icono de la aplicación y seleccionar *No mantener en el lanzador*.

El tablero

El tablero nos ayuda a encontrar rápidamente aplicaciones y archivos en el equipo. El tablero es similar al menú de inicio de Windows o a la pantalla de inicio de Windows. En comparación con versiones anteriores de Ubuntu o alguna distribución de Gnome Linux, el tablero reemplaza a los menús de Gnome. El tablero permite buscar información, tanto en el equipo (aplicaciones instaladas, archivos recientes, marcadores, etc.) como en remoto (Twitter, Google Docs, etc.).

Para explorar el tablero, hay que hacer clic sobre el icono superior del lanzador, el que contiene el logotipo de Ubuntu. Tras pulsar el icono del tablero, en el escritorio se sobrepondrá una ventana traslúcida con una barra de búsqueda en la parte superior, así como un grupo de las aplicaciones, archivos y descargas accedidas recientemente. Ubuntu también incluye resultados de servicios web populares. La barra de búsqueda proporciona resultados dinámicos según se vayan escribiendo los términos de la búsqueda.

Lentes

La búsqueda se lleva a cabo empleado una o más lentes, también conocidas como ámbitos; cada lente es responsable de proporcionar una categoría de resultados de búsqueda para el tablero. Las siete lentes instaladas de forma predeterminada en la parte inferior son enlaces a la lente de Inicio, lente de Aplicaciones, lente de Música, lente de Fotos, lente de Vídeos y lente de Mensajes de redes sociales.

Las lentes actúan como categorías de búsqueda especializadas del tablero. Desde el punto de vista del usuario, las lentes son simplemente iconos. Para un purista de Ubuntu, las imágenes que aparecen en la parte inferior del tablero son lentes; para el resto, son iconos.

Existen muchos sitios en Internet dedicados a la creación y emisión de lentes para el escritorio Unity de Ubuntu. Algunos de esos sitios incluso enseñan cómo crear nuestras propias lentes para maximizar la eficiencia en el uso del interfaz Unity de Ubuntu.

Buscar archivos y aplicaciones con el tablero

El tablero es una herramienta extremadamente potente que nos permite buscar aplicaciones y archivos por todo el equipo.

El tablero nos puede ayudar a encontrar nombres de archivos o carpetas. Simplemente tecleando una parte del nombre de archivo o carpeta. A medida que se vaya escribiendo, los resultados aparecerán en el tablero. La lente de Archivos y carpetas puede ayudar también a buscar archivos o carpetas, mostrando los elementos accedidos más recientemente así como las últimas descargas. Se puede usar el botón Filtrar resultados, situado en la esquina superior derecha del tablero para refinar los resultados por hora de modificación del archivo o carpeta, tipo de archivo (.odt, .pdf, .doc, .tex, etc.) o tamaño.

La instalación estándar de Ubuntu viene con muchas aplicaciones. Los usuarios pueden también descargar miles de aplicaciones más desde el Centro de software de Ubuntu. Según se vaya recopilando una colección de aplicaciones, puede resultar difícil recordar el nombre de una aplicación en particular. La lente de aplicaciones del tablero puede ayudar en esta búsqueda. Esta lente clasifica automáticamente en categorías las aplicaciones instaladas en «Usadas recientemente», «Instaladas» y «Más sugerencias». También se puede introducir el nombre aplicaciones cuyos nombres cumplan un criterio de búsqueda. Incluso si no se recuerda el nombre de la aplicación en absoluto, se puede escribir una palabra clave que esté relacionada con la aplicación y el tablero la encontrará en la mayoría de los casos. Por ejemplo, al introducir música el tablero devolverá el reproductor de música predeterminado, y cualquier otro reproductor de música que se haya usado.

Resultados de búsqueda externos

Además de buscar aplicaciones y archivos localmente en el equipo el tablero también buscará en varias ubicaciones en línea (por ejemplo, Amazon.com). Los resultados pertinentes para los criterios de búsqueda se muestran en el tablero. Para evitar que los

términos locales de sus búsquedas se envíen a Internet se puede desactivar esta característica en la sección **Privacidad** de la Configuración del sistema.

Los resultados de las búsquedas en línea dentro del tablero se activan de forma predeterminada durante la instalación. Si no se desea resultados de búsquedas externas, hay que ir a **Configuración del sistema ‣ Privacidad ‣ Resultados de la búsqueda** y dejar apagado el interruptor «*Incluir resultados de búsquedas en línea*». Como una medida adicional de privacidad se puede también impedir que cualquier actividad de búsqueda sea grabada.

Áreas de trabajo

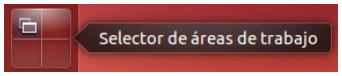
Las áreas de trabajo también se conocen como escritorios virtuales. Estas vistas separadas del escritorio permiten agrupar las aplicaciones y, al hacerlo, reducir el desorden y mejorar la navegación en el escritorio.

Por ejemplo, en un área de trabajo se puede abrir todas las aplicaciones multimedia; en otra las aplicaciones ofimáticas, y un navegador de Internet en una tercera área de trabajo. Ubuntu, de forma predeterminada, tiene cuatro áreas de trabajo.

La característica de las áreas de trabajo no está activa de forma predeterminada en Ubuntu 16.04. Para activar las áreas de trabajo hay que pulsar en **Configuración del sistema ‣ Apariencia** y luego en la pestaña **Comportamiento** y finalmente activar la casilla *Activar las áreas de trabajo*. Cuando esta casilla está activada se añade un icono más, que aparenta ser un panel de ventanas, en la parte inferior del lanzador, es el selector de áreas de trabajo.

Cambiar entre áreas de trabajo

Si se ha activado la característica del selector de áreas de trabajo como se ha descrito arriba, se puede cambiar entre áreas de trabajo pulsando sobre el icono del selector de áreas de trabajo situado en el lanzador. Esta utilidad le permite alternar entre las áreas de trabajo (contengan o no aplicaciones abiertas), y elegir la que desea usar.



Opciones de sesión

Cuando hemos terminado de trabajar con el equipo, podemos elegir entre cerrar sesión, suspender, reiniciar o apagar, mediante el **Indicador de sesión** en el extremo derecho del panel superior.

Cerrar sesión

Al cerrar la sesión se dejará el equipo corriendo, pero se volverá a la pantalla de inicio de sesión. Esto es útil para cambiar entre sesiones de usuario, como cuando una persona distinta desea iniciar sesión en su cuenta, o si se nos ha informado que «*cierre sesión y vuelva a iniciarla*». También se puede cerrar la sesión pulsando las teclas Ctrl+Alt+Supr. Antes de cerrar sesión, hay que comprobar siempre que se ha guardado el trabajo en todas las aplicaciones abiertas.

Suspensión

Para ahorrar energía, se puede dejar el equipo en modo suspendido, lo cual guardará las aplicaciones actualmente abiertas en la memoria interna, apagará todos los dispositivos, y permitirá retornar a dicho estado más rápidamente. Mientras se encuentra estado de suspensión el equipo usará el mínimo de energía; esto es necesario ya que la sesión es guardada en la memoria interna, y sin alimentación a esa memoria los datos se perderían.

Para poner su equipo en modo suspendido, hay que seleccionar **Suspender** en el «*indicador de sesión*».

Reiniciar

Para reiniciar el equipo hay que seleccionar **Apagar** en el «*Indicador de sesión*» y pulsar sobre el icono de *Reiniciar*.

Apagar

Para apagar completamente el equipo, hay que seleccionar **Apagar** del «*indicador de sesión*» y pulse sobre el icono de *Apagar*.

Otras opciones

En el «indicador de sesiones» podemos seleccionar *Bloquear/Cambiar de cuenta* para bloquear la pantalla del usuario actual o cambiar de cuenta de usuario.

Se puede bloquear la pantalla rápidamente usando el atajo de teclado Ctrl+Alt+L. Se recomienda bloquear la pantalla si se va a alejarse del equipo por cualquier periodo de tiempo.

5. Gestión de la información con interfaz gráfica

Así como Windows tiene el Explorador de Windows y Mac OS tiene Finder para navegar por los archivos y carpetas, Ubuntu 16.04 usa *Archivos* como administrador de archivos predeterminado. Realmente, el programa es el *Nautilus* en su versión 3.14. Nautilus es el explorador de ficheros que acompaña al escritorio gráfico Gnome y Unity.

La ventana del gestor de archivos *Archivos*

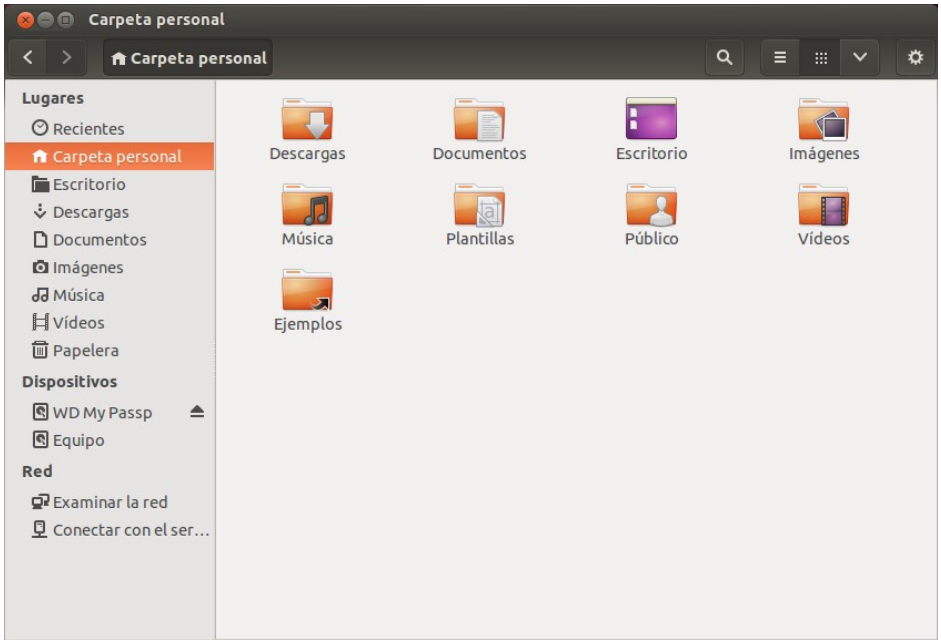
Al seleccionar el atajo Archivos en el lanzador, pulsar sobre una carpeta en el tablero o hacer doble clic sobre una carpeta del escritorio se abrirá la ventana del administrador de archivos. La ventana predeterminada contiene las siguientes características:

