

TEMA 4

LA TERMINAL

ÍNDICE

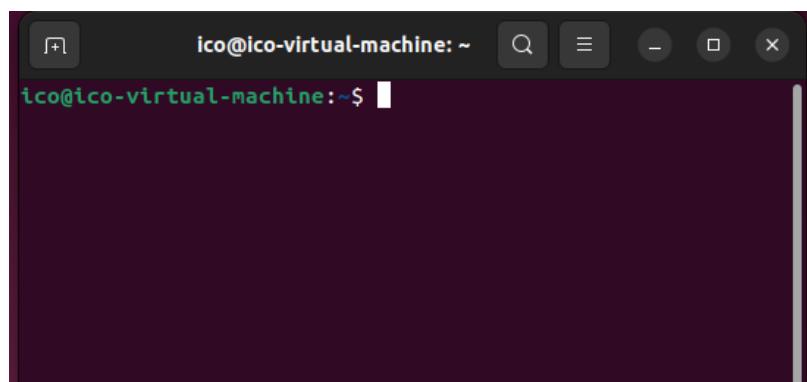
4. LA TERMINAL.....	4
4.1 Estructura del sistema de archivos de Ubuntu	9
4.2 Comandos básicos con directorios.....	10
4.2.1 Comando cd	10
4.2.2 Comando ls.....	13
4.2.3 Comando man	17
4.3 Comandos básicos con ficheros	17
4.3.1 Comando touch.....	17
4.3.2 Comando cat	18
4.3.3 Editores de texto	18
4.3.4 Caracteres comodín	19
4.5 Gestión de ficheros y directorios	21
4.5.1 Gestión de directorios.....	21
4.5.1.1 Comando mkdir	21
4.5.1.2 Comando rm.....	22
4.5.1.3 Comando cp	22
4.5.1.4 Comando mv	23
4.5.2 Gestión de ficheros	24
4.5.2.1 Comando rm.....	24
4.5.2.2 Comando cp	24
4.5.2.3 Comando mv	25
4.6. Administración de usuarios.....	25
4.6.1 Administrador del sistema (root).....	25
4.6.2 Comandos su, whoami y groups	26
4.6.3 Cuentas de usuario.....	26
4.6.3.1 ID de usuarios. UID.....	27
4.6.3.2 Funcionamiento de la autenticación en Linux	27
4.6.3.3 Gestión de usuarios.....	29
4.6.3.3.1 Comando useradd	29
4.6.3.3.2 Comando adduser	29
4.6.3.3.3 Comando passwd	31
4.6.3.3.4 Asignar grupos a los usuarios.....	32

4.6.3.3.5 Comando userdel	33
4.6.3.3.6 Comando usermod	33
4.6.4 Cuentas de grupo	33
4.6.4.1 Uso de los grupos	33
4.6.4.2 ID de grupos. GID.	34
4.6.4.3 Gestión de grupos	34
4.6.4.3.1 Comando groupadd.....	34
4.6.4.3.2 Comando groupdel.....	34
4.6.4.3.3 Comando groupmod	35
4.7. Permisos de archivos y directorios.....	35
4.7.1 Comando chmod	38
4.7.1.1 Formato octal	39
4.7.1.2 Formato simbólico.....	40
4.7.2 Comandos chown y chgrp	41
4.8 CRON	45
4.9 GREP	50

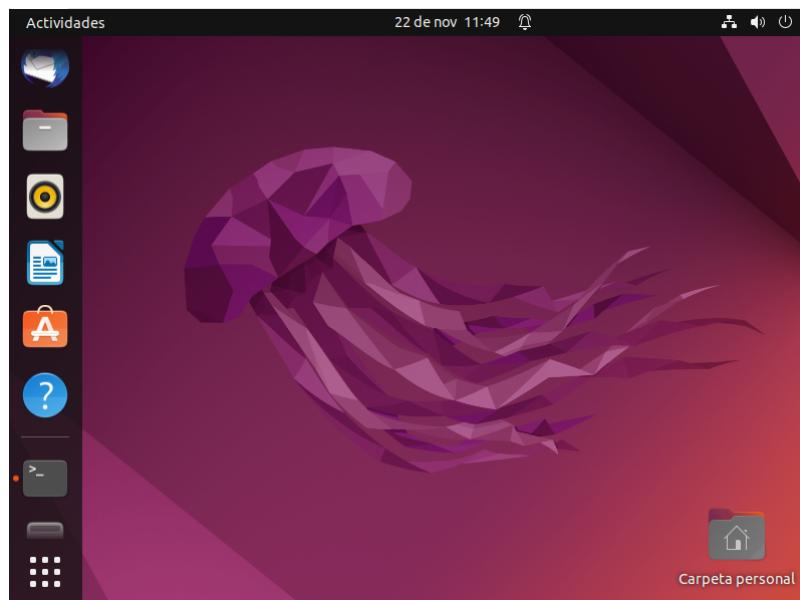
4. LA TERMINAL

A lo largo de este tutorial nos hemos centrado principalmente en la interfaz de usuario del escritorio gráfico. Para comprender completamente la potencia de Ubuntu, es posible que quiera aprender cómo usar el terminal.

Los sistemas operativos como Linux y Windows ofrecen dos tipos de interfaces de usuario, una interfaz gráfica (GUI) orientada a los usuarios normales, es decir, que no realizan tareas de configuración avanzada del sistema operativo y otra basada en línea de comandos que permite a los usuarios avanzados, administradores, realizar una configuración avanzada del sistema operativo. En el caso de Ubuntu, la interfaz de comandos es la aplicación “Terminal”. En la siguiente imagen puedes ver la “Terminal” desde donde escribirás los comandos para configurar Linux.



La interfaz gráfica de Ubuntu es la siguiente:



¿Por qué debería usar el Terminal?

Puede realizar la mayoría de las actividades del día a día sin necesitar abrir nunca el terminal. Sin embargo, el terminal es una herramienta potente e inestimable que se puede usar para realizar multitud de tareas que no se pueden realizar con un GUI. Por ejemplo:

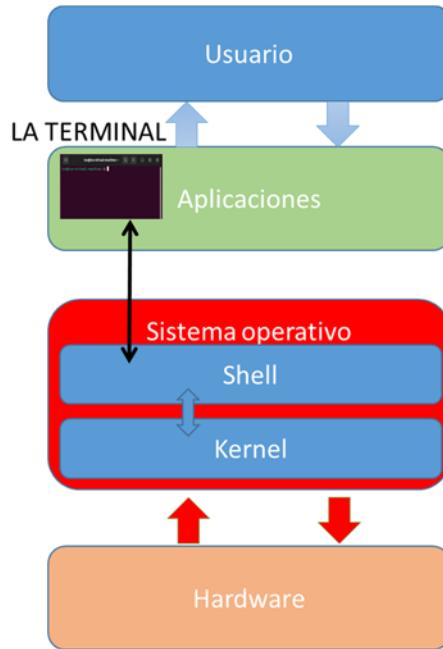
- Diagnosticar problemas que pueden aparecer al usar Ubuntu requiere muchas veces recurrir a la terminal.
- En ocasiones la interfaz de línea de órdenes es una forma más rápida de realizar una tarea. Por ejemplo, a veces es más fácil realizar operaciones en muchos archivos de manera simultánea usando el terminal.
- El aprendizaje de la interfaz de línea de órdenes es el primer paso para adquirir posteriormente conocimientos avanzados de diagnóstico de problemas, administración de sistemas y desarrollo de software. Si está interesado en ser desarrollador o un usuario avanzado de Ubuntu, el conocimiento de la línea de órdenes es esencial.

La Terminal es una aplicación que te permite escribir órdenes (comandos) para realizar ciertas tareas. ¿Pero quién verdaderamente realiza las órdenes? El Shell o intérprete de comandos.

El intérprete de comandos o Shell es la interfaz que permite al usuario interactuar con el sistema. El usuario introduce sus órdenes, el intérprete las procesa y genera la salida correspondiente. El hecho de que el Shell sea el intérprete entre el usuario y el sistema le dota de este nombre, que se podría traducir como “caparazón”.

El intérprete de comandos es tanto una interfaz de ejecución de órdenes y utilidades como un lenguaje de programación que admite crear nuevas órdenes (denominadas “guiones” o “shellscripts”), utilizando combinaciones de comandos y estructuras lógicas de control, que cuentan con características similares a las del sistema y que permiten que los usuarios y grupos de la máquina cuenten con su entorno personalizados.

En el siguiente diagrama se muestra, de modo simplificado, cómo el usuario interacciona con el sistema de acuerdo a esta estructuración de capas:



El kernel es el núcleo del sistema operativo. Es el responsable de la ejecución de tareas y servicios críticos del sistema operativo. Además, es el programa principal que interactúa con todos los componentes del hardware y provee soporte para la ejecución de aplicaciones.

La capa formada por el Shell permitirá la ejecución de aplicaciones, controladas o no por un usuario y además permitirá a éste una interacción directa con el kernel en caso de necesidad.

Tipos de Shell

En Unix existen dos familias principales de intérpretes de comandos:

- Los basados en el intérprete de Bourne: SH, KSH y BASH
- Los basados en el intérprete C: CSH y TCSH

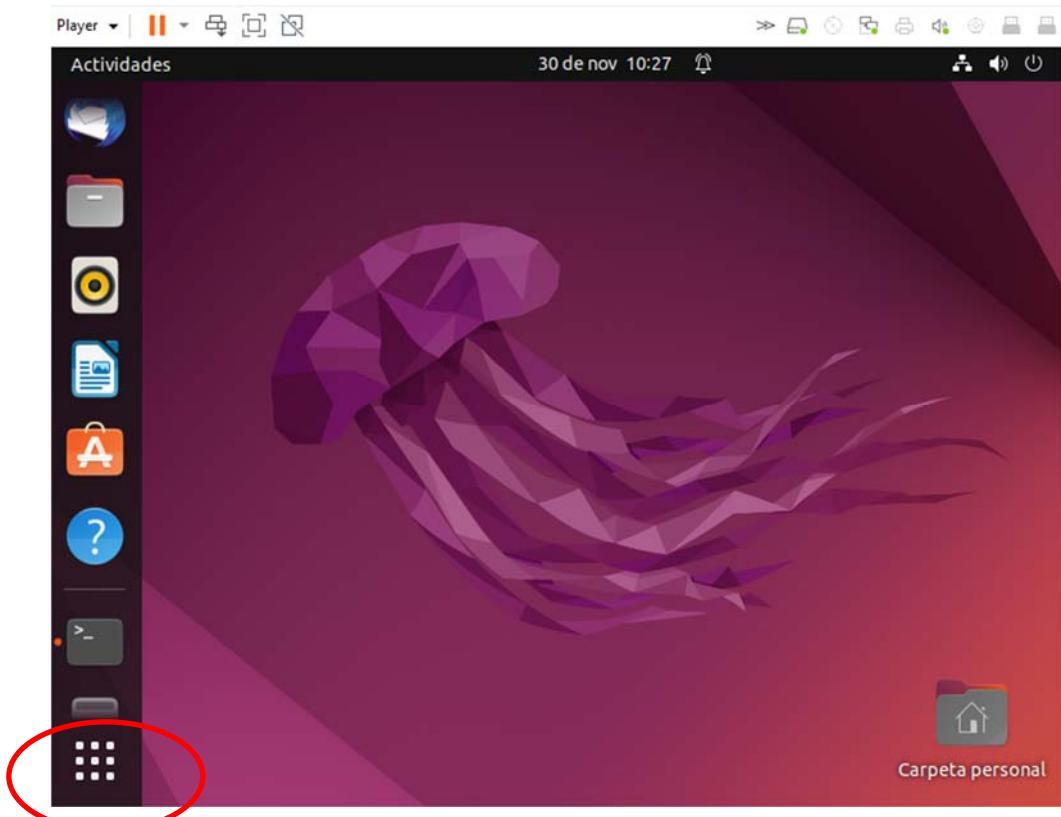
Bash (Bourne Again Shell) es el más moderno y el más utilizado, debido a que se trata del Shell Estándar de GNU/Linux, además de ser muy intuitivo y flexible.