- *Barra de menús*: La barra de menús está situada en la parte superior de la pantalla. El menú Archivos permite modificar la disposición del navegador, mostrar, navegar y eliminar marcadores, abrir un documento de ayuda, abrir una nueva ventana, conectarse a un servidor o salir. Elegimos *Introducir lugar* para abrir el campo de texto de ubicaciones en el que se puede introducir cualquier ubicación directamente.
- *Barra de título*: La barra de título muestra el nombre de la carpeta actualmente seleccionada. También contiene los botones Cerrar, Minimizar, y Maximizar.
- *Barra de herramientas*: En la parte derecha de la barra de herramientas se disponen cinco botones: *Buscar* (representado por una lupa), *Ver elementos como lista*, *Ver elementos como rejilla de iconos, Opciones de visualización y Opciones de la ubicación* (representado por una rueda dentada). Pulsando en el icono *Buscar* se abre un campo para que pueda buscar un archivo o carpeta por su nombre. Pulsando en los botones *Opciones de la ubicación* o *Opciones de la vista* abre un menú con varias opciones. En la parte izquierda de la barra de

herramientas se verá una representación de la navegación actual. Es similar a la función de historial de la mayoría de los navegadores; guarda el registro de dónde ha estado y nos permite retroceder si es necesario. Se puede hacer clic sobre cualquiera de las ubicaciones para recuperarla mediante al explorador de archivos. Los botones *Ir al lugar anterior* visitado y *Ir al siguiente lugar* visitado permiten moverse por las ubicaciones visitadas anteriormente.

- *Panel izquierdo*: El panel izquierdo del explorador de archivos, con el título *Lugares*, contiene atajos a las carpetas usadas habitualmente. También crear un marcador de una carpeta mediante la barra de herramientas seleccionando **Opciones de la ubicación** (rueda dentada) ‣ **Añadir este lugar a marcadores**. Una vez que ha creado el marcador de la carpeta aparecerá en el panel izquierdo. Independientemente de la carpeta que esté abierta, el panel izquierdo siempre contiene las mismas carpetas.
- *Panel derecho*: El panel más grande muestra los archivos y carpetas dentro de la carpeta en la que actualmente estamos explorando.

Para explorar una carpeta, hacemos doble clic sobre su icono; puede encontrarse en el panel derecho, en el izquierdo o en la barra de herramientas.



Abrir archivos

Un archivo, en su forma más simple, son datos. Los datos pueden representar un documento de texto, la información de una base de datos u otro medio como música o

vídeos. Para abrir un archivo se puede hacer doble clic sobre su icono. Ubuntu intentará determinar qué aplicación es la más conveniente para abrir el archivo seleccionado. En algunos casos puede que deseemos abrir el archivo usando alguna otra aplicación distinta de la seleccionada por Ubuntu. Para emplear una aplicación hacemos clic derecho sobre el icono y seleccionamos una opción de las disponibles en *Abrir* con otra aplicación.

Crear carpetas nuevas

Para crear una nueva carpeta dentro del administrador de archivos, hacemos clic derecho en una región vacía del panel derecho y seleccionamos *Nueva carpeta* en el menú emergente (esta acción funciona también en el escritorio). También se puede crear una nueva carpeta pulsando Ctrl+Mayús+N.

Archivos y carpetas ocultos

Si queremos ocultar ciertas carpetas o archivos, basta con añadir un punto (.) al comienzo del nombre (por ejemplo, «*.Finanzas personales*»).

Se pueden ver fácilmente los archivos ocultos pulsando **Opciones de la vista ‣ Mostrar archivos ocultos** o pulsando **Ctrl+H**. Ocultar los archivos cuyo nombre comienza por un punto (.) no es una medida de seguridad, sino que es simplemente una forma de ayudarnos a organizar los archivos.

Copiar y mover archivos y carpetas

Podemos cortar, copiar y pegar archivos o carpetas en el administrador de archivos haciendo clic derecho sobre el elemento y seleccionando el botón correspondiente en el menú emergente. También se pueden usar los atajos de teclado **Ctrl+X, Ctrl+C** y **Ctrl+V** para cortar, copiar y pegar archivos y carpetas, respectivamente.

Se pueden seleccionar varios archivos haciendo clic en una zona vacía (por ejemplo, fuera de cualquier archivo o carpeta), manteniendo el botón del ratón pulsado, y arrastrando el cursor por encima de los archivos o carpetas deseados. Esta acción de «*hacer clic y arrastrar*» es útil para seleccionar elementos que están agrupados juntos. Para seleccionar varios archivos o carpetas que no están dispuestos unos junto a otros,

mantenemos pulsada la tecla **Ctrl** mientras va haciendo clic sobre cada elemento individual.

Una vez se hayan seleccionado los archivos y carpetas deseados, hacemos clic derecho sobre cualquiera de los elementos seleccionados para ejecutar la acción, igual que haríamos con un único elemento.

Cuando hayamos *«copiado»* uno o más elementos, navegamos hasta la ubicación deseada, luego hacemos clic derecho sobre una zona vacía de la ventana y seleccionamos *Pegar* para copiarlos a la nueva ubicación. Mientras que la orden *copiar* se puede usar para duplicar archivos o carpetas en una nueva ubicación, la orden *cortar* se puede usar para moverlos a otro sitio. O lo que es lo mismo, se creará una copia en la nueva ubicación, y el original de eliminará de su ubicación actual.

Cuando «cortamos» o «copiamos» un archivo o una carpeta, no pasará nada hasta que se «pegue» en algún lado. «Pegar» solo afectará a los elementos más recientemente cortados o copiados.

Para mover un archivo o una carpeta, seleccionamos el elemento a mover, luego hacemos clic sobre **Editar ‣ Pegar**. Si hacemos clic sobre un archivo o carpeta, lo arrastramos, mantenemos la tecla Alt y la soltamos en la carpeta de destino, aparecerá un menú preguntándonos si deseamos copiar, mover o enlazar el elemento.

Igual que con la orden de copiar anterior, también se puede realizar esta acción usando el menú contextual (clic derecho), y funcionará para varios archivos y carpetas a la vez.

Un modo alternativo para mover un archivo o carpeta es hacer clic sobre el mismo y arrastrarlo a una nueva ubicación.

Usar múltiples pestañas y múltiples ventanas de Archivos

Abrir varias ventanas del administrador de archivos Archivos puede resultar útil para arrastrar archivos y carpetas entre distintas ubicaciones.

También se pueden emplear varias pestañas para explorar distintas ubicaciones de forma simultánea.

Para abrir una segunda ventana mientras explora una carpeta en Archivos, seleccionamos **Archivo ‣ Ventana nueva** o pulsamos **Ctrl+N**. Esto abrirá una nueva ventana, permitiendo arrastrar archivos o carpetas entre dos ubicaciones. Para abrir una nueva pestaña, hacemos clic en **Opciones de la ubicación** (rueda dentada) ‣ **Pestaña nueva** o pulsamos Ctrl+T. Aparecerá una nueva fila sobre el espacio usado para explorar los archivos, que contiene dos pestañas, ambas mostrarán el directorio que estaba explorando originalmente. Podemos pulsar sobre esas pestañas para cambiar entre ellas, y arrastrar archivos o carpetas entre las pestañas, igual que haríamos entre ventanas.

Cuando arrastremos elementos entre ventanas o pestañas aparecerá un pequeño símbolo sobre el cursor del ratón, para hacerle saber la acción que se va a efectuar cuando se suelte el botón del ratón. Un signo más (+) indica que se está a punto de copiar el elemento, mientras que una flecha pequeña significa que el elemento será movido. La acción por defecto dependerá de las carpetas que se esté usando.

6. Shell de comandos

Una vez nos hemos familiarizado con la vía de interactuación gráfica, vamos a centrarnos en la interfaz texto conocida como consola o terminal.

Al igual que Unix, Linux ofrece el mecanismo de consolas o terminales virtuales. Este consiste en que a partir de una entrada (el teclado) y con una salida (el monitor) se simulen varias terminales, donde el mismo, o distintos usuarios puedan conectarse indistintamente. De esta forma es posible tener más de una sesión abierta en la misma máquina y trabajar en ellas. Este mecanismo también facilita la característica multiusuario del sistema Linux pues las diferentes conexiones se pueden establecer con diferentes usuarios.

Por defecto, las consolas desde la uno a la seis tienen asociado un programa que permite conectarse al sistema en modo texto, mientras que la siete, si se instaló y activó el "modo gráfico", constituye una consola gráfica.

El cambio de una consola a otra se realiza a través de la combinación de teclas Alt y Fx (las teclas de Función), donde x oscila entre 1 y 12. De esta forma se pueden acceder un total de 24 consolas virtuales: para las doce primeras se utiliza el Alt izquierdo y para las otras doce el derecho. Por ejemplo para llegar a la consola 16 se presionarían las teclas Alt derecho y F4. No obstante normalmente solo se puede acceder a las consolas con algún proceso o funcionalidad definida.

Desde una consola gráfica para cambiar a otra tipo texto se debe además presionar la tecla Ctrl, pues las combinaciones Alt + Fx son capturadas e interpretadas por las aplicaciones gráficas de otra forma.

Con la tecla Alt izquierda combinada con los cursores (derecho e izquierdo) se puede además, realizar un movimiento circular entre todas aquellas consolas que tengan un proceso asociado (texto, gráfico, etc.).

Si nos remitimos a una consola texto podremos apreciar que en ella se mostrará, el nombre de la distribución, la versión de la distribución, la versión del kernel y la arquitectura de la máquina. También aparecerá el nombre que se le asignó al sistema en la instalación y la palabra **login**. Aquí puede entrarse el login de un usuario del sistema. Luego se pedirá el **password** que al ser entrado no se muestra ningún eco en la pantalla. Si ambos son válidos se establecerá la conexión y se mostrará lo que se conoce como prompt, con forma similar a esta:
[alumno@pinguino:/root]$

Para abreviar, en lo sucesivo usaremos el carácter $ para hacer referencia al prompt de un usuario común, y # para el prompt del superusuario root.

Al igual que en Unix, el programa que se ejecuta en Linux cada vez que un usuario se conecta y que le permite interactuar con el sistema se conoce como shell (bash normalmente). Este es capaz de interpretar una gran gama de comandos y sentencias.

Constituye a su vez un poderoso lenguaje de Scripting (programación). Muchos programas y comandos son shell scripts, por lo que el conocer este lenguaje permite no sólo entenderlos y ajustarlos a las necesidades particulares sino también, hacer nuevos scripts para automatizar diversas tareas.

Existen muchos tipos de shells orientados a caracteres para Linux.
Ejemplos:

- **ash**: A shell
- **csh**: C shell
- **tcsh**: una extensión al C shell
- **ksh**: Korn Shell
- **bsh**: Bourne Shell
- **bash**: Bourne Again Shell

Algunos shells se originan de otros o toman sus características más avanzadas. De los mencionados anteriormente **bash** es el más empleado en Linux. Este es un producto de la FSF (Free Software Foundation). Es uno de los más desarrollados y está portado para múltiples plataformas. Toma las facilidades de csh y ksh. Ofrece entre otras posibilidades las siguientes:

- Auto completar durante la escritura. Al teclear uno o varios caracteres se puede pulsar TAB con el objetivo de que en caso de que pueda completarse de forma unívoca un comando, nombre de fichero o una variable (en dependencia del contexto), complete de forma automática (se escriba el resto de la palabra). Si existieran varias posibilidades para completar la palabra, se oirá un sonido y volviendo a pulsar TAB se mostrarán en pantalla todas las posibilidades existentes. En caso de existir muchas posibilidades (por defecto más de 100) se pregunta si se desea mostrarlas todas o no.
- Historial de comandos. Esta es una facilidad de muchos otros shells que permite el movimiento a través de los últimos N comandos ejecutados, en la sesión actual o en las anteriores. N por defecto es 1000, pero puede modificarse. Para moverse arriba y abajo se suelen utilizar los cursores.
- Poderosas estructuras de control para realizar scripts. (Procesos por lotes). Se pueden utilizar instrucciones if, for, while, select, case, etc.

- Definición de funciones y alias para comandos. Las funciones permiten definir subrutinas programadas usando el lenguaje de bash y los alias, asociar nombres a comandos con ciertas opciones y argumentos de forma más nemotécnica o abreviada.

En todas las distribuciones de GNU/Linux modernas, siempre tenemos que crear como mínimo un usuario durante el proceso de instalación. En algunas de las distribuciones, este usuario no tiene derechos de administración, es decir: es un usuario que puede utilizar el sistema, pero que no lo puede administrar. En este sentido podemos decir que este es el perfil de usuario predeterminado. Ubuntu, el usuario que crea durante la instalación sí que tiene privilegios de administrador.

Además del usuario predeterminado, en el proceso de instalación se crea un usuario especial, llamado **root** con privilegios de administrador. Este es el administrador primario y tiene ciertos privilegios que los demás administradores no tienen. Es usual asignar la contraseña de *root* durante la instalación, aunque no siempre se hace, de hecho, al instalar Ubuntu no nos la ha solicitado. Veremos cómo cambiársela.

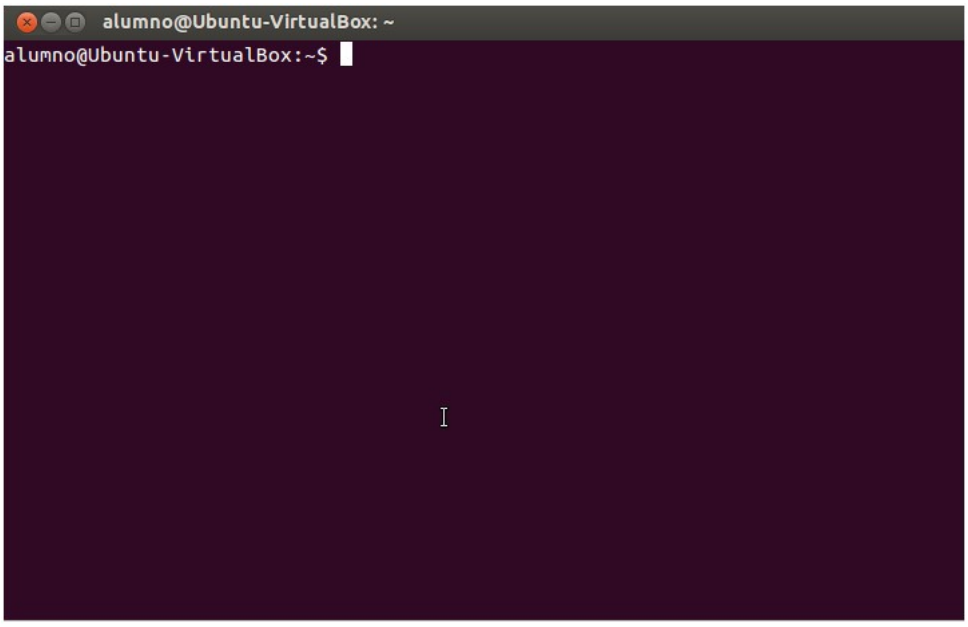
Se considera una mala idea conectarse gráficamente utilizando el nombre de usuario *root*, la cuenta del administrador del sistema, ya que los entornos gráficos incluyen la ejecución de muchos programas extras, que en el caso que nos conectáramos utilizando el nombre de usuario *root* funcionarían con permisos excesivos. Para reducir riesgos al mínimo, hay que usar una cuenta de usuario normal (aunque sea con privilegios de administrador) para conectarse a la interfaz gráfica. Debemos tener muy presente este consejo general para todos los usos que hacemos de la cuenta *root*: sólo accederemos como administradores del sistema cuando se requieran privilegios extras.

Una vez hemos accedido al sistema con un usuario que no sea *root*, nos presentará el escritorio. Durante la visualización de los menús, nos daremos cuenta que se pueden hacer muchas cosas sin tener que introducir comandos desde el teclado: la mayoría de usuarios tendrán suficiente con la interfaz gráfica. Pero hay operaciones de una complejidad superior que requieren que el administrador del sistema y de la red hurgue por las interioridades del sistema. Necesitan una herramienta más potente que un ratón para manejar todas las tareas, que es el intérprete de comandos y que en modo gráfico la

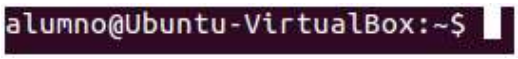
activamos abriendo una ventana de terminal. Esta ventana de terminal también recibe el nombre de **consola**.

Para abrir una ventana de terminal o xterm (X es el nombre del software que se encarga de hacer que el entorno gráfico pueda funcionar) hay que ir al tablero y buscar la aplicación Terminal. Sería conveniente, una vez ejecutándose, anclarla al Lanzador.

La ventana de terminal nos da el control del sistema, desde la que se puede hacer casi todo en el sistema. Cada ventana de terminal que se abre debería mostrar siempre un indicador de comandos (command prompt). Así pues, la presencia de este indicador es cómo el sistema indica que se encuentra preparado para recibir las órdenes del usuario y ejecutarlas.



El indicador de comandos (prompt) estándar suele mostrar el nombre de entrada del usuario y el directorio de trabajo actual, representado por una tilde (carácter ~ , se puede sacar con la combinación de teclas Alt Gr + 4).



Un indicador de comandos puede mostrar información diversa, la cual no forma parte de las órdenes que introducimos en el sistema. El tipo de información que suele contener es el nombre de usuario con el que hemos entrado en el sistema, el nombre de la máquina, el directorio actual, la hora, etc.

[usuario@host:dir]$

Aquí, usuario será nuestro nombre de acceso (*alumno*), host será el nombre de la máquina a la que hemos accedido (*Ubuntu-VirtualBox*), y dir será una indicación de nuestra ubicación actual en el sistema de archivos (~).

Si el indicador de comandos se visualiza en pantalla con el carácter **$** seguido de un cursor para invitar al usuario a la introducción de órdenes, entonces el usuario no tiene privilegios de administración. Si el carácter es el símbolo # en lugar de **$**, significa que estamos con privilegios de administrador.

Podemos cambiar el aspecto del prompt cambiando la variable de entorno PS1:

```
alumno@Ubuntu-VirtualBox:~$ PS1="Nuevo Prompt: "
Nuevo Prompt: export PS1
Nuevo Prompt: PS1="(\t)[\u]$ "
(12:49:30)[alumno]$ export PS1
```

Ubuntu incorpora un mandato especial llamado **sudo** que permite ejecutar cualquier comando con privilegios de administrador (sin necesidad de ser root). Sí que es necesario que el usuario que ejecuta ese mandato sea del grupo administradores. Es el equivalente al UAC (User Access Control) que vimos en Windows que aparecía la ventana pidiendo conformidad cuando se realizaba una tarea de administración del sistema. Sólo los usuarios pertenecientes al grupo administradores podían realizar dicha tarea (realmente son los usuarios *sudoers*, lo veremos en el próximo tema).

Para ejecutar una orden con privilegios de administrador bastará con escribir el mandato *sudo* delante de dicha orden.

```
$ sudo cat /etc/shadow      #Ejecuta el mandato cat /etc/shadow
                            con privilegios de administrador
```

A continuación nos solicitará nuestra contraseña.

En muchas distribuciones, para iniciar una sesión como usuario administrador debemos escribir en la ventana de terminal:

```
$ su -
```

Y escribir la contraseña de *root*.

En la distribución Ubuntu, el usuario con privilegios de administrador (sin necesidad de ser *root*), mediante su contraseña, puede acceder a las tareas de administración sin necesidad de teclear **sudo** cada vez. Podemos acceder a una consola como root si escribimos:

```
$ sudo su -
```

o bien:

```
$ sudo bash
```

En este caso habrá que escribir la contraseña del usuario con derechos de administración.