Abrir el Terminal

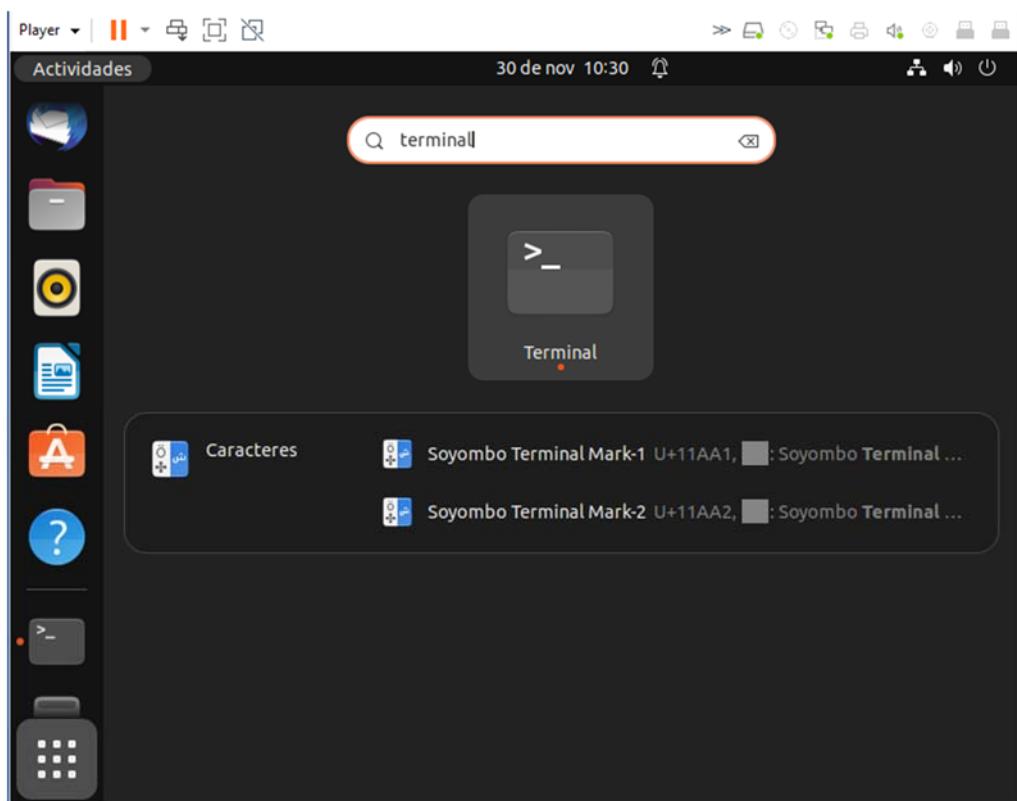
Comenzamos a interactuar con la Terminal. Para ello, abrimos nuestra máquina virtual creada con VMWare Workstation Player.



Para abrir la Terminal, debes pulsar el botón situado en la parte inferior izquierda.

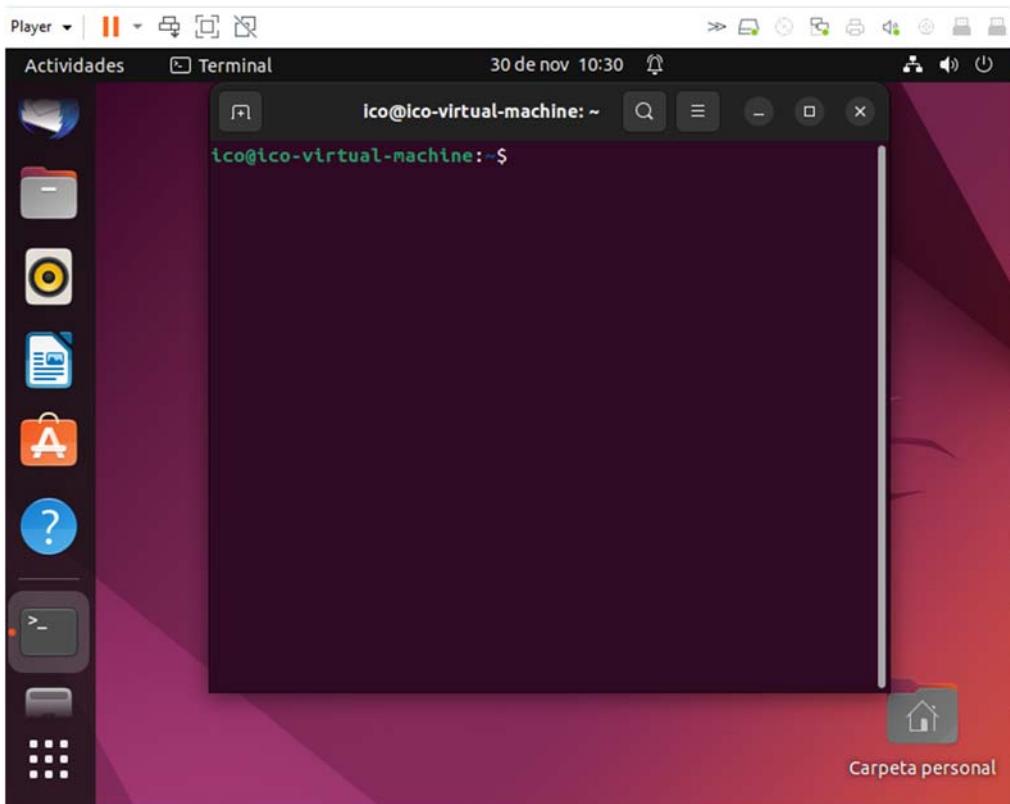


A continuación, escriba terminal en el cuadro de búsqueda.



Y seleccione "Terminal" entre los resultados de la búsqueda.

Como resultado, se mostrará la terminal.



Como puedes ver, la terminal tiene fondo negro y aparece la siguiente información:

nombre de usuario@ nombre del equipo:~\$

En el caso de mi máquina virtual donde el nombre de usuario es “ico” y el nombre de mi equipo es “ico-virtual-machine” aparece:

ico@ico.virtual-machine:~\$

El símbolo ~ indica el directorio actual es su directorio personal.

El carácter \$, algunas veces, puede ser sustituido por # y su significado es:

- \$ indica que el usuario no tiene privilegios particulares
- # indica que el usuario es el administrador root que tiene todos los privilegios

El directorio personal es la carpeta o directorio donde se almacenarán por defecto todos los archivos de ese usuario. El comando que nos permite visualizar mi directorio personal es **pwd**, que son las iniciales de “print working directory”.ico

Escriba en la terminal **pwd** y pulse Intro. El terminal mostrará /home/su-nombre-de-usuario. En mi caso /home/ico.

```
ico@ico-virtual-machine: ~
ico@ico-virtual-machine:~$ pwd
/home/ico
ico@ico-virtual-machine:~$
```

Es importante conocer que todos los comandos tienen el mismo patrón: el nombre del comando seguido de una serie de parámetros que son opcionales.

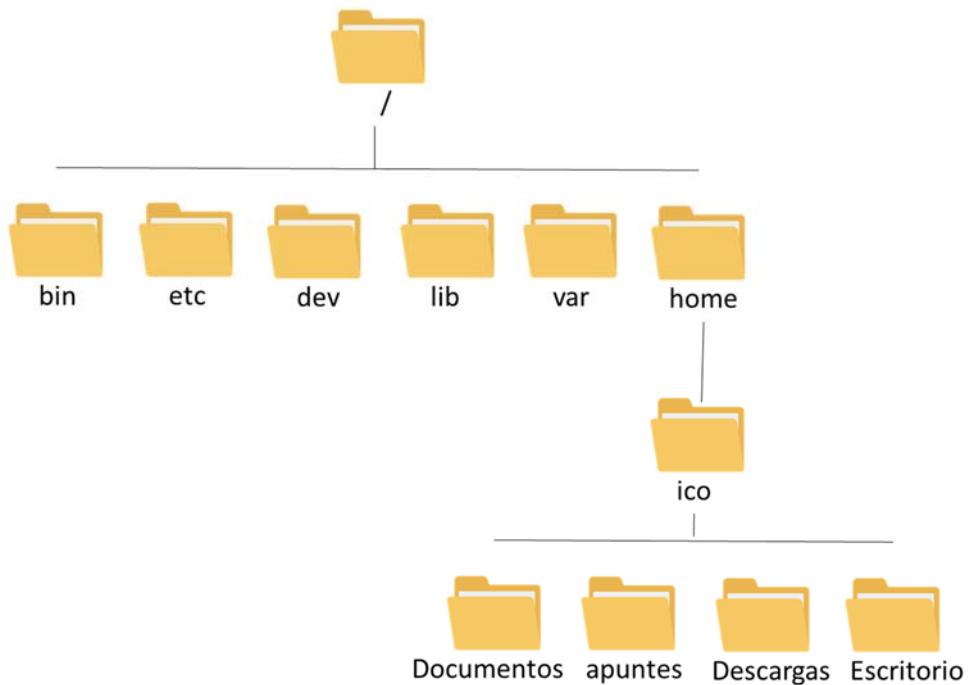


En la imagen podéis ver el comando **ls** que permite listar el contenido del directorio que se indique como parámetro, en este caso /home/ico.

4.1 Estructura del sistema de archivos de Ubuntu

Ubuntu usa el sistema de archivos de Linux, que utiliza carpetas para mantener organizada la información que almacena. Cuando estamos en un entorno gráfico hablamos de carpetas mientras que cuando trabajamos con un entorno de comandos se denominan **directorios**. Así que cuando trabajamos con la Terminal, usaremos el término “directorio”.

Un aspecto importante es que los directorios se estructuran en forma de árbol donde el directorio raíz, que se representa con **/**, es el directorio que contiene al resto como se puede ver en la imagen.



Ubuntu posee una serie de directorios esenciales para su funcionamiento, algunos de ellos son:

bin	Contiene programa esenciales del sistema
dev	Contiene los archivos que permiten interactuar con diferentes dispositivos hardware como el disco duro, la tarjeta gráfica, la impresora, el ratón...
etc	Contiene los archivos de configuración globales del sistema
lib	Almacena las librerías y los módulos del kernel
var	Almacena los archivos de registros (registro de errores) escritos por los programas
home	Contiene los directorios de trabajo de los diferentes usuarios. Cada usuario tiene su propio directorio dentro de home. En la imagen anterior el usuario "ico" tiene una carpeta de trabajo denominada "ico". Por defecto, cuando se da de alta un usuario, se crea su carpeta de trabajo y dentro de ella, una serie de carpetas como "Descargas, "Documentos" y "Escritorio".

Cada directorio tiene una **ruta completa o absoluta** que indica dónde se encuentra dentro del árbol de archivos. La ruta comienza por / (el directorio raíz) y se añaden los directorios por los que hay que pasar para llegar al directorio objetivo.

Por ejemplo, la ubicación del directorio "apuntes" del usuario "ico" mediante rutas absolutas es:

/home/ico/apuntes

4.2 Comandos básicos con directorios

Comencemos a trabajar con los comandos básicos que nos permiten movernos por el sistema y que servirán de apoyo para el uso de otros comandos.

4.2.1 Comando cd

El comando `cd` (es el acrónimo de "change directory") permite movernos por la estructura de directorios que se hemos visto en el apartado anterior.

Por ejemplo, supongamos que queremos desplazarnos al directorio donde se encuentran los directorios de usuario ("~/home"). Simplemente habrá que escribir el `cd` junto con la ruta hacia el directorio destino.

COMANDO RUTA AL DIRECTORIO DESTINO

cd

/home

```
ICO@ICO-VIRTUAL-MACHINE: /home
ICO@ICO-VIRTUAL-MACHINE:~/ $ cd /home
ICO@ICO-VIRTUAL-MACHINE:/home$
```

Fíjese que después de ejecutar el comando el prompt del sistema (en este caso, ico#ico-virtual-machine:/\$) ha cambiado a ico#ico-virtual-machine:/home\$, donde /home indica el directorio donde se encuentra ahora.

¿Qué comando utilizarías para moverte al directorio etc? Localízalo en la estructura de directorios y escribe el comando.

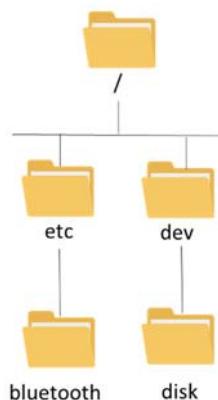
```
ICO@ICO-VIRTUAL-MACHINE: /etc
ICO@ICO-VIRTUAL-MACHINE:~/ $ cd /etc
ICO@ICO-VIRTUAL-MACHINE:/etc$
```

Para indicar el directorio al que quieras ir hay que detallar dónde se encuentra ese directorio, es decir, el camino desde la raíz de la estructura de directorios (/) hasta el directorio destino, a este “camino” se le denomina **ruta**.

Las rutas podrán ser **absolutas o relativas**.

- Las rutas absolutas comienzan desde la raíz de la estructura de directorios (es decir, desde "/") y de ahí hacia abajo hasta llegar al directorio destino. Por ejemplo,
- Las rutas relativas comienzan desde el directorio actual y desde ahí hasta el directorio destino.

Dada la siguiente estructura de directorios, vamos a realizar los siguientes ejercicios:



- a) Suponiendo que me encuentro en el directorio dev, indicar el comando para desplazarte hasta el directorio disk con rutas absolutas y con rutas relativas.

Con rutas absolutas el comando sería:

```
cd /dev/disk
```

indicando el camino hasta el directorio destino desde la raíz (“/”)

```
ico@ico-virtual-machine: /dev/disk
ico@ico-virtual-machine:/dev$ cd /dev/disk
ico@ico-virtual-machine:/dev/disk$
```

Con rutas relativas el comando sería:

```
cd disk
```

partimos del directorio donde nos encontramos y añadimos la ruta necesaria para llegar hasta el directorio destino.

```
ico@ico-virtual-machine: /dev/disk
ico@ico-virtual-machine:/dev$ cd disk
ico@ico-virtual-machine:/dev/disk$
```

Los administradores usan estas rutas relativas ya que permiten acortar los comandos a escribir.

- b) Suponiendo que me encuentro en el directorio disk, indicar el comando para desplazarte hasta el directorio dev usando rutas absolutas y rutas relativas.

Con rutas absolutas el comando sería:

```
cd /dev
```

```
ico@ico-virtual-machine:/dev/disk$ cd /dev
ico@ico-virtual-machine:/dev$
```

Con rutas relativas el comando sería:

```
cd ..
```

```
ico@ico-virtual-machine:/dev/disk$ cd ..
ico@ico-virtual-machine:/dev$
```

Los dos puntos nos permiten dar un paso atrás en la estructura de directorios. Es decir, nos permite ir al directorio padre.

¿Y si queremos ir dos pasos hacia atrás? Por ejemplo, si estamos en disk y queremos ir al directorio raíz. El comando sería:

```
cd ../../
```

```
ico@ico-virtual-machine:/dev/disk$ cd ../../
ico@ico-virtual-machine:/$
```

- c) Suponiendo que me encuentro en el directorio disk, indicar el comando para desplazarte hasta el directorio bluetooth usando rutas absolutas y rutas relativas.

Con rutas absolutas el comando sería:

```
cd /etc/bluetooth
```

```
ico@ico-virtual-machine:/dev/disk$ cd /etc/bluetooth  
ico@ico-virtual-machine:/etc/bluetooth$ █
```

Con rutas relativas el comando sería:

```
cd ../../etc/bluetooth
```

Es decir, voy hasta la raíz con .. y luego desciendo por la estructura hasta bluetooth.

```
ico@ico-virtual-machine:/dev/disk$ cd ../../etc/bluetooth  
ico@ico-virtual-machine:/etc/bluetooth$ █
```

Prueba a realizar más desplazamientos dentro de esta estructura.

Existen una serie de parámetros que se pueden añadir a cd para que el desplazamiento sea más sencillo, por ejemplo...

```
cd -
```

Te permite volver al directorio desde dónde vienes.

```
ico@ico-virtual-machine:/dev/disk$ cd ../../etc/bluetooth  
ico@ico-virtual-machine:/etc/bluetooth$ cd -  
/dev/disk  
ico@ico-virtual-machine:/dev/disk$ █
```

```
cd ~
```

Vuelves a tu directorio personal, en este caso, /home/ico. Para escribir el símbolo ~ en el terminal hay que pulsar a la vez Alt Gr+4

```
ico@ico-virtual-machine:/dev/disk$ cd ~  
ico@ico-virtual-machine:~$ pwd  
/home/ico  
ico@ico-virtual-machine:~$
```

4.2.2 Comando ls

COMANDO	PARÁMETROS
ls	/home/ico

Este comando permite listar el contenido del directorio que se indique como parámetro, en este caso /home/ico. Si no se pone parámetros, se muestra el contenido del directorio actual.

En azul se mostrarán los nombres de los directorios mientras que en blanco los nombres de los ficheros que se encuentran dentro de ese directorio.

```
ico@ico-virtual-machine:/$ ls /home/ico
bar.txt      dir2          fichero.txt  prueba.txt  vengadores
capítulo3   Documentos    Imágenes     Público      Vídeos
Descargas    ejemplo.txt   Música       secret.txt
dir1         Escritorio   Plantillas  snap
```

Las rutas, tanto las absolutas como las relativas, se pueden utilizar en la mayoría de comandos. Por ejemplo, con `ls` el directorio que se quiere listar se puede indicar con rutas absolutas y relativas.

- Suponiendo que me encuentro en mi directorio personal, mostrar el contenido del directorio padre usando rutas relativas.

```
ico@ico-virtual-machine:/$ cd /home/ico
ico@ico-virtual-machine:~$ ls ..
ana edurne ico juan mickey prueba
ico@ico-virtual-machine:~$
```

- Suponiendo que me encuentro en el directorio `bluetooth`, indicar el comando para mostrar el contenido del directorio `disk` con rutas relativas.

```
ico@ico-virtual-machine:/etc/bluetooth$ ls ../../dev/disk
by-id  by-partlabel  by-partuuid  by-path  by-uuid
ico@ico-virtual-machine:/etc/bluetooth$
```

Si se quiere un listado más completo se disponen de las siguientes opciones:

Opción	Descripción
<code>-a</code>	Muestra todos los archivos, incluyendo los ocultos (cuyo nombre comienza por un punto). También mostrará en el listado los caracteres “..” y “.” que hacen referencia al directorio padre y directorio actual respectivamente.

```

ico@ico-virtual-machine:/home$ cd ico
ico@ico-virtual-machine:~$ ls
bar.txt      dir2          fichero.txt  prueba.txt  vengadores
capítulo3   Documentos    Imágenes     Público      Vídeos
Descargas    ejemplo.txt  Música       secret.txt
dir1         Escritorio   Plantillas  snap
ico@ico-virtual-machine:~$ ls -a
.           dir2          prueba.txt
..          Documentos    Público
bar.txt     ejemplo.txt  secret.txt
.bash_history Escritorio   .selected_editor
.bash_logout  fichero.txt  snap
.bashrc      .gnupg        .ssh
.cache       Imágenes     .sudo_as_admin_successful
capítulo3   .local        vengadores
.config      Música       Vídeos
Descargas    Plantillas
dir1        .profile
ico@ico-virtual-machine:~$

```

-l

Muestra un listado detallado, con los permisos, y detalles del propietario, grupo, la última fecha de modificación de cada archivo, su tamaño, ...

```

ico@ico-virtual-machine:~$ ls -l
total 60
-rw-rw-r-- 1 ico      ico      25 dic 12 08:18 bar.txt
drwxrwxr-x 4 ico      ico      4096 oct 10 14:19 capítulo3
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Descargas
drwxrwxrwx 2 ico      ico      4096 oct 18 07:59 dir1
drwxrwxrwx 2 ico      ico      4096 oct 18 07:59 dir2
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Documentos
-rw-rwx--- 1 mickey  Disney    0 oct 18 07:50 ejemplo.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Escritorio
-rw-rw-r-- 1 ico      ico      0 oct 18 07:45 fichero.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Imágenes
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Música
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Plantillas
-rw-rw-r-- 1 ico      ico      13 oct 10 14:20 prueba.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Público
-r----- 1 ico      ico      0 oct 18 08:01 secret.txt
drwx----- 3 ico      ico      4096 jun  6 2022 snap
drwxrw-rw- 2 mickey  marvel   4096 oct 18 07:57 vengadores
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Vídeos
ico@ico-virtual-machine:~$ 

```

-F

Muestra un listado detallado indicando el tipo de cada uno de los archivos. Pone / detrás de los nombres que son directorios.

```

ico@ico-virtual-machine:$ ls -F /home/ico
bar.txt      dir2/          fichero.txt  prueba.txt  vengadores/
capítulo3/  Documentos/    Imágenes/    Público/    Vídeos/
Descargas/   ejemplo.txt*  Música/     secret.txt
dir1/        Escritorio/   Plantillas/  snap/
ico@ico-virtual-machine:$ 