Cabe decir que muchas de las tareas de administración en sistemas GNU/Linux implican la modificación de una serie de archivos de configuración del sistema. Habitualmente, estos archivos de configuración están dentro del directorio */etc*, independientemente de la distribución de GNU/Linux que utilizamos.

Por último, hay que destacar que cada usuario creado en el sistema tiene un directorio personal asignado, normalmente nombrado con el mismo nombre de usuario. Este directorio se le conoce como *home* (o $HOME) y se encuentra ubicado bajo el directorio ***/home***. Por defecto, cada vez que un usuario abre sesión en un terminal, su directorio actual (en el que está situado) es su directorio personal. En el caso del usuario **alumno** su directorio $HOME es ***/home/alumno***

Existe un caso especial y es el del usuario **root**. Su directorio personal no está ubicado, como en el resto de usuarios, en */home*, sino que cuelga directamente de la raíz del sistema de ficheros, es decir, ***/root***

7. Uso de la ayuda en la línea de comandos.

Existen múltiples y variadas formas de obtener ayuda en un sistema Linux. A continuación se describen algunas de ellas:

1) Muchos comandos poseen una opción para mostrar una ayuda breve acerca de su utilización. Esta opción usualmente es **-h**, **--help** ó -?.
   Ejemplo: `$ ls --help`

2) El comando **man** formatea y despliega un manual bastante amplio acerca de comandos, formatos de ficheros de configuración, llamados al sistema, etc. Los manuales están disponibles y pueden instalarse en múltiples idiomas. Estos se dividen internamente en secciones. Un mismo objetivo puede estar representado en varias secciones. De no especificarse ninguna sección a través del primer argumento del comando se tomará la primera donde aparezca.
   Ejemplo: `$ man chmod`

3) Existe un conjunto de comandos integrados (builtin) al bash que no poseen un manual propio. Para ellos se puede emplear el comando help. Si se usa man sólo se mostrará la lista de los comandos integrados. La ayuda que muestra help es un fragmento del manual de bash ($ **man bash**), el cual es muy amplio y resulta incómoda la búsqueda. Algunos de los comandos integrados al bash (órdenes internas) y que veremos más adelante son: *cd*, *fg*, *bg*, *logout*, *exit*, *umask*, *set*, *help*, *source*, *alias*, *echo*, *kill*, *jobs* y *export*.
   Ejemplo: `$ help logout`

4) El programa **info** despliega información acerca de comandos, en ocasiones más amplia que la que brinda *man*. Las páginas de *info* poseen una estructura arbórea (nodos), a través de las cuales se puede navegar con ayuda de comandos especiales.
   Ejemplo: `$ info ln`

5) El comando **whatis** realiza una búsqueda en las secciones del *man* y muestra la descripción abreviada del comando en cada una de las secciones que aparezca.

Este comando trabaja con una base de datos que se actualiza periódicamente y se crea con el comando *makewhatis*.

Ejemplo: `$ whatis shadow`

6) El comando **apropos**, dada una palabra busca en toda la base de datos del *whatis* desplegando todo lo encontrado.

Ejemplo: `$ apropos passwd`

7) Muchos programas se empaquetan con su documentación en diversos formatos. Normalmente estas ayudas se agrupan en el directorio **/usr/share/doc**. También existen los **HOWTOs** (Como lograr...) escritos en muchos idiomas y formatos para varios temas disponibles. Algunos se empaquetan como cualquier otro programa o se localizan en Internet,

8) En **Internet** de forma general existen una gran cantidad de grupos de noticias, listas de discusión, sitios Web y FTP sobre Linux. En muchos de ellos incluso se pueden encontrar páginas de ayuda sobre los comandos en castellano, mientras que en nuestra versión de Linux es posible que nos aparezcan en inglés.

Por defecto, las páginas del manual (**man**) vienen en inglés. Podemos actualizar el manual con las páginas en castellano con los siguientes comandos:

```
$ sudo apt-get install manpages-es manpages-es-extra
$ sudo dpkg-reconfigure locales
```

8. Uso de comodines

Al igual que vimos en Windows, los comodines son caracteres del teclado como el asterisco (*) o el signo de interrogación (?) que se pueden utilizar para representar uno o más caracteres reales al buscar archivos o carpetas. A menudo, los comodines se utilizan en lugar de uno o varios caracteres cuando no se sabe el carácter real o no se desea escribir el nombre completo.

Su funcionamiento es similar al del sistema operativo Windows.

Además, hay otras maneras para refinar la especificación de caracteres en una determinada posición del nombre. Los caracteres que se han de escoger o filtrar se tienen que escribir entre corchetes [ ].

Carácter/es	Sustituyen
?	Cualquier carácter.
*	Cualquier secuencia de caracteres.
[abc]	Uno de los tres caracteres a, b o bien c.
[^123]	Los caracteres que no son 1, 2 o 3 (también es válido el carácter ! en lugar de ^).
[a-h]	Caracteres entre a y h, ambos incluidos.

Podemos combinar más de un carácter comodín. Los caracteres comodín se pueden utilizar en instrucciones que trabajan sobre archivos y directorios, como por ejemplo *ls*.

9. Comandos

Clasificaremos las órdenes en dos tipos: las órdenes internas y las órdenes externas.

Las órdenes internas se encuentran implementadas dentro del mismo shell y, gracias a ello, éste las reconoce inmediatamente y las ejecuta al instante. Las órdenes externas, en cambio, se encuentran implementadas fuera de la shell, ubicadas en algún espacio del disco. Esto hace que el sistema tenga que dedicar un tiempo a la búsqueda de este tipo de órdenes en el disco.

La orden **which** muestra el camino donde se encuentra una determinada orden externa, siempre que se pueda encontrar mediante el camino de búsqueda. Por ejemplo:

```
$ which ls
```

mostrará el directorio del disco donde se encuentra el comando *ls*.

El tiempo dedicado a buscar las órdenes externas en el disco es corto, gracias a un mecanismo que indica al intérprete de comandos cuáles son los lugares en que las ha de buscar. Este mecanismo se denomina camino de búsqueda (*search path*) y es una lista de directorios que el shell utilizará para buscar las órdenes externas. Pensemos que si un

sistema está formado por unos dos mil directorios, el camino de búsqueda puede consistir en una lista de sólo unos siete u ocho, con lo cual el tiempo de búsqueda se reduce notablemente.

Podemos visualizar esta ruta mediante la visualización de la variable $PATH:

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

Son ejemplos de órdenes internas las siguientes: *cd*, *exec*, *arg*, *eval* y *exit*.

Son ejemplos de órdenes externas las siguientes: *ls*, *who*, *date* y *man*.

Hay órdenes que pueden ser internas y tener a su vez un comando externo con el mismo nombre. En tal caso, los internos tienen preferencia. Un ejemplo de esto es el mandato *pwd*.

```
$ pwd
/home/alumno
```

Podemos determinar mediante el mandato ***type*** si *pwd* es un comando interno o externo:

```
$ type -a pwd
pwd es una orden interna del Shell
pwd is /bin/pwd
```

Comprobamos que hay 2 mandatos *pwd*. Uno interno y otro externo ubicado en el directorio */bin*. Podemos ejecutar este último indicando la ruta.

```
$ /bin/pwd
/home/alumno            #En este caso el resultado es idéntico
```

Algunas órdenes se pueden ejecutar escribiendo el nombre y nada más, y dan un resultado, como por ejemplo **ls**. Hay otros comandos que permiten especificar opciones.

Las opciones indican a una orden que tenga un comportamiento diferente al habitual. Añadir una opción a una orden es escribir detrás una letra, habitualmente precedida de un guión (-). Si la opción en lugar de ser una letra es una palabra (más de una letra) debe estar precedida por dos guiones:

```
$ ls -h          #Opción h
```

ó

```
$ ls --help      #Opción help
```

En el ejemplo anterior ambas opciones son equivalentes.

Por ejemplo, el comando **ls** se comporta de manera diferente a la hora de elegir qué archivos debe mostrar cuando se añade detrás la opción **-a**, formando la instrucción:

```
$ ls -a
```

La opción **-a** indica a ls que muestre todos los archivos para mostrar, incluso los ocultos. El mismo carácter de la opción puede tener un significado diferente para otra orden.

Las órdenes también permiten que se les indique sobre qué objetos se debe actuar. Cada objeto especificado llama **parámetro**.

Los parámetros pasados a una orden son las especificaciones del objeto u objetos sobre los que queremos que esta orden actúe.

Un ejemplo de orden con un parámetro es:

```
$ ls /etc
```

Donde el directorio **/etc** es el parámetro de la orden **ls**. Este parámetro indica a *ls* que queremos ver qué hay dentro del directorio **/etc**, en lugar de obtener el resultado por defecto, que consistiría en ver el contenido del directorio actual, sólo introduciendo la orden *ls* seguida de la tecla de intro.

Algunas órdenes requieren parámetros obligatoriamente, y para otros los parámetros son opcionales. Cuando un parámetro es opcional, se suele escribir entre corchetes en la descripción de la sintaxis de la orden. Estos corchetes no se deben introducir: sólo indican que lo que hay dentro se puede escribir o no.

Comprobando la ayuda en línea de una orden, podremos saber si la orden acepta opciones y parámetros, y cuáles de estos son válidos.

Por ejemplo,

```
cd [directorio]
```

indica que la instrucción

```
$ cd
```

es válida, y las instrucciones

```
$ cd /home/jep
```

y

```
$ cd /usr/share
```

también lo son.

10. Comandos para manipular ficheros y directorios.

Comando ls

El comando *ls* permite listar el contenido de un directorio (el equivalente al DIR de Windows).

Sintaxis: ls [opciones] [directorio | fichero]

Algunas opciones:

-l : muestra la salida en formato largo.

-R: lista recursivamente un directorio.

-a : lista además los ficheros ocultos (sus nombres comienzan con punto).

-h : muestra el tamaño de los ficheros en forma más legible (Ej.: 16M, 4k, etc.)