```

-R	Muestra recursivamente todo el contenido de los directorios incluidos en el actual. <pre>ico@ico-virtual-machine:\$ ls -R /home/ico /home/ico: bar.txt dir2 fichero.txt prueba.txt vengadores capítulo3 Documentos Imágenes Público Vídeos Descargas ejemplo.txt Música secret.txt dir1 Escritorio Plantillas snap /home/ico/capítulo3: color forma /home/ico/capítulo3/color: frio /home/ico/capítulo3/color/frio: /home/ico/capítulo3/forma: angulo curva /home/ico/capítulo3/forma/angulo: /home/ico/capítulo3/forma/curva: /home/ico/Descargas:</pre>
-S	Muestra el listado ordenado por el tamaño de los archivos. <pre>ico@ico-virtual-machine:\$ ls -ls /home/ico total 60 drwxrwxr-x 4 ico ico 4096 oct 10 14:19 capítulo3 drwxr-xr-x 2 ico ico 4096 jun 6 2022 Descargas drwxrwxrwx 2 ico ico 4096 oct 18 07:59 dir1 drwxrwxrwx 2 ico ico 4096 oct 18 07:59 dir2 drwxr-xr-x 2 ico ico 4096 jun 6 2022 Documentos drwxr-xr-x 2 ico ico 4096 jun 6 2022 Escritorio drwxr-xr-x 2 ico ico 4096 jun 6 2022 Imágenes drwxr-xr-x 2 ico ico 4096 jun 6 2022 Música drwxr-xr-x 2 ico ico 4096 jun 6 2022 Plantillas drwxr-xr-x 2 ico ico 4096 jun 6 2022 Público drwx----- 3 ico ico 4096 jun 6 2022 snap drwxrw-rw- 2 mickey marvel 4096 oct 18 07:57 vengadores drwxr-xr-x 2 ico ico 4096 jun 6 2022 Vídeos -rw-rw-r-- 1 ico ico 25 dic 12 08:18 bar.txt -rw-rw-r-- 1 ico ico 13 oct 10 14:20 prueba.txt -rw-rwx--- 1 mickey Disney 0 oct 18 07:50 ejemplo.txt -rw-rw-r-- 1 ico ico 0 oct 18 07:45 fichero.txt -r----- 1 ico ico 0 oct 18 08:01 secret.txt ico@ico-virtual-machine:\$</pre>

4.2.3 Comando man

La herramienta `man`, es la que nos permite visualizar las “páginas del manual” de Linux correspondiente al comando que se indique como parámetro. Por ejemplo, si queremos obtener información sobre el comando `ls` solo tenemos que ejecutar el comando `man ls` , obteniendo el siguiente resultado.

```
LS(1) User Commands LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
Manual page ls(1) line 1 (press h for help or q to quit)
```

Nos proporciona ayuda sobre la sintaxis del comando, así como el uso de las distintas opciones que posee. Para avanzar por las páginas del manual deberás utilizar la barra espaciadora. Si quieres ir línea a línea puedes utilizar las teclas de dirección del teclado para avanzar o retroceder. Para salir del manual sólo tienes que pulsar la tecla “q”.

→ Hacer Actividad 4.1

4.3 Comandos básicos con ficheros

4.3.1 Comando touch

El comando `touch` permite crear un fichero vacío de una manera rápida.

COMANDO	RUTA AL DIRECTORIO +NOMBRE DEL FICHERO
<code>touch</code>	<code>/home/ico/mifichero.txt</code>

Touch tiene como parámetro el nombre del fichero que se quiere crear junto con el directorio donde lo quieres almacenar. En este caso, se crearía un fichero denominado “mifichero.txt” que se almacenará en `/home/ico`. En caso de no especificar el directorio, se creará en el directorio actual. Por ejemplo, `touch mifichero.txt`

```
ico@ico-virtual-machine:~$ touch mifichero.txt
ico@ico-virtual-machine:~$ ls
bar.txt  dir2      fichero.txt  Plantillas  snap
capítulo3 Documentos  Imágenes  prueba.txt  Vengadores
Descargas ejemplo.txt  mifichero.txt  Público    Videos
dir1     Escritorio  Música    secret.txt
ico@ico-virtual-machine:~$
```

4.3.2 Comando cat

Este comando permite visualizar el contenido de un fichero que se indica como parámetro.

COMANDO	RUTA AL DIRECTORIO +NOMBRE DEL FICHERO
cat	/home/ico/mifichero.txt

Si no se especifica la ruta del directorio donde se encuentra el fichero, tomará por defecto el directorio actual.

```
ico@ico-virtual-machine:~$ cat mifichero.txt
ico@ico-virtual-machine:~$
```

Si cogemos el fichero del apartado anterior, veremos que está vacío. Para añadirle contenido, necesitaremos usar un editor de texto, vayamos al siguiente apartado.

4.3.3 Editores de texto

Un editor de textos permite crear sencillos archivos de texto (similar al bloc de notas de Windows), y editar un fichero de texto que contenga la configuración de una aplicación.

Son muchos los editores de texto que podemos encontrar en Linux, y cada uno de ellos posee unas características específicas que lo hacen idóneo para determinado grupo de usuarios.

En esta asignatura se usará uno de los más sencillo llamado “nano”. El comando a utilizar para editar un fichero con nano es:

COMANDO	RUTA AL DIRECTORIO +NOMBRE DEL FICHERO
nano	/home/ico/mifichero.txt

La pantalla del editor que aparecerá será la siguiente:

The screenshot shows a terminal window titled "GNU nano 6.2" with the subtitle "Nuevo búfer". The main area is a dark grey rectangle. At the bottom, there is a status bar containing the following text:

```
[ Bienvenido a nano. Para obtener ayuda básica, pulse Ctrl+G. ]
^G Ayuda      ^O Guardar     ^W Buscar     ^K Cortar      ^T Ejecutar
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar      ^J Justificar
```

En la parte baja, se muestra un resumen de las operaciones más comunes que pueden realizarse (y cuál es su codificación).

Algunas opciones interesantes son:

Tecla	Función
Control + g	Muestra la ayuda de nano
Control + x	Salir sin guardar los cambios
Control + o	Guardar los cambios realizados en el fichero

4.3.4 Caracteres comodín

A menudo, se necesita llevar a cabo acciones sobre muchos archivos o directorios al mismo tiempo. Por ejemplo, supongamos que he creado 3 ficheros de texto, fich1.txt, fich2.txt y fich3.txt y quiero ver su contenido de una manera rápida, se podría hacer de la siguiente manera:

```
ico@ico-virtual-machine:~$ nano fich1.txt
ico@ico-virtual-machine:~$ nano fich2.txt
ico@ico-virtual-machine:~$ nano fich3.txt
ico@ico-virtual-machine:~$ cat fich1.txt fich2.txt fich3.txt
Mi fichero 1
Mi fichero 2
Mi fichero 3
ico@ico-virtual-machine:~$
```

Pero existe una manera más sencilla usando caracteres comodín basándonos en el patrón que comparten los tres ficheros, en este caso, los tres ficheros comienza por fich.

Podemos usar el **comodín “*”** que representa a cualquier número de caracteres incluso la cadena vacía. En el siguiente comando, se muestra el contenido de todos los ficheros cuyo nombre comienza por “fich” y les sigue cualquier combinación de caracteres.

```
ico@ico-virtual-machine:~$ cat fich*
Mi fichero 1
Mi fichero 2
Mi fichero 3
ico@ico-virtual-machine:~$
```

Este carácter se puede colocar en cualquier lugar. Por ejemplo, para mostrar todos los ficheros que empiezan por la letra “f” y terminan por “t” dentro del directorio /home/ico:

```
ico@ico-virtual-machine:~/ls /home/ico/f*t
/home/ico/fich1.txt /home/ico/fich2.txt /home/ico/fich3.txt
ico@ico-virtual-machine:~$
```

Otro comodín muy utilizado es “?” que representa a un solo carácter sin importar cuál. Por ejemplo, se quiere mostrar el contenido de los ficheros que comienzan por “fich”, a continuación, tienen un carácter, no me importa cuál, y a continuación terminan con “.txt”. El comando sería:

```
ico@ico-virtual-machine:~$ cat fich?.txt
Mi fichero 1
Mi fichero 2
Mi fichero 3
ico@ico-virtual-machine:~$
```

También es posible combinar varios caracteres comodín en un mismo comando. Supongamos que se quiere mostrar todos los ficheros del directorio /home/ico que comienzan por “fich”, a continuación, tienen un carácter, no me importa cuál, y a continuación, un “.” y, por último, les sigue cualquier combinación de caracteres. El comando sería:

```
ico@ico-virtual-machine:~/ls /home/ico/fich?.*
/home/ico/fich1.txt /home/ico/fich2.txt /home/ico/fich3.txt
ico@ico-virtual-machine:~$
```

Otra de las funcionalidades que ofrece la terminal es la posibilidad de especificar **el rango de caracteres** que queremos que aparezca. Por ejemplo, para mostrar el contenido de los ficheros que comienzan por “fich” seguido de un número del uno al tres y que termina por “.txt” se puede usar el siguiente comando:

```
ico@ico-virtual-machine:~$ cat fich[1-3].txt
Mi fichero 1
Mi fichero 2
Mi fichero 3
ico@ico-virtual-machine:~$
```

Los corchetes se usan de manera parecida al comodín “?” aunque, a diferencia de éste, permiten especificar un poco más. Por ejemplo, [hjko] significa cualquiera de los caracteres h, j, k u o. [Dd]ocumento es un patrón que encaja tanto con Documento como con documento. [a-z]* representa cualquier cadena de caracteres que comienza con una letra minúsculas.

Carácter comodín	Descripción
?	Representa cualquier carácter, pero sólo uno
*	Representa cualquier secuencia de cero o más caracteres
[]	Representa un rango de caracteres. Por ejemplo [a-z] Con el símbolo ! se indica la negación
{ }	Permite indicar palabras a escoger. Por ejemplo, ls {con, conf}* muestra los ficheros que comienzan con "con" o "conf"

Clases con nombres

Para facilitar la especificación de rangos de letras y números existen varias clases con nombre. Las clases con nombre se abren con [: y se cierran con :] y pueden ser usadas dentro de expresiones en corchetes.

Clase	Función
[:alnum:]	Representa a todos los caracteres alfanuméricos
[:digit:]	Representa a los dígitos de 0 hasta 9
[:upper:]	Representa a las letras en mayúscula
[:lower:]	Representa a las letras en minúscula
[:blank:]	Representa a los caracteres de espacio y tabulación
^ (negación)	^ niega el sentido de los caracteres restantes para que la coincidencia ocurra solamente si un carácter no está en la clase.