-i : muestra el identificador del i-nodo asociado a cada elemento.

-d : en caso de especificar un directorio, no nos mostrará su contenido, sino las características del mismo (como si de un fichero se tratara).

Ejemplos:

```
$ ls -hl /etc
$ ls -R /usr      #Muestra el contenido de /usr y de todos los
                  directorios que cuelgan del mismo, y así
                  recursivamente hasta el final del árbol.
$ ls -la
```

```
$ ls -lia ..
```

Un grupo de opciones que se suele utilizar bastante es *-la* (`ls -la`).

Es interesante matizar la diferencia entre utilizar el parámetro *–d* cuando indicamos un directorio y no hacerlo:

```
$ ls -l /home/alumno
drwxr-xr-x 2 alumno alumno 0 sep 27 14:59 Documentos
drwxr-xr-x 2 alumno alumno 0 sep 27 14:59 Escritorio
-rw-r--r-- 1 alumno alumno 0 sep 27 18:00 Leeme.txt

$ ls -ld /home/alumno
drwxr-xr-x 2 alumno alumno 0 sep 27 14:59 /home/alumno
```

Comando cd

El comando *cd* se utiliza para cambiar el directorio actual.

Sintaxis: cd [directorio]

Ejemplos:

```
$ cd /tmp
$ cd /etc/default
$ cd        #cambia al directorio home del usuario.
$ cd ~      #cambia al directorio home del usuario.
```

Comando pwd

El comando *pwd* indica el camino (ruta) absoluto del directorio en el cual nos encontramos actualmente.

Ejemplo:

```
$ pwd            # nos devuelve algo como /home/alumno/backup/pruebas
```

Comando mkdir

El comando *mkdir* se utiliza para crear directorios.

Algunas opciones:

-p : crea los directorios intermedios en caso de que no existan.

Ejemplos:

```
$ mkdir entregas
$ mkdir ~/Apuntes
$ mkdir -p docs/linuxdocs/howtos/pdf      # se crean los directorios
                                          intermedios si es necesario
```

Podemos utilizar las llaves {} para indicar varios directorios a crear en el mismo nivel:

```
$ mkdir apuntes/{Sistemas,Inglés,FOL}     # se crean los directorios
                                          Sistemas, Inglés y FOL en
                                          ./apuntes (debe existir)
$ mkdir -p apuntes/{Sistemas,Inglés,FOL}  # Mismo ejemplo que el
                                          anterior pero en este caso
                                          crea ./apuntes en caso de no
                                          existir
```

Comando cp

El comando *cp* permite copiar un fichero en otro, o varios ficheros en un directorio. Si se especifica como destino un nombre de fichero distinto, lo que hace es copiarlo pero con el nuevo nombre.

Sintaxis:

cp [opciones] <origen> <destino>

cp [opciones] <ficheros> <directorio>

Si el *origen* es un único fichero, el *destino* puede ser un directorio o un nuevo nombre de fichero (con su ruta). Si el *origen* son varios ficheros, necesariamente el *destino* ha de ser un directorio.

Algunas opciones:

-R : si se especifica como *origen* un directorio se copia recursivamente, es decir se copia el directorio completo junto con sus subdirectorios.

-i : utiliza una forma interactiva (pregunta antes de sobrescribir el destino).

-p : conserva (preserva) la propiedad (permisos y propietario) del origen si es posible.

-u : realiza una copia actualizada, copiando solamente si el original es más reciente que el destino o éste no existe.

Ejemplos:

```
$ cp /etc/passwd .              # copia un fichero en el directorio
                                actual (.)
$ cp /etc/passwd ./usuarios     # copia el fichero passwd en el
                                directorio actual con el nombre usarios

$ cp -i /usr/bin/*sh /tmp       # copia interactivamente los ficheros
                                terminados en sh de /usr/bin en un
                                directorio llamado /tmp

# cp –u /etc/fstab /root/fstab-backup    #Sólo copia si el origen es
                                    más reciente que el archivo destino
```

Comando mv

El comando *mv* mueve un fichero hacia otro, o varios ficheros hacia un directorio. Este comando permite a su vez **renombrar** ficheros o directorios. Recibe muchas de las opciones de *cp*, siendo su sintaxis igual.

Sintaxis: mv [opciones] <origen> <destino>
mv [opciones] <ficheros> <directorio>

Algunas opciones:

-i : ejecuta el comando de forma interactiva, pregunta ante de sobrescribir el destino si existiera.

-u : actualiza (upgrade) el destino con el fuente solo si este es más reciente.

Ejemplos:

```
$ mv mail.cf mail.cf.old          # renombra un fichero
$ mv -i *.txt /tmp                # mueve ficheros terminados en .txt
                                  al directorio /tmp
$ mv -u program.c src/            # actualiza el fichero destino si
                                  es menos reciente que el fuente
```

Comando rm

El comando *rm* se utiliza para borrar (desenlazar) ficheros

Sintaxis: rm [opciones] <ficheros | directorios>

Algunas opciones:

-r: borra recursivamente un directorio.

-f: borra forzosamente en caso de que no se tenga permiso de escritura en forma directa.

-i: ejecuta el comando de forma interactiva.

Ejemplos:

```
$ rm prueba
$ rm -i bin/*
$ rm -rf temp/
```

Linux no proporciona por defecto ningún tipo de funcionalidad de tipo "papelera de reciclaje" para el comando *rm*. Una vez borrado el archivo ya no se puede recuperar.

Comando touch

Para cada archivo, Linux mantiene tres tipos de marcas temporales:

- Hora de última modificación.
- Hora de último cambio de i-nodo.
- Hora de último acceso.

Mediante el comando *touch* se pueden cambiar estas marcas temporales. Por defecto *touch* asigna la hora actual como hora de modificación y acceso.

Sintaxis: touch [opciones] <ficheros>

En caso de que el fichero especificado no exista, lo crea vacío. Es por ello que es un comando que se suele utilizar cuando queremos crear archivos vacíos, por ejemplo para experimentar con otros comandos.

```
$ touch prueba    #si no existe el fichero prueba, lo crea vacío.
                  Exista o no, deja como último acceso y modificación
                  del fichero la hora actual
```

Comando df

El comando *df* nos informa del uso de disco en los distintos volúmenes. Así podemos saber cuánto espacio tenemos en cada volumen, y cuánto está usado y cuánto libre.

```
# df –h          # El parámetro –h es para que sea legible por
                 "humanos"
```

Comando free

El comando *free* nos informa del consumo de memoria del sistema, tanto en memoria real como en memoria virtual (swap).

```
# free
```

Comando alias

El comando *alias* permite asignarle otros nombres a los comandos. De esta forma se pueden abreviar o llamarlos de forma más nemotécnica.
Sintaxis: alias [nombre[=valor]...]

Sin argumentos muestra todos los alias definidos por el usuario actual. Para deshabilitar un alias se emplea el comando *unalias*.
Ejemplos:

```
$ alias
$ alias l='ls -l --color'
$ alias l
```

Comando type

El comando *type* indica cómo interpreta el *bash* una orden En caso de ser un comando mostraría el camino (*path*) de este, si fuera un alias, cual es el comando original y para una función mostraría su código. Una misma palabra puede constituir un comando, un alias y una función. Bash interpreta con mayor prioridad la función, luego el alias y por último, el comando.
Sintaxis: type [-all] [-path|-type] [palabras]

Opciones:
-all : muestra todas las posibilidades de interpretación.

-path : muestra el camino completo al fichero que representa al comando, en caso de serlo.

-type : muestra la forma de interpretación asumida: alias, keyword, function, builtin y file.

Ejemplos:

```
$ type l
l is aliased to `ls -l --color'

$ type passwd
passwd is /usr/bin/passwd

# type -a rm
rm is aliased to `rm -i'
rm is /bin/rm
```

Comando stat

El comando *stat* muestra las características de un fichero. Por ejemplo: su nombre, permisos, tamaño en bytes, número del i-nodo que lo representa, las fechas de modificación y acceso, el tipo, el dueño, el grupo, etc.

Ejemplos:

```
$ stat /etc/shadow
```

Comando diff

El comando *diff* nos permite comparar ficheros de texto línea a línea, y nos indica en caso de que no sean iguales, en qué líneas cambian. Dado que en Linux se realizan muchas configuraciones del sistema modificando ficheros de texto, es una orden interesante para saber si algo se ha modificado.

```
# diff fichero1 fichero2
```

Comando du

El comando *du* permite conocer la longitud (expresada en kilobytes por defecto) de una jerarquía de ficheros a partir de un directorio.

Sintaxis: du [opciones] [ficheros | directorios]

Algunas opciones:

-h : (human readable view) imprime las unidades de la forma más representativa (Ej. M para megabytes).

-s : sumariza el tamaño de cada fichero/directorio sin profundizar recursivamente en estos últimos.

-c : produce un total cuando se utilizan varios argumentos.

Ejemplos:

```
$ du -h *
$ du -hsc /usr /home
```

11. Comandos para paginar, visualizar y editar ficheros.

Comando cat

El comando *cat* concatena (conCATenate) ficheros y los imprime en la salida estándar. Si no se le pasa ningún argumento lee de la entrada estándar.

Existe también *zcat* que hace lo mismo pero con ficheros comprimidos. Si solo se da el origen a *cat*, utiliza como salida la pantalla. Es decir, `cat nombres` muestra por pantalla el fichero nombres. Si solo se da la salida a *cat* (`cat > fichero`) utiliza como entrada el teclado.

Ejemplo:

```
# cat /etc/passwd                    #Muestra por pantalla el contenido
                                      del fichero /etc/passwd
# cat /etc/passwd /etc/shadow        #Muestra por pantalla el contenido
                                     de los ficheros /etc/passwd y
                                     /etc/shadow
$ cat > fichero
```

Editor vi

El editor vi es el editor estándar de Unix y está orientado a comandos. Existe una versión conocida como **vim** (Vi IMproved) muy poderosa que permite la edición de múltiples ficheros, edición automática para varios lenguajes de programación, ayuda en línea, selección visual, varios niveles de deshacer, etc. Para algunos usuarios, vi resulta incómodo pues para utilizar todas sus potencialidades es necesario conocer muchas combinaciones de teclas, pero si se llega a dominar resulta muy funcional.