➔ Hacer Actividad 4.2

4.5 Gestión de ficheros y directorios

4.5.1 Gestión de directorios

4.5.1.1 Comando mkdir

Este comando nos permite crear un nuevo directorio.



```
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich3.txt prueba.txt Vídeos
capítulo3 ejemplo.txt Imágenes Público
Descargas Escritorio mifichero.txt secret.txt
dir1 fich1.txt Música snap
dir2 fich2.txt Plantillas vengadores
ico@ico-virtual-machine:~$ mkdir midirectorio
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich3.txt Plantillas vengadores
capítulo3 ejemplo.txt Imágenes prueba.txt Vídeos
Descargas Escritorio midirectorio Público
dir1 fich1.txt mifichero.txt secret.txt
dir2 fich2.txt Música snap
ico@ico-virtual-machine:~$
```

4.5.1.2 Comando rm

Este comando permite borrar un directorio.



```
ico@ico-virtual-machine:/$ rm -R /home/ico/midirectorio
ico@ico-virtual-machine:/$ ls /home/ico
bar.txt Documentos fich3.txt prueba.txt Vídeos
capítulo3 ejemplo.txt Imágenes Público
Descargas Escritorio mifichero.txt secret.txt
dir1 fich1.txt Música snap
dir2 fich2.txt Plantillas vengadores
ico@ico-virtual-machine:/$
```

La opción “-R” indica que borre recursivamente todos los archivos dentro del directorio.

4.5.1.3 Comando cp

Este comando permite copiar el contenido de un directorio en otro.



Se suele poner las siguientes opciones:

- “-r” para indicar que recorra recursivamente su contenido.
- “-p”, que se utiliza para indicar que, una vez realizada la copia, los ficheros mantendrán los mismos permisos que el original
- “-f”, para forzar la copia

En el siguiente ejemplo queremos copiar el contenido del directorio Descargas en el directorio Documentos:

```
ico@ico-virtual-machine:~$ ls Descargas
midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$ cp -prf Descargas/* Documentos
ico@ico-virtual-machine:~$ ls Documentos
midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$
```

4.5.1.4 Comando mv

Este comando permite mover un directorio con su contenido a otro directorio.

COMANDO	DIRECTORIO A MOVER	DIRECTORIO DESTINO
mv	directorio_origen	directorio_dest

En el siguiente ejemplo queremos mover el directorio dir1 al directorio Documentos:

```
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich3.txt prueba.txt Vídeos
capítulo3 ejemplo.txt Imágenes Público
Descargas Escritorio mifichero.txt secret.txt
dir1 fich1.txt Música snap
dir2 fich2.txt Plantillas Vengadores
ico@ico-virtual-machine:~$ mv dir1 Documentos
ico@ico-virtual-machine:~$ ls Documentos
dir1 midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$ ls Documentos/dir1
fichero.txt mi
ico@ico-virtual-machine:~$
```

El comando mv nos permite mover un directorio y cambiarle de nombre a la vez. En el siguiente ejemplo queremos mover el directorio dir1 al directorio Documentos y cambiarle nombre a Directorio1.

```
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich3.txt prueba.txt Vídeos
capítulo3 ejemplo.txt Imágenes Público
Descargas Escritorio mifichero.txt secret.txt
dir1 fich1.txt Música snap
dir2 fich2.txt Plantillas Vengadores
ico@ico-virtual-machine:~$ ls Documentos
midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$ mv dir1 Documentos/Directorio1
ico@ico-virtual-machine:~$ ls Documentos
Directorio1 midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$
```

Para modificar el nombre de un directorio sin moverlo el comando es mv.

COMANDO	NOMBRE ACTUAL DEL DIRECTORIO	NOMBRE NUEVO DEL DIRECTORIO
mv	directorio_origen	directorio_dest

En el siguiente ejemplo queremos mover el directorio dir2 al directorio Directorio2:

```

ico@ico-virtual-machine:~$ ls Documentos
Directorio1 midescarga1.txt midescarga2.txt midescarga3.txt
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich2.txt Música secret.txt
capítulo3 ejemplo.txt fich3.txt Plantillas snap
Descargas Escritorio Imágenes prueba.txt vengadores
dir2 fich1.txt mifichero.txt Público Videos
ico@ico-virtual-machine:~$ mv dir2 Directorio2
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich2.txt Música secret.txt
capítulo3 ejemplo.txt fich3.txt Plantillas snap
Descargas Escritorio Imágenes prueba.txt vengadores
Directorio2 fich1.txt mifichero.txt Público Videos
ico@ico-virtual-machine:~$ 

```

4.5.2 Gestión de ficheros

4.5.2.1 Comando rm

Este comando permite borrar un fichero o archivo. La buena noticia es que coincide con el comando utilizado para borrar un directorio.



En el siguiente ejemplo queremos borrar el fichero fich3.txt:

```

ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich2.txt Música secret.txt
capítulo3 ejemplo.txt fich3.txt Plantillas snap
Descargas Escritorio Imágenes prueba.txt vengadores
Directorio2 fich1.txt mifichero.txt Público Videos
ico@ico-virtual-machine:~$ rm fich3.txt
ico@ico-virtual-machine:~$ ls
bar.txt Documentos fich2.txt Plantillas snap
capítulo3 ejemplo.txt Imágenes prueba.txt vengadores
Descargas Escritorio mifichero.txt Público Videos
Directorio2 fich1.txt Música secret.txt
ico@ico-virtual-machine:~$ 

```

4.5.2.2 Comando cp

Este comando permite copiar un fichero a un directorio determinado.



En el siguiente ejemplo queremos copiar el fichero fich1.txt al directorio Documentos:

```

ico@ico-virtual-machine:~$ ls
bar.txt      Documentos    fich2.txt      Plantillas  snap
capítulo3   ejemplo.txt  Imágenes      prueba.txt  Vengadores
Descargas    Escritorio   mifichero.txt Público    Videos
Directorio2  fich1.txt    Música       secret.txt
ico@ico-virtual-machine:~$ cp fich1.txt Documentos
ico@ico-virtual-machine:~$ ls Documentos
Directorio1  midescarga1.txt midescarga3.txt
fich1.txt    midescarga2.txt
ico@ico-virtual-machine:~$ 

```

4.5.2.3 Comando mv

Este comando permite mover un fichero a otro directorio.



En el siguiente ejemplo queremos mover el fichero fich1.txt del directorio Documentos al directorio padre:

```

ico@ico-virtual-machine:~$ cd Documentos
ico@ico-virtual-machine:~/Documentos$ ls
Directorio1  midescarga1.txt midescarga3.txt
fich1.txt    midescarga2.txt
ico@ico-virtual-machine:~/Documentos$ mv fich1.txt ..
ico@ico-virtual-machine:~/Documentos$ ls ..
bar.txt      Documentos    fich2.txt      Plantillas  snap
capítulo3   ejemplo.txt  Imágenes      prueba.txt  Vengadores
Descargas    Escritorio   mifichero.txt Público    Videos
Directorio2  fich1.txt    Música       secret.txt
ico@ico-virtual-machine:~/Documentos$ 

```

➔ Hacer Actividad 4.3

4.6. Administración de usuarios

4.6.1 Administrador del sistema (root)

El administrador del sistema o simplemente el root, es un usuario especial que tiene privilegios para cambiar la configuración, borrar y crear ficheros en cualquier directorio, crear nuevos grupos y usuarios, etc, por lo que, para ejecutar ciertos comandos, necesitamos “convertirnos” en root.

Para tener los privilegios de root se debe poner el comando **sudo** antes del comando a ejecutar, sino mostrará un mensaje de error “Permission denied”. En el ejemplo, se usa el comando **useradd** que permite crear un nuevo usuario y que requiere permisos de root para ejecutarse. En primer lugar, se intenta ejecutar sin “sudo” dando error y a continuación, se utiliza el comando “sudo” delante del comando “useradd”, pedirá la clave del usuario ico que es el administrador del sistema operativo y finalmente, creará el usuario ana.

```
ico@ico-virtual-machine:/$ useradd -m ana
useradd: Permission denied.
useradd: no se pudo bloquear /etc/passwd, inténtelo de nuevo.
ico@ico-virtual-machine:/$ sudo useradd -m ana
[sudo] contraseña para ico:
ico@ico-virtual-machine:/$
```

4.6.2 Comandos su, whoami y groups

El comando `whoami` permite determinar con qué usuario estamos logeados. Esto es útil ya que desde la terminal es posible ejecutar comandos como otro usuario.

```
ico@ico-virtual-machine:~$ whoami
ico
ico@ico-virtual-machine:~$
```

El comando `su` permite ejecutar comandos como otro usuario distinto, siempre y cuando sepamos la contraseña de ese otro usuario. Es importante darse en cuenta que cuando introduces la contraseña, ésta no se muestra por pantalla para evitar robos, por lo que parecerá que no estamos escribiendo, aunque sí estemos introduciendo la contraseña.

```
ico@ico-virtual-machine:~$ whoami
ico
ico@ico-virtual-machine:~$ su usuario_prueba
Contraseña:
usuario_prueba@ico-virtual-machine:/home/ico$ whoami
usuario_prueba
usuario_prueba@ico-virtual-machine:/home/ico$
```

Para volver a ser el usuario original basta con utilizar `exit`.

Todo usuario pertenece, por defecto, a un grupo de usuarios, aunque puede pertenecer a varios como veremos en el siguiente. Con el comando `groups` se puede ver a qué grupo pertenecemos.

```
ico@ico-virtual-machine:~$ groups
ico adm cdrom sudo dip plugdev lpadmin lxd sambashare
ico@ico-virtual-machine:~$
```

4.6.3 Cuentas de usuario

Todo ordenador con sistema operativo Linux está preparado para ser utilizado por más de un usuario, incluso simultáneamente (ya que se trata de un sistema operativo multitarea). Pero además de los usuarios reales, el sistema operativo hace uso de otros usuarios, con ciertos privilegios, asociados a los servicios que ofrece.

4.6.3.1 ID de usuarios. UID.

Linux asigna a cada usuario un número de identificación que se conoce como ID de usuario (UID). Linux en realidad no utiliza los nombres de usuario para llevar el control de los mismos sino que utiliza los UID. El UID 0 (cero) es el utilizado para el usuario root, los UID de 1 a 99 los reserva el sistema para su propio uso asignándolos a sus propias cuentas internas. Más allá del ID 100, las ID de usuario se encuentran disponibles para los usuarios normales, aunque muchas distribuciones reservan hasta el UID 500 o incluso el 1000 para fines especiales. Así, es corriente que el primer usuario que se cree en el sistema por nuestra parte reciba por parte del sistema el UID 1000. El límite de ID de usuarios es de 65.536 en los kernel 2.2.x y sobre 4,2 billones con los kernel 2.4.x y posteriores.