Su principal virtud es que encontraremos vi en prácticamente cualquier versión de Unix que usemos, cosa que no se puede decir de otros editores (joe, pico, edit, gedit, nano, emacs, etc.).

El editor *vi* viene instalado, pero es preferible instalar la versión mejorada **vim**:

```
$ sudo apt-get install vim
```

Para editar un fichero ejecutamos *vi* seguido del nombre del fichero (junto con su ruta si no estuviera ubicado en el directorio donde nos encontramos).

```
$ vi nombre_archivo
```

Básicamente *vi* posee dos modos de interacción: el de inserción (edición) y el de comandos. Para pasar al modo comando se pulsa **Esc** y para pasar al de inserción se pulsa **i**.

Algunos comandos útiles en *vi* (pulsando ESC para pasar al modo de comandos, que veremos en la última línea de la pantalla).

dd - borra la línea actual.

D - borra desde la posición actual hasta el final de la línea.

dG - borra hasta el final del fichero.

u - deshace el último comando.

:q - sale del editor (si se han hecho modificaciones y no se ha salvado se genera un error).

:q! - sale sin salvar.

:w - salva.

:wq - salva y sale.

:x - salva y sale.

<n><comando> - ejecuta el comando c n veces.

Editor nano

Nano es el editor de textos para el terminal que viene instalado por defecto en Ubuntu. No es tan potente como Vim o Emacs pero es mucho más fácil de manejar que estos.

Para editar un archivo con *nano* hay que ejecutar el siguiente comando:

```
$ nano nombre_archivo
```

donde *nombre_archivo* será el nombre del archivo que queramos editar. En caso de que el archivo no existiera, se creará un archivo vacío con ese nombre.

Si modificamos el archivo veremos en la parte superior derecha el texto *Modificado*. Para guardar los cambios, pulsaremos la combinación de teclas Control+o. Y para salir, Control+x.

Nano está pensado para ser usado con el teclado, no con el ratón, por lo que tiene asociadas multitud de acciones a combinaciones de teclas. Algunas de las más importantes:

Control+g o F1	Muestra la ayuda
Control+x o F2	Salir sin guardar
Control+o o F3	Guarda el archivo actual
Control+w o F6	Busca una cadena de texto o expresión regular
Control+k o F9	Corta la línea actual
Control+u o F10	Pega la línea cortada
Alt+m	Activa o desactiva el soporte para el ratón
Alt+r	Busca una cadena y la reemplaza por otra

Comandos more y less

Los comandos *more* y *less* paginan (dividen en páginas) uno o varios ficheros y los muestran en la terminal (pantalla). De no indicárseles un fichero, paginan la entrada estándar (que se manda mediante una tubería).

Se diferencian en las facilidades que brindan. Por ejemplo *more* es más restrictivo en cuanto al movimiento dentro del texto, mientras que *less* no limita este aspecto pues acepta el empleo de todas las teclas de movimiento tradicionales. Cuando se alcanza el final del último fichero a paginar, *more* termina automáticamente, no así *less*. También *more* muestra sucesivamente el porcentaje del fichero visto hasta el momento. Tanto *less* como *more* proveen una serie de comandos para moverse con facilidad dentro del texto paginado.

Ejemplos:

```
$ less /etc/passwd
$ more /etc/passwd
$ cat fichero | less
```

Algunas teclas que podemos usar mientras usamos estos programas son:
q - permite interrumpir el proceso y salir.
/p - realiza búsquedas del patrón p dentro del texto. Para repetir la búsqueda del mismo patrón sólo es necesario escribir /.
[n]b - en more permite regresar n páginas (por defecto n es 1).
[n]f - en more se adelantan n páginas y en less, n líneas.

El mandato **man**, para dar formato a su salida, utiliza por defecto el paginador *less*. Existen además los comando *zless* y *zmore* que permiten paginar con *less* y *more* respectivamente, a los ficheros comprimidos sin necesidad de descomprimirlos previamente.

12. Comandos para hacer búsquedas de ficheros y patrones.

Comando grep

El comando *grep* (Globally Regular Expressions Pattern) busca patrones en ficheros. Por defecto devuelve todas las líneas que contienen un patrón (cadena de texto) determinado en uno o varios ficheros. Utilizando las opciones se puede variar mucho

este comportamiento. Si no se le pasa ningún fichero como argumento hace la búsqueda en su entrada estándar.

Sintaxis: grep [opciones] <patrón> [ficheros]

Algunas opciones:

-c devuelve sólo la cantidad de líneas que contienen al patrón.

-i ignora las diferencias entre mayúsculas y minúsculas.

-H imprime además de las líneas, el nombre del fichero donde se encontró el patrón. Es así por defecto cuando se hace la búsqueda en más de un fichero.

-l cuando son múltiples ficheros sólo muestra los nombres de aquellos donde se encontró al patrón y no las líneas correspondientes.

-v devuelve las líneas que no contienen el patrón.

-r busca en un directorio de forma recursiva.

-n imprime el número de cada línea que contiene al patrón.

Ejemplos:

```
$ grep linux /usr/share/doc          # Busca el patrón "linux" en el
                                     fichero /usr/share/doc
$ grep root  /etc/passwd             # Busca el patrón "root" en el
                                     fichero /etc/passwd
# grep -n error /var/log/messages    # Busca el patrón "error" en el
                                     fichero /var/log/messages y muestra
                                     la línea en la que aparece con su
                                     número delante
$ grep -i pepe /etc/passwd           # Ignora diferencias entre mays min
$ grep -c root /etc/group            # Devuelve el número de líneas en
                                     las que aparece el patrón
$ grep -l -r -i img /var/www/html/
$ ls -lia | grep "carta roja"
```

13. Comandos para filtrar ficheros.

Comando sort

El comando *sort* ordena las líneas de un fichero mostrándolas por la salida estándar. De no especificarse un fichero toma la entrada estándar.

Sintaxis: sort [opciones] [fichero]

Algunas opciones:

-r ordena al revés.

-f trata las mayúsculas y minúsculas por igual.

-n ordena de forma numérica, de modo que no es necesario que los números se rellenen con ceros por la izquierda.

-kn ordena por el campo número n, donde n es un número

Ejemplo:

```
$ sort -f /etc/passwd          #Ordena el fichero /etc/passwd
$ sort -n numeros              #Ordena el fichero números por orden
                               numérico y no alfabético
$ sort alumnos -k2             #Ordena el fichero alumnos por el segundo
                               campo (Apellido, por ejemplo)
```

Comando uniq

El comando *uniq* elimina las líneas repetidas de un fichero ordenado, imprimiéndolo por la salida estándar o en otro fichero argumento. De no especificarse un fichero toma la entrada estándar.

Sintaxis: uniq [opciones] [fichero] [salida]

Algunas opciones:

-c utiliza como prefijo en cada línea su número de ocurrencias.

-d sólo imprime las líneas duplicadas.

Ejemplo:

```
$ uniq -d lista.txt
```

Comandos tail y head

Los comandos *tail* y *head* muestran respectivamente el final y el comienzo (10 líneas por defecto) de uno o varios ficheros.

Sintaxis:

tail [opciones] [ficheros]

head [opciones] [ficheros]

Algunas opciones:

-f para el caso de tail se ejecuta de forma sostenida, es decir, se continúa visualizando el final del fichero hasta que se interrumpa el proceso (Ctrl+C).

-q no coloca los encabezamiento con el nombre de los ficheros cuando se indican varios (quiet).

-<n> imprime las n últimas (primeras) líneas en lugar de las diez establecidas por defecto.

Ejemplos:

```
# tail -f /var/log/messages
# tail -20 /var/log/secure
# head -50 /var/spool/mail/pepe
# head -2 -q /etc/*.conf
```

Comando wc

El comando *wc* imprime el número de líneas, palabras y bytes (caracteres) de uno o varios ficheros. Si son varios ficheros hace también un resumen de los totales. De no especificarse un fichero toma la entrada estándar.

Sintaxis: wc [opciones] [ficheros]

Algunas opciones:

-l sólo cuenta líneas.

-c sólo cuenta bytes.

-w sólo cuenta palabras.

Ejemplos:

```
$ wc -l /etc/passwd
$ wc -w /doc/dicciorario.txt
```

Comando cut

El comando *cut* nos permite cortar una línea de texto, para obtener un subconjunto en lugar de la línea completa. Podemos cortar por número de caracteres, por campos, etc.

Sintaxis: cut [opciones] [ficheros]

Algunas opciones:

-c N-M corta desde el carácter número N hasta el carácter número M.

-c N- corta desde el carácter número N hasta el final

-c –N corta desde el principio hasta el carácter número N

-c N,M corta el carácter número N y el carácter número M

-d":" -f1 separa la línea en campos divididos por el carácter : y nos muestra sólo el primer campo.

-d"-" -f3 separa la línea en campos divididos por el carácter - y nos muestra sólo el tercer campo.

Ejemplos:

```
$ cut –c 3-9 /etc/passwd          #Nos muestra los caracteres del 3
                                  al 9 de cada línea del fichero
                                  /etc/passwd
$ cut –c d":" –f4 /etc/passwd     #Nos muestra el cuarto campo
                                  delimitado por el carácter
                                  separador : del fichero /etc/passwd
```

14. Comandos para transformar ficheros.

Comando tr

El mandato *tr* transforma (traduce), comprime y/o borra caracteres de la entrada estándar, escribiendo a la salida estándar.

Si se quiere hacer la transformación sobre un fichero, habrá que redireccionar la entrada estándar con el símbolo < (o hacer que la salida de un mandato *cat* sea la entrada de *tr* mediante una tubería).

Sintaxis:

tr [opciones] conjunto1 [conjunto2]

Algunas de las opciones:

-s reemplaza la secuencia de entrada de un carácter repetido.

-d elimina los caracteres que se especifican en conjunto1.

El mandato *tr* acepta varios atajos como [:alnum:] (todos los números y letras), [:upper:] (todas las letras mayúsculas), [:lower:] (todas las letras minúsculas) y [:digit:] (todos los números).