4.6.3.2 Funcionamiento de la autenticación en Linux

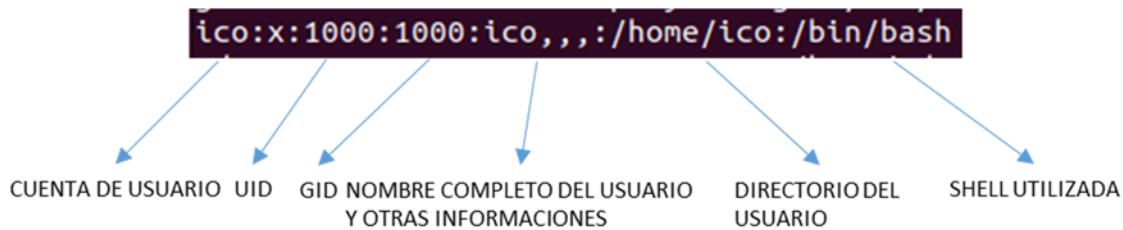
En los primeros tiempos de Linux el mecanismo de identificación de usuarios se basaba en el fichero “/etc/passwd”, donde se encontraban las claves de los usuarios cifradas, pero podía ser leído su contenido por cualquiera de sus usuarios (era necesario, pero añadía cierta inseguridad).

Para superar la fragilidad de este primer sistema de autenticación apareció el sistema actual denominado “shadow suite” (creado por Julianne F. Haugh) que mejoraba sustancialmente la seguridad y hacía más cómodo el mantenimiento de las cuentas de usuario. Este nuevo mecanismo de autenticación se basa en el mantenimiento del archivo “/etc/passwd” donde se encuentran las cuentas de usuario, pero se añade un archivo “/etc/shadow” donde se encuentran las claves cifradas de dichos usuarios y donde únicamente el usuario administrador puede acceder, de esta manera se aumenta la seguridad del sistema.

Vamos a ver el contenido del fichero /etc/passwd”, en lugar de usar el comando cat, se ha utilizado el comando tail que muestra las últimas filas de un fichero. Es muy útil cuando el fichero es muy grande y quieres visualizar el final del mismo.

```
ico@ico-virtual-machine:~$ tail /etc/passwd
gnome-initial-setup:x:125:65534::/run/gnome-initial-setup/:/bin/false
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false
ico:x:1000:1000:ico,,,:/home/ico:/bin/bash
edurne:x:1001:1002:Ana,2,33,33,3:/home/edurne:/bin/bash
juan:x:1002:1002:Jul,44,4,4,4:/home/juan:/bin/bash
mickey:x:1003:1004:Mickey,4,444,3333:/home/mickey:/bin/bash
prueba:x:1004:1006:prueba,3,3,3:/home/prueba:/bin/bash
ana:x:1005:1007::/home/ana:/bin/sh
usuario_prueba:x:1006:1008:UsuarioPrueba,3,44,55:/home/usuario_prueba:/bin/bash
ico@ico-virtual-machine:~$
```

Por ejemplo, la información del usuario ico es la siguiente:



El fichero “/etc/passwd” está compuesto por líneas cuyo contenido es similar al que muestra la imagen anterior. Cada línea define un usuario del sistema, donde cada una de las características de éste están separadas por el carácter “:”. Fíjese que además de las cuentas de usuario (como la correspondiente a “ICO”) existen otras cuentas (por ejemplo “hplip”) usadas por servicios del sistema.

Este fichero se complementa con el fichero “/etc/shadow” que contiene la clave cifrada del usuario, cuyo contenido es:

```

ico@ico-virtual-machine:~$ sudo tail /etc/shadow
gnome-initial-setup:*:19101:0:99999:7:::
hplip:*:19101:0:99999:7:::
gdm:*:19101:0:99999:7:::
ico:$y$j9T$.cdBw9ydM29gaQcZ1Hecc.$Ilye2SJmqyex0NTdNH0qCRn5X5Zc.kZHfCMJt1Jia3C:19149:0
:99999:7:::
edurne:$y$j9T$dzEyVOXnEJnU5AQKvKINo.$lMrwVgrGMtyy0Qczdt1c9.ft5EUl2xGZhHhKUZFMS/:1927
5:0:99999:7:::
juan:$y$j9T$jsQwnyUX.LLHWyJbfMDJE/$v.63bVcd13do6rubgYdr02WHf.MYZgC4YjcOWDiECP.:19275:
0:99999:7:::
mickey:$y$j9T$4n30A8W6CLBLrCQQmQcFr1$Ud6oipxjoRHOSWwP4Ymv4Nz9C/wCgxdHIp3078L6oL7:1928
3:0:99999:7:::
prueba:$y$j9T$QiUHihzH21lNHPi1UfbId0$Mhw4IfPgFIHlfL7alqXRBC6KtnzRYllRmonDVB165/3:1928
3:0:99999:7:::
ana:!:19326:0:99999:7:::
usuario_prueba:$y$j9T$jdNVFBJKDJakq8.I8mBQE0$cdbnUrGLSJoVxDpAWxjEJYEoKgrRWU58c5J.pwulo
qWA:19340:0:99999:7:::
ico@ico-virtual-machine:~$ 

```

Si nos centramos en el usuario ico:

```

ico:$y$j9T$.cdBw9ydM29gaQcZ1Hecc.$Ilye2SJmqyex0NTdNH0qCRn5X5Zc.kZHfCMJt1Jia3C:19149:0
    |
    +--> CUENTA DE USUARIO
    +--> CLAVE CIFRADA
    +--> PROPIEDADES DE LA CUENTA

```

Como ocurría con el fichero “/etc/passwd”, este otro (“/etc/shadow”) contiene una línea por cada usuario del sistema, y en ella cada una de las características de la cuenta viene separada por el carácter “:”.

Las “Propiedades de la cuenta” de usuario hacen referencia a información sobre cuando fue cambiada por última vez la clave, cuantos días faltan hasta que sea solicitado el cambio de clave, cuanto tiempo queda hasta que la cuenta expire o quede inactiva.

4.6.3.3 Gestión de usuarios

4.6.3.3.1 Comando useradd

Comenzaremos por el comando `useradd` que es el necesario para crear una nueva cuenta de usuario en el sistema. Su sintaxis es:

COMANDO	OPCIONES	CUENTA DE USUARIO
<code>useradd</code>	<code>-opciones</code>	<code>nombre_cuenta</code>

Si ejecutamos el comando sin opción alguna, por ejemplo, "useradd usuario1" creará una nueva cuenta llamada "usuario1" cuyas características serán las tomadas por defecto del sistema (la shell será normalmente "/bin/sh", no tendrá carpeta de usuario). Es sencillo y directo, pero nos puede interesar cuales son las opciones que podemos utilizar, sobre todo porque están relacionadas con aspectos tan importantes como el tiempo de activación de la cuenta o el directorio de trabajo de dicho usuario. Estas opciones son:

Opción	Descripción
<code>-b <directorio></code>	Si queremos cambiar el directorio del usuario
<code>-c <nombre></code>	Para añadir el nombre completo del usuario
<code>-e <fecha></code>	Para indicar la fecha en la que la cuenta expira. Se deberá especificar con el formato YYYY-MM-DD
<code>-s <Shell></code>	Para indicar el Shell que queremos para ese usuario
<code>-f <nº días></code>	Para indicar el número de días en los que la cuenta estará inactiva hasta que finalmente sea desactivada
<code>-m</code>	Para crear el directorio de usuario si éste no existe.

```
ico@ico-virtual-machine:~$ sudo useradd -m usuario1  
[sudo] contraseña para ico:
```

```
ico@ico-virtual-machine:~$ sudo tail /etc/passwd  
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false  
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false  
ico:x:1000:1000:ico,,,:/home/ico:/bin/bash  
edurne:x:1001:1002:Ana,2,33,33,3:/home/edurne:/bin/bash  
juan:x:1002:1002:Jul,44,4,4,4:/home/juan:/bin/bash  
mickey:x:1003:1004:Mickey,4,444,3333:/home/mickey:/bin/bash  
prueba:x:1004:1006:prueba,3,3,3:/home/prueba:/bin/bash  
ana:x:1005:1007:::/home/ana:/bin/sh  
usuario_prueba:x:1006:1008:UsuarioPrueba,3,44,55:/home/usuario_prueba:/bin/bash  
usuario1:x:1007:1009:::/home/usuario1:/bin/sh  
ico@ico-virtual-machine:~$ ls /home  
ana edurne ico juan mickey prueba usuario1 usuario_prueba  
ico@ico-virtual-machine:~$
```

Añade una cuenta llamada "usuario1", a la que se creará un directorio de usuario denominado /home/usuario1.

4.6.3.3.2 Comando adduser

El comando `adduser` nos permite crear cuentas de usuario, pero con unas diferencias importantes. La sintaxis es:

COMANDO	OPCIONES	CUENTA DE USUARIO
adduser	-opciones	nombre_cuenta

Veamos un ejemplo sencillo de cómo el comando adduser crea un usuario.

```
lico@lico-virtual-machine:~$ sudo adduser usuario1
Añadiendo el usuario `usuario1' ...
Añadiendo el nuevo grupo `usuario1' (1009) ...
Añadiendo el nuevo usuario `usuario1' (1007) con grupo `usuario1' ...
El directorio personal `/home/usuario1' ya existe. No se copiará desde `/etc/skel'.
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para usuario1
Introduzca el nuevo valor, o presione INTRO para el predeterminado
    Nombre completo []: Usuario 1
    Número de habitación []:
    Teléfono del trabajo []: 944139000
    Teléfono de casa []:
    Otro []:
¿Es correcta la información? [S/n] s
lico@lico-virtual-machine:~$ tail /etc/passwd
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:127:133:Gnome Display Manager:/var/lib/gdm3:/bin/false
ico:x:1000:1000:ico,,,:/home/ico:/bin/bash
edurne:x:1001:1002:Ana,2,33,33,3:/home/edurne:/bin/bash
juan:x:1002:1002:Jul,44,4,4,4:/home/juan:/bin/bash
mickey:x:1003:1004:Mickey,4,444,3333:/home/mickey:/bin/bash
prueba:x:1004:1006:prueba,3,3,3:/home/prueba:/bin/bash
ana:x:1005:1007::/home/ana:/bin/sh
usuario_prueba:x:1006:1008:UsuarioPrueba,3,44,55:/home/usuario_prueba:/bin/bash
usuario1:x:1007:1009:Usuario 1,,944139000,:/home/usuario1:/bin/bash
lico@lico-virtual-machine:~$
```

Una vez que ejecuta el comando “adduser usuario1” se le preguntarán por una serie de preguntas relacionadas con el usuario, tales como la password, nombre y otros datos.

Como aprecia, este comando es más completo que “useradd”. Finalmente, comprobamos si ha creado el usuario en el fichero “/etc/passwd” y el directorio de trabajo del usuario.

En esta imagen puede apreciar la diferencia entre crear un usuario con el comando “useradd” y el comando “adduser” para el fichero “/etc/passwd”.

Pero, ¿cómo afecta ambos comandos al fichero “/etc/shadow”? Veámoslo.