Ejemplos:

```
$ tr ":" " " < /etc/passwd    #transforma el símbolo : por un espacio
                              en el fichero /etc/passwd
$ tr ":" "\n" < /etc/passwd   #transforma el símbolo : por un salto de
                              línea (intro) en el fichero /etc/passwd
$ cat /etc/passwd | tr ":" " "      #transforma el símbolo : por un
                                    espacio en el fichero /etc/passwd
$ tr "a-z" "A-Z" < nombres    #transforma las minúsculas en mayúsculas
                              en el fichero nombres
$ tr [:lower:] [:upper:] < nombres #transforma   las   minúsculas   en
                                    mayúsculas en el fichero nombres
$ tr  -s " " < nombres        #convierte los espacios seguidos en un
                              solo espacio en el fichero nombres
```

15. Comandos para compactar y agrupar ficheros.

Comandos gzip y gunzip

Los comandos *gzip* y *gunzip* permiten compactar y descompactar (comprimir y descomprimir) respectivamente uno o varios ficheros en formato *gz*.

Sintaxis:

gzip [opciones] <ficheros/directorio>

gunzip [opciones] <ficheros/directorio>

Algunas opciones:

-r : dado un directorio comprime todos los ficheros presentes en él recursivamente.

-1 a -9 : especifica el grado de la compresión (-1 menor y más rápida -9 mayor y más lenta).

-S <sufijo> : permite especificar un sufijo o extensión para el fichero resultado (por defecto es gz).

Ejemplos:

```
$ gzip -9 *             # Comprime todos los ficheros del directorio
                        actual (su extensión cambia a .gz)
$ gunzip big-file.gz    # descomprime el fichero big-file.gz
```

Al comprimir con *gzip* el fichero original se comprime y se le añade la extensión *.gz*, es decir, no se realiza una copia comprimida, sino que se comprime el propio fichero.

Todo lo compactado con *gzip* se puede descompactar con el Winzip de los sistemas Windows. También existen los pares de comandos *zip* y *unzip* (compatibles en ambos sentidos con Winzip), y compress y uncompress.

Comandos bzip2 y bunzip2

Los comandos *bzip2* y *bunzip2* permiten compactar y descompactar (comprimir y descomprimir) respectivamente uno o varios ficheros en formato *bz2*. Es mejor que *gzip* a la hora de comprimir pero consume más recursos y puede costarle más tiempo de proceso en archivos muy grandes.

Sintaxis:

bzip2 [opciones] <ficheros/directorio>

bunzip2 [opciones] <ficheros/directorio>

Algunas opciones:

-1 a -9 : especifica el grado de la compresión (-1 menor y más rápida -9 mayor y más lenta).

Ejemplos:

```
$ bzip2 big-file        # Comprime el fichero big-file y cambia su
                        extensión a .bz2)
$ bunzip2 big-file.bz2  # descomprime el fichero big-file.bz2 y lo deja
                        como big-file
```

Comando tar

El comando *tar* (Tape Archiver) es una herramienta para agrupar varios ficheros aislados o el contenido de un directorio en otro fichero o dispositivo especial. El

comando *tar* no comprime o compacta absolutamente nada, se limita a agrupar varios ficheros en uno solo, sin comprimirlos. Existe una opción -z que automáticamente ejecuta un *gzip* o un *gunzip* sobre el fichero agrupado o la opción -j que hace lo propio pero con *bzip2* o *bunzip2*.

Sintaxis: tar [comandos] [opciones] <fuentes>

Algunos comandos:

c permite crear (tarear), es decir, agrupar ficheros en uno solo.

x permite extraer (destarear), es decir, desagrupar ficheros.

v activa el modo debug, donde se ven todos los mensajes.

t lista el contenido de un fichero resultado de un agrupamiento.

Algunas opciones:

f <fichero> agrupa o desagrupa en o hacia un fichero y no utilizando la salida o entrada estándar como es por defecto. (Ojo, esta opción la usaremos siempre).

z compacta o descompacta el fichero resultante una vez agrupado o desagrupado con *gzip* y *gunzip* respectivamente.

j compacta o descompacta el fichero resultante una vez agrupado o desagrupado con *bzip2* y *bunzip2* respectivamente.

C <dir> Cambia al directorio *dir* antes de realizar las operaciones.

El comando *tar* conserva la estructura jerárquica original de lo agrupado excluyendo el carácter / que representa a la raíz.

Históricamente se incluía un guión (-) delante de los comandos de *tar*. Sin embargo, ya no es necesario y en algunas distribuciones de Linux genera un error.

Ejemplos:

```
$ tar cvzf grande *     # crea un fichero grande donde estarán
                        agrupados todos los ficheros del directorio
                        actual y que además estará comprimido.
$ tar xvzf grande       # desagrupa en el directorio actual el fichero
                        grande y ademas lo descomprime.
# tar cf uconf.tar /etc/passwd /etc/shadow /etc/groups # agrupa en el
                                        fichero uconf.tar los
```

```
                                        ficheros passwd, shadow
                                        y groups
# tar tf uconf.tar      # muestra los ficheros agrupados en uconf.tar
# tar xf uconf.tar      # desagrupa el fichero uconf.tar
```

El comando *tar* se utliza para agrupar y desagrupar ficheros. No comprime. Es decir, el tamaño del fichero agrupado resultante es igual (aproximadamente) a la suma de los tamaños de los ficheros a agrupar. Es cierto que podemos comprimir el fichero resultante mediante *gzip* o *bzip*. Es cierto también que el mismo comando *tar* permite comprimir el fichero resultante (con –z y –j) como si lo hiciéramos también con esos comandos.

Para agrupar ficheros utilizaremos el comando de la siguiente manera:
tar **cvf** nombre_del_fichero_agrupado ficheros_a_agrupar

Para agrupar los ficheros *tema1.pdf*, *tema2.pdf* y *tema3.pdf* en un único fichero llamado *apuntes.tar:*

```
$ tar cvf apuntes.tar tema1.pdf tema2.pdf tema3.pdf
```

Si quisiéramos almacenar el fichero *apuntes.tar* en un directorio diferente al que nos encontramos bastaría con indicarle la ruta al principio del nombre del fichero:

```
$ tar cvf /media/USB/apuntes.tar tema1.pdf tema2.pdf tema3.pdf
```

Del mismo modo, si los ficheros a incluir estuvieran en una ruta distinta al directorio donde nos encontramos, se lo deberíamos indicar también en el nombre del fichero:

```
$ tar cvf /media/USB/apuntes.tar /Apuntes/asir/tema1.pdf
 ~/Apuntes/asir/tema2.pdf ~/Apuntes/asir/tema3.pdf
```

El fichero resultante se llamará *apuntes.tar* y lo almacenará en la ruta */media/USB*

Si quisiéramos comprimir el archivo mediante *gzip* bastaría indicarle en el mismo comando la opción -z

```
$ tar cvzf /media/USB/apuntes.tar.gz /Apuntes/asir/tema1.pdf
 ~/Apuntes/asir/tema2.pdf ~/Apuntes/asir/tema3.pdf
```

OJO, Es importante el orden de las opciones (cvzf, poniendo la c al principio y la f al final).

Si quisiéramos el compresor fuera *bz2* en lugar de *gz*, debemos utilizar la opción –j en lugar de la opción -z

```
$ tar cvjf /media/USB/apuntes.tar.bz2 ~/Apuntes/asir/tema1.pdf
~/Apuntes/asir/tema2.pdf ~/Apuntes/asir/tema3.pdf
```

Para **desagrupar**:

```
$ tar xvf  nombre_del_fichero_agrupado
```

Para desagrupar el fichero llamado *apuntes.tar*:

```
$ tar xvf apuntes.tar
```

Si el fichero hubiera sido además comprimido con *gzip*, debemos utilizar la opción -z

```
$ tar xvzf apuntes.tar.gz
```

Si el fichero hubiera sido comprimido con *bz2*, debemos utilizar la opción -j

```
$ tar xvjf apuntes.tar.bz2
```

Si el fichero se encontrara en otra ruta, bastaría con indicarlo al principio del nombre del fichero:

```
$ tar xvjf /media/USB/apuntes.tar.bz2
```

Y por último, si quisiéramos que los archivos incluidos se desagrupen en un directorio distinto pondríamos con la opción -C (mayúscula) el directorio destino:

```
$ tar xvjf /media/USB/apuntes.tar.bz2 -C ~/Apuntes/Sistemas
```

16. Enlaces a ficheros

En Linux, un enlace es un medio de proporcionar varias identidades a un archivo, similar a los accesos directos de Windows y los alias de Mac OS. En Linux se utilizan los enlaces para:

- Facilitar el acceso a los archivos.

- Dotar de varios nombres a un comando.
- Permitir a los programas buscar los mismos archivos en distintas ubicaciones para acceder a los mismos.

Hay 2 tipos de enlaces:

- De **referencia**, también llamados **rígidos** o **duros**, o **físicos** (*hard links*).
- **Simbólicos** (*soft links*).

Los enlaces de **referencia** se obtienen creando 2 entradas de directorio que apunten al mismo archivo (mismo i-nodo). Ambos nombres de archivo tienen la misma validez e importancia. Es una forma de identificar el mismo contenido con diferentes nombres. Éste enlace no es una copia separada del archivo anterior sino un nombre diferente para exactamente el mismo contenido.

Un enlace de referencia se puede borrar usando el comando *rm* de la misma manera que se borra un archivo, sin embargo **el contenido del i-nodo no se eliminará mientras haya un enlace físico que le haga referencia**.

Los enlaces **simbólicos**, por el contrario, son un tipo especial de archivo. Este archivo es un archivo independiente cuyo contenido apunta al archivo enlazado. Acceder a un enlace simbólico también es como acceder al archivo original. **El enlace simbólico no contiene los datos del archivo**, simplemente apunta al registro del sistema de archivos donde se encuentran los datos. Tiene mucha similitud a un *acceso directo* en Windows o un *alias* en OS X.

Sobre un enlace simbólico también se pueden usar todos los comandos básicos de archivos (*rm*, *mv*, *cp*, etc), sin embargo cuando el archivo original es borrado o movido a una ubicación diferente el enlace dejará de funcionar y se dice que el enlace **está roto**

Del mismo modo, al eliminar el enlace simbólico el fichero original no se ve afectado en absoluto.

Para crear enlaces, tanto de referencia como simbólicos, utilizamos el mandato **ln**.

La sintaxis del comando *ln* es:

ln [opciones] origen [destino]

ln [opciones] origen... directorio

Sus opciones más importantes son las siguientes:

-d Permite al superusuario hacer enlaces rígidos a directorios

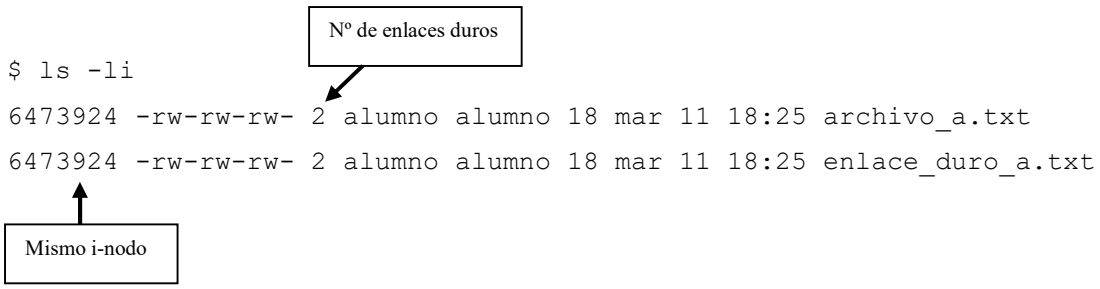
-s Crear enlace simbólico.

Ejemplos:

Para crear un enlace de referencia del archivo *archivo_a.txt* a *enlace_duro_a.txt*, ejecutamos el mandato *ln*.

```
$ ln archivo_a.txt enlace_duro_a.txt
```

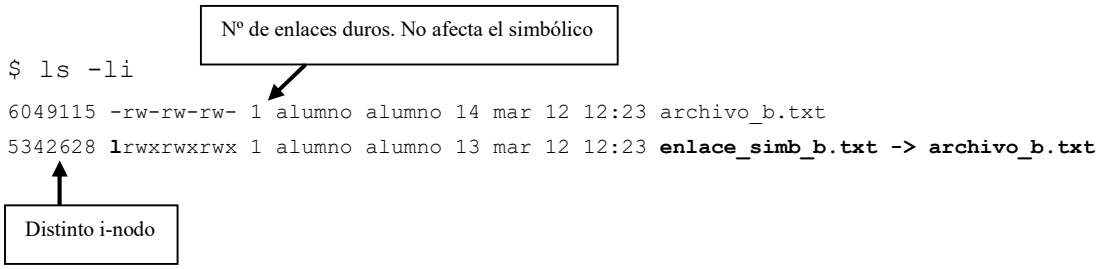
El enlace aparecerá como otro archivo más en el directorio y apuntará al mismo contenido de *archivo_a.txt*. Cualquier cambio que se haga se reflejará de la misma manera tanto para *archivo_a.txt* como para *enlace_duro _a.txt.*



Para crear un enlace simbólico del archivo *archivo_b.txt* a *enlace_simb_b.txt*, ejecutamos el mandato *ln* con el parámetro -s.

```
$ ln –s archivo_b.txt enlace_simb_b.txt
```

En este caso el enlace aparecerá como otro archivo más en el directorio con una indicación de que apunta al fichero *archivo_b.txt*.



Observamos en el fichero enlace que el tipo de archivo (siguiente tema) aparece como una letra **l** y al final de la línea vemos a qué fichero original apunta el enlace mediante una flecha (->).

En la práctica, los enlaces simbólicos son más comunes que los de referencia. Sus desventajas son menores y la capacidad de enlazar entre distintos sistemas de archivos y a directorios puede ser importante. Veremos que los scripts de inicio del sistema utilizan enlaces simbólicos en los directorios de modo de ejecución.

17. Redireccionamiento y Tuberías.

Al igual que el sistema operativo Windows, Linux también permite en su intérprete de comandos el redireccionamiento de la entrada estándar (*stdin*), la salida estándar (*stdout*) y la de error (*stderror*) de sus comandos.

El funcionamiento es el mismo.

>	Redirecciona *stdout*. Es decir, nos permite indicar una salida para la orden que no sea el monitor. Normalmente un fichero.
2>	Redirecciona *stderror*. Es decir, nos permite indicar una salida para los errores de la orden que no sea el monitor.
<	Redirecciona *stdin*. Es decir, nos permite indicar una entrada para la orden que no sea el teclado. Por ejemplo un fichero de respuestas previamente creado.
>>	Igual que >, pero la salida de la orden se añade a la salida que indiquemos. Con > la salida de la orden reescribe la salida que indiquemos.
|	El indicador de tubería. Nos permite indicar que la entrada de una orden será la salida de otra orden. Es decir, el *stdout* de la 1ª orden, será el *stdin* de la 2ª.

Veamos algunos ejemplos de estas redirecciones y tuberías. Si escribimos `ls` veremos como esta orden no nos pide nada (no usa *stdin*) y nos muestra unas líneas (*stdout*) por pantalla. Vamos a cambiarle *stdout*, para ello escribimos:

```
$ ls > salida.txt
```

Veremos como por pantalla no nos sale nada, ya que hemos cambiado *stdout*. Si ahora miramos en el directorio, comprobaremos que se ha creado un fichero *salida.txt* que en su interior (`cat salida.txt`) contiene la salida de la anterior orden ls.

¿Qué ocurriría si escribimos las siguientes órdenes?

```
$ echo "HOLA MUNDO" > fichero1
$ echo "ESTO ES UN EJEMPLO" > fichero1
```

Si ahora miramos el contenido de fichero1 (`cat fichero1`) veremos como sólo contiene la última línea.

```
$ cat fichero1
ESTO ES UN EJEMPLO
```

Esto es así porque > siempre sobrescribe la salida. Para evitar esto escribimos:

```
$ echo "HOLA MUNDO" > fichero1
$ echo "ESTO ES UN EJEMPLO" >> fichero1
$ cat fichero1
HOLA MUNDO
ESTO ES UN EJEMPLO
```

Otro ejemplo:

```
$ ls -l
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_1
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_2
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_3
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_4
$ ls -l > salida.txt                # no muestra nada por pantalla
$ cat salida.txt
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_1
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_2
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_3
-rw-rw-rw- 1 alumno alumno 0 mar 12 12:49 fichero_4
```

Un truco habitual consiste en redireccionar la salida estándar y/o la salida de error hacia el dispositivo (fichero de dispositivo, lo veremos en el tema siguiente) */dev/null*. Este archivo es un dispositivo que no está conectado a nada, se utiliza para deshacerse de datos.

Veamos cómo funciona la redirección de *stdin*. En el siguiente ejemplo se le pasa el fichero */etc/passwd* como entrada del mandato *tr*:

```
$ tr ":" " " < /etc/passwd
```

Para almacenar los posibles mensajes de error de un comando en un fichero redireccionamos la salida de error (*stderr*):

```
$ tr ":" " " < /etc/passwZ 2> error.txt
```

En este caso, al teclear mal el nombre del fichero (*/etc/passwZ* en lugar de */etc/passwd*) nos daría un error que no aparecería por pantalla, sino en un fichero que se crearía a tal efecto denominado *error.txt*.

Por último, usamos la tubería (|) cuando queremos usar la salida de una orden como entrada de la siguiente. Por ejemplo, queremos buscar un patrón en un fichero y que el resultado salga ordenado. Para ello utilizaríamos el mandato **grep** y una vez filtrado, ordenaríamos el resultado.

```
$ grep "García" alumnos | sort
```

En ese ejemplo buscamos en el fichero alumnos el patrón "García" y todas las líneas que contengan dicho patrón aparecerían ordenadas alfabéticamente por el primer carácter.

Algo que **nunca hay que hacer** es poner en el segundo comando un fichero de entrada, como por ejemplo:

```
$ grep "García" alumnos | sort alumnos    #ESTO ESTÁ MAL
```

En este caso el mandato *sort* actuaría sobre la totalidad del fichero *alumnos* y no sobre el filtro de aquellos alumnos que contienen la palabra *García,* que es lo que pretendíamos, no teniendo en cuenta el mandato *grep*.

Ejemplos (hay comandos que aún no hemos visto):

Para obtener la dirección MAC del adaptador de red *enp0s3*:

```
$ ifconfig enp0s3 | grep "direcc" | cut –d " " –f9
```

Para saber cuál es el directorio que más espacio ocupa dentro de /usr:

```
$ du -s /usr/* | sort -g | tail -1
```

Para saber el número de fotos (.jpg) que hay en el directorio /home/alumno:

```
$ find /home/alumno -iname "*.jpg" | wc -l
```

Obtener el directorio $HOME de los usuarios que contengan en su nombre la cadena "Ana" sin tener en cuenta las mayúsculas:

```
$ grep -i ana /etc/passwd | cut -d ":" -f6
```

Obtener el número de usuarios que están identificados en el sistema en este momento:

```
$ who | cut -d " " -f1 | sort | uniq | wc -l
```

Mostrar los nombres de usuarios que tengan la contraseña activa en el sistema:

```
# cat /etc/shadow | cut -d ":" -f1,2 | grep -v \* | grep -v \! | cut -
d ":" -f1
```

Material elaborado a partir de:

- Manual de Ubuntu: Primeros Pasos con Ubuntu 13.10
- LPIC-1 Guía de estudio. Christine Bresnahan, Richard Blum. Anaya Multimedia.
- Administració de la Informació. Implantació de Sistemes Operatius. José Luis Antúnez Reales, Institut Obert Catalunya.
- Apuntes Sistemas Informáticos Monopuesto y Multipuesto. IES Romero Vargas.
- Apuntes Implantación de Sistemas Operativos. José Antonio Carrasco Díaz. IES Romero Vargas.
- Elaboración propia.