Primero crearemos un “usuario2” con el comando useradd.

```
lico@lico-virtual-machine:~$ sudo useradd usuario2
```

Y ahora veamos el contenido el fichero /etc/shadow:

```

ico@ico-virtual-machine:~$ sudo tail /etc/shadow
gdm:*:19101:0:99999:7:::
ico:$y$j9T$.cdBw9ydM29gaQcZ1Hecc.$Ilye2SJmqyex0NTdNH0qCRn5X5Zc.kZHfCMJt1Jia3C:19149:0:99999:7:::
edurne:$y$j9T$dzEyVOXnEJnUSAQKVkIno.$lMrwVgrGMtYY0Qczdtic9.fTEU12xGZhHhKUZFMS:/19275:0:99999:7:::
juan:$y$j9T$jsQwnyUX.LLHWyJbfMDJE/$v.63bVcd13do6rubgYdrO2WHz.MYZgC4Yjc0WDiECP.:19275:0:99999:7:::
mickey:$y$j9T$4n30A8W6CLBLrCQ0mQcFr1$Ud6o1pxj0RH0SwP4Ymv4Nz9C/wCxdhIp3078L6oL7:19283:0:99999:7:::
prueba:$y$j9T$QiuHlhZ21NHPI1UfbId0$Mhw4IfPgFIHlfL7alqXRBC6KtnzRYllRmonDVB165/3:19283:0:99999:7:::
ana:!:19326:0:99999:7:::
usuario_prueba:$y$j9T$jdNVFBJKDJakq8.I8mBQE0$cdbnUrGLSJoVxDpAWxjEJYEoKgrRWU58c5J.pwul0qWA:19340:0:99999:9:::
usuario1:$y$j9T$TY75bZRUDCC09j.JJno0V1$ob3sQ7Vzk/tVSw4VtZYOWU5cVIYYlUr2xJhY8J6oKAC:19340:0:99999:7:::
usuario2:!:19340:0:99999:7:::
ico@ico-virtual-machine:~$ 

```

El carácter “!” del usuario 2 indica que la cuenta no tiene contraseña. ¡El usuario “usuario2”, creado con el comando “useradd” no tiene contraseña! ¡cualquier usuario podría usar esta cuenta! Es lógico que ocurra esto si ha ejecutado el comando “useradd” que le ponía como ejemplo, ya que en ningún momento nos pidió clave alguna; mientras que el comando “adduser”, mediante el pequeño asistente, nos preguntó por dicha contraseña.

La diferencia entre uno y otro comando es que “adduser” se utiliza para la creación de cuentas de usuario, mientras que “useradd” se usa para crear cuentas para los servicios del sistema (aunque ha podido comprobar que también podemos crear cuentas de usuario, pero de una manera más primitiva).

4.6.3.3.3 Comando passwd

El problema anterior (usuario sin contraseña) se puede solucionar con el comando passwd que se utiliza para crear o modificar la contraseña de un usuario del sistema. Su sintaxis es:

COMANDO	OPCIONES	CUENTA DE USUARIO
passwd	-opciones	nombre_cuenta

Si queremos crear una contraseña para el usuario “usuario2” tendríamos que ejecutar el comando “passwd usuario2”, y esto hará que nos pregunte por dicha contraseña.

```

ico@ico-virtual-machine:~$ sudo passwd usuario2
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
ico@ico-virtual-machine:~$ 

```

Algunas opciones adicionales que pueden indicarse en el comando “passwd” son:

Opción	Descripción
-d	Borra la contraseña del usuario especificado
-e	Fuerza la caducidad de la contraseña del usuario
--help	Muestra una ayuda breve sobre el comando passwd
-i INACTIVE	Si la contraseña de la cuenta lleva caducada INACTIVE días, entonces la cuenta será desactivada
-k	El cambio de contraseña sólo tiene lugar si está caducada
-l	Bloquea la cuenta indicada
-n MIN_DAYS	Obliga a que la contraseña se cambie, como muy pronto, cada MIN_DAYS días.
-S	Informa del estado de la contraseña
-u	Desbloquea la cuenta indicada

-w WARN_DAYS	Preaviso de la fecha de caducidad WARN_DAYS días antes
-x MAX_DAYS	Obliga a que la contraseña se cambia, como muy tarde, cada MAX_DAYS días.

Como siempre, si requiere información adicional puede consultar las páginas del manual (“man passwd”).

4.6.3.3.4 Asignar grupos a los usuarios

Como se ha indicado previamente, los usuarios pueden pertenecer a grupos de usuarios que ayudarán a establecer los mismos permisos a todos los usuarios de un mismo grupo. Se puede indicar el grupo al que pertenecerá el usuario durante la ejecución del comando adduser.

Supongamos que queremos crear el usuario “usuario3” y asignarle el grupo “clase3”. Lo primero que habrá que hacer es crear el grupo “clase3”, con el comando groupadd, cuya sintaxis es:

COMANDO	OPCIONES	CUENTA DE GRUPO
groupadd	-opciones	nombre_grupo

```
ico@ico-virtual-machine:~$ sudo groupadd clase3
ico@ico-virtual-machine:~$
```

Y a continuación, crearemos el usuario3 con la opción ingroup para añadirle al grupo clase3.

```
ico@ico-virtual-machine:~$ sudo adduser usuario3 --ingroup clase3
Añadiendo el usuario `usuario3' ...
Añadiendo el nuevo usuario `usuario3' (1009) con grupo `clase3' ...
Creando el directorio personal `/home/usuario3' ...
Copiando los ficheros desde `/etc/skel' ...
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para usuario3
Introduzca el nuevo valor, o presione INTRO para el predeterminado
  Nombre completo []: Usuario 3
  Número de habitación []:
  Teléfono del trabajo []:
  Teléfono de casa []:
  Otro []:
¿Es correcta la información? [S/n] s
ico@ico-virtual-machine:~$
```

Imaginemos que queremos que el usuario3 pertenezca también al grupo clase4. Primero, crearemos el grupo, al no existir, y después le asignaremos el grupo al usuario3 con el comando adduser.

```
ico@ico-virtual-machine:~$ sudo groupadd clase4
ico@ico-virtual-machine:~$ sudo adduser usuario3 clase4
Añadiendo al usuario `usuario3' al grupo `clase4' ...
Añadiendo al usuario usuario3 al grupo clase4
Hecho.
ico@ico-virtual-machine:~$
```

Para comprobar que se ha realizado correctamente, el comando groups permite conocer los grupos a los que pertenece un usuario.

COMANDO	CUENTA DE USUARIO
groups	nombre_cuenta

```
ico@ico-virtual-machine:~$ groups usuario3
usuario3 : clase3 clase4
ico@ico-virtual-machine:~$
```

4.6.3.3.5 Comando userdel

Una buena medida de seguridad es eliminar aquellas cuentas de usuario que no estén siendo utilizadas, para ello contamos con el comando “userdel”, cuya sintaxis de utilización es:

COMANDO	OPCIONES	CUENTA DE USUARIO
userdel	-opciones	nombre_cuenta

Si queremos eliminar la cuenta de usuario “usuario2”, y el directorio donde almacena sus archivos, solo tendremos que ejecutar el comando “userdel -r usuario2”.

```
ico@ico-virtual-machine:~$ sudo userdel -r usuario2
```

4.6.3.3.6 Comando usermod

El comando “usermod” permite modificar alguna de las propiedades de la cuenta de usuario.

Su sintaxis es:

COMANDO	OPCIONES	CUENTA DE USUARIO
usermod	-opciones	nombre_cuenta

Las opciones que podemos utilizar son:

Opción	Descripción
-d <directorio>	Si queremos cambiar el directorio del usuario
-e <fecha>	Para indicar la fecha en la que la cuenta expira. Se deberá especificar con el formato YYYY-MM-DD
-l <nuevo_usuario>	Para indicar un nuevo nombre para la cuenta de usuario
-f <nº días>	Para indicar el número de días en los que la cuenta estará inactiva hasta que finalmente sea desactivada

Si quiere consultar el listado completo de opciones puede hacerlo en las hojas del manual “man usermod”.

4.6.4 Cuentas de grupo

4.6.4.1 Uso de los grupos

Los grupos se utilizan para organizar a los usuarios. Cada fichero en un sistema Linux está asociado a un grupo específico, pudiéndose asignar permisos a dicho grupo. Esto permite que un usuario pueda tener permisos en un fichero simplemente por pertenecer al grupo que tiene permisos en dicho fichero. Un usuario puede pertenecer a múltiples grupos y un grupo puede contar con varios, con uno o con ningún usuario miembro. Así, si el grupo profesores cuenta con permisos para leer un fichero, todo usuario miembro del grupo podrá leerlo.

Cada usuario pertenece a un grupo por defecto o grupo primario que viene definido en la configuración del usuario (fichero /etc/passwd). Cuando un usuario realiza acciones en el sistema (crear un fichero, ejecutar un programa...) estas acciones se asocian con el grupo primario del usuario de modo que, si por ejemplo un usuario crea un fichero, ese fichero pasa

a ser propiedad del grupo principal del usuario. Aparte de este grupo primario, cada usuario de Linux puede pertenecer a todos los grupos secundarios que necesite.

Es común en varias distribuciones Linux crear automáticamente una cuenta de grupo por cada cuenta de usuario usando el mismo nombre para ambas y asignar dicho grupo como grupo principal del usuario. Así, si creamos una cuenta de usuario con nombre alumno, se crea también una cuenta de grupo con nombre alumno que será su grupo primario.

4.6.4.2 ID de grupos. GID.

Al igual que las cuentas de usuario, cada cuenta de grupo es identificada por el sistema mediante un número de identificación (GID). Todo lo que se ha explicado para los ID de usuario se aplica a los ID de grupo, así por ejemplo el GID del grupo root es el 0.

4.6.4.3 Gestión de grupos

Cuando hemos comentado la utilización de la “Shadow Suite” como el mecanismo utilizado para la autenticación de usuarios, hemos hablado de la participación de los ficheros “/etc/passwd” y “/etc/shadow”, pero no hemos comentado que también participa el archivo “/etc/group”, que es el fichero que almacena la información correspondiente a los grupos de usuarios del sistema.

```
ico@ico-virtual-machine:/home$ sudo tail /etc/group
diseño:x:1002:
maquetacion:x:1003:edurne
Disney:x:1004:
marvel:x:1005:
prueba:x:1006:
ana:x:1007:
usuario_prueba:x:1008:
usuario1:x:1009:
clase3:x:1011:
clase4:x:1012:usuario3
ico@ico-virtual-machine:/home$
```

Este es el aspecto del fichero “/etc/group”, y como puede ver, lo habitual es que para cada usuario existirá su propio grupo que además tendrá el mismo nombre. Esto quiere decir que cuando hemos utilizado el comando “adduser” para crear un usuario, además de crear una cuenta de usuario en el fichero “/etc/passwd” también creará una cuenta de grupo en el fichero “/etc/group”.

4.6.4.3.1 Comando groupadd

Recordando, para crear un nuevo grupo se utiliza el comando:

COMANDO	OPCIONES	CUENTA DE GRUPO
groupadd	-opciones	nombre_grupo

4.6.4.3.2 Comando groupdel

Si queremos eliminar un grupo existente disponemos del comando “groupdel”, al que solo tenemos que indicar el nombre de grupo.

```
ico@ico-virtual-machine:~$ sudo groupadd grupo1
ico@ico-virtual-machine:~$ sudo groupdel grupo1
```

4.6.4.3.3 Comando groupmod

El comando “groupmod” permite modificar el grupo. Su sintaxis es:

COMANDO	OPCIONES	CUENTA DE GRUPO
groupmod	-opciones	nombre_grupo

Supongamos que creamos el grupo11 y queremos cambiarle el nombre de grupo11 a grupo1, los comandos serían:

```
ico@ico-virtual-machine:~$ sudo groupadd grupo11
ico@ico-virtual-machine:~$ sudo groupmod -n grupo1 grupo11
ico@ico-virtual-machine:~$ sudo tail /etc/group
maquetacion:x:1003:edurne
Disney:x:1004:
marvel:x:1005:
prueba:x:1006:
ana:x:1007:
usuario_prueba:x:1008:
usuario1:x:1009:
clase3:x:1011:
clase4:x:1012:usuario3
grupo1:x:1013:
ico@ico-virtual-machine:~$
```

→ Hacer Actividad 4.4

→ Hacer Actividad 4.5

4.7. Permisos de archivos y directorios

Con los comandos que conoce hasta ahora puede moverse entre los directorios del sistema de ficheros y listar su contenido, pero poco más. Es el momento de dar el siguiente paso, aprender a identificar un directorio de un fichero, y reconocer los atributos que poseen éstos.

Para entender todo lo que queremos explicar a continuación, partiremos de un ejemplo. Consiste en el listado del contenido del directorio “/etc/X11” mediante el comando “ls”.

```
ico@ico-virtual-machine:~$ ls -al /etc/X11
total 104
drwxr-xr-x 12 root root 4096 abr 19 2022 .
drwxr-xr-x 130 root root 12288 dic 14 09:55 ..
drwxr-xr-x 2 root root 4096 abr 19 2022 app-defaults
drwxr-xr-x 2 root root 4096 abr 19 2022 cursors
-rw-r--r-- 1 root root 15 abr 19 2022 default-display-manager
drwxr-xr-x 4 root root 4096 abr 19 2022 fonts
-rw-r--r-- 1 root root 17394 oct 9 2020 rgb.txt
drwxr-xr-x 2 root root 4096 abr 19 2022 xinit
drwxr-xr-x 2 root root 4096 ago 18 2021 xkb
drwxr-xr-x 2 root root 4096 mar 25 2022 xorg.conf.d
-rwrxr-xr-x 1 root root 709 oct 9 2020 Xreset
drwxr-xr-x 2 root root 4096 abr 19 2022 Xreset.d
drwxr-xr-x 2 root root 4096 abr 19 2022 Xresources
-rwrxr-xr-x 1 root root 4139 oct 19 2021 Xsession
drwxr-xr-x 2 root root 4096 dic 13 09:15 Xsession.d
-rw-r--r-- 1 root root 265 oct 9 2020 Xsession.options
drwxr-xr-x 2 root root 4096 abr 19 2022 xsm
-rw-r--r-- 1 root root 13 mar 24 2022 XvMCConfig
-rw-r--r-- 1 root root 630 abr 19 2022 Xwrapper.config
ico@ico-virtual-machine:~$
```

Cada una de estas líneas se refiere a un fichero o directorio que se encuentra en el directorio “/etc/X11”. Vamos a explicar que quiere decir cada una de los datos que aparecen en esas líneas.

Comencemos por el conjunto de 10 caracteres que aparecen al principio de cada línea y que representan el tipo de archivo (solo el primero de dichos caracteres) y los permisos que posee (los restantes 9 caracteres).



Recuerde que todo es un archivo en Linux. El primero de los caracteres indica el tipo de archivo al que corresponde esa entrada. Existen siete tipos de archivos y son representados mediante ese carácter de la siguiente forma:

Indicador	Tipo de archivo
-	Archivo normal
d	Directorio
l	Enlace
c	Dispositivo de caracteres
s	Socket (relacionado con la comunicación TCP/IP)
P	Tubería con nombre

Los más utilizados son los tres primeros tipos correspondientes a un archivo normal, un directorio y un enlace (que no es más que el clásico “acceso directo” a un archivo) respectivamente.

Lo siguiente que veremos será los siguientes nueve caracteres. Todos ellos se refieren a permisos que tendrán determinados usuarios, de forma que:

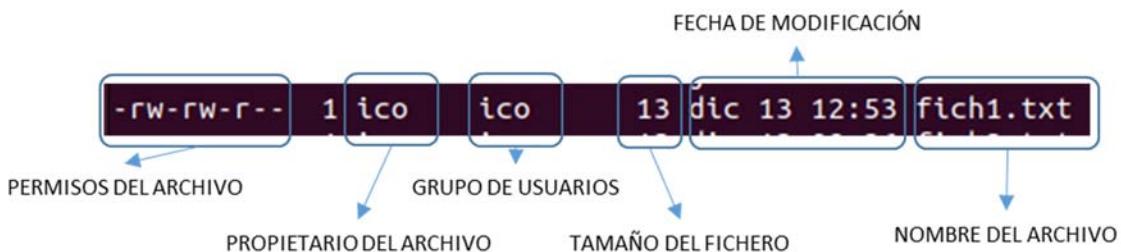
- Los tres primeros caracteres indican los permisos de lectura (representada por la letra “r”), escritura (representada por la letra “w”) y ejecución (representada por la letra “x”) para el propietario del archivo. Se considera propietario del archivo a aquel que ha creado el mismo (en el caso de ficheros del sistema, el propietario suele ser “root” o un usuario específico asociado al servicio que ofrezca).
- Los siguientes tres caracteres corresponden a los permisos de lectura, escritura y ejecución para el grupo de usuarios propietario del archivo. En las primeras versiones de Linux, los usuarios solían agruparse en grupos que compartían ciertas características; en la actualidad, cada usuario tiene su propio grupo, que además tendrá el mismo nombre que el usuario.
- Los últimos tres caracteres representan los permisos de lectura, escritura y ejecución para cualquier otro usuario que no pertenece al sistema.

La forma en la que se codifican los permisos para cada grupo funciona de la siguiente forma. Observe los tres primeros caracteres que representan los permisos para el propietario del fichero.



Cuando en el primero de ellos aparece la letra “r” indica que el propietario del fichero tiene permiso para leerlo, pero si aparece un guión en su lugar indicará que el propietario del fichero no tiene permiso para leerlo. El siguiente carácter será el que corresponde al permiso de escritura, y si aparece la letra “w” indicará que el propietario tiene permiso para la escritura, pero si en su lugar aparece un guión (“-”), eso indicará que el propietario no puede escribir sobre él (o lo que es lo mismo, modificarlo). El tercer carácter representará el permiso de ejecución, de tal forma que si aparece la letra “x” indicará que el propietario del fichero puede ejecutarlo, pero si aparece un guión indicará que no puede ejecutarlo.

Esa misma codificación se aplica para los otros dos grupos (grupo y otros usuarios) de permisos. De esta forma se puede indicar perfectamente el nivel de permisos para cada usuario. Pongamos un ejemplo completo. Imagine que tenemos un archivo como el mostrado a continuación.



Como puede apreciar, el usuario “ico” es el propietario del archivo (también se llama “ico” el grupo que tiene permisos sobre el archivo). A la vista de los permisos podemos afirmar lo siguiente:

- El usuario “ico”, como propietario del archivo “fich1.txt”, podrá leer y escribir, pero no ejecutar el contenido de archivo.
- El grupo “ico” podrá leer y escribir en el archivo, aunque no podrá ejecutarlo.
- Por último, el resto de usuarios que no pertenecen al grupo podrán leer el archivo.

Lo que nos falta por comentar, después de ver los tipos de archivos y los permisos de usuarios, es el tamaño del fichero o directorio que aparecerá justo a continuación del nombre del grupo propietario del fichero. Y a continuación del tamaño encontraremos la fecha de creación o modificación del archivo.

Como puede observar, gracias al comando “ls -al” ejecutado en un directorio podremos conocer si los archivos listados son ficheros o directorios, cuáles son sus permisos para el propietario y grupo, y tamaño.

A continuación, veremos que comandos tenemos en Linux para poder modificar el propietario o grupo asociado a un archivo, o cómo cambiar los permisos para cada uno de ellos.

- **chmod**: este comando nos permitirá cambiar los permisos que posee el archivo o directorio.
- **chown**: será el comando que tenemos que utilizar si queremos cambiar el propietario de un archivo.
- **chgrp**: cambia el grupo propietario del fichero o directorio.

Veamos con más detenimiento cada uno de ellos.

4.7.1 Comando chmod

En algunas ocasiones puede ser necesario realizar un ajuste en los permisos de un archivo, porque pretendemos que un archivo no se pueda ejecutar o porque sencillamente no queremos que otros usuarios puedan leer el contenido de un archivo o directorio. Para poder hacer esos ajustes en los permisos de un archivo tenemos el comando “chmod”.

Existen dos formas de trabajar con el comando “chmod”: formato octal y formato simbólico

4.7.1.1 Formato octal

Su sintaxis es:

COMANDO	NOMBRE ARCHIVO
chmod	valor_octal nombre_archivo

El valor octal es un número de tres cifras que codifica los permisos que queremos para el fichero o directorio indicado como segundo argumento del comando. Cada una de esas tres cifras puede tomar un valor que va desde 0 a 7. La primera de las tres cifras representará los permisos para el propietario del archivo o directorio. La segunda cifra representa los permisos para el grupo propietario del archivo, y la tercera cifra representa los permisos para los restantes usuarios.

Los valores que puede tomar cada una de esas tres cifras es:

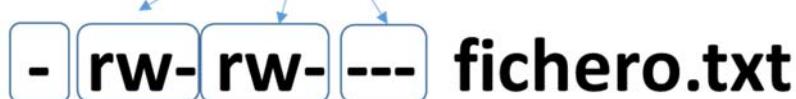
Permiso	Valor octal
rwx	7
rw-	6
r-x	5
r --	4
-wx	3
-w-	2
--x	1
---	0

La mejor manera de entender cómo funciona esta notación es mediante un ejemplo. Suponga que tenemos un fichero con los siguientes permisos:



Desea cambiar los permisos del fichero “fichero.txt” para que el propietario pueda leer y escribir sobre el documento, el grupo pueda también leer y escribir, aunque el resto de usuarios no dispondrán de ningún permiso sobre el fichero. Esto podemos hacerlo mediante el siguiente comando.

chmod 660 fichero.txt



4.7.1.2 Formato simbólico

Su sintaxis es

COMANDO	PERMISOS	NOMBRE ARCHIVO
chmod	<usuario> <operador> <permiso>	nombre_archivo

El argumento <usuario> indicará los permisos de que usuario (propietario, grupo u otros usuarios) se cambiarán. Los valores que puede tomar son: u (usuario propietario), g (grupo propietario), o (otros usuarios), o a (todos los usuarios: propietario, grupo y otros).

El argumento <operador> indica si se añaden (utiliza el símbolo "+") o eliminan (utiliza el símbolo "-") o sobrescriben (utiliza el símbolo "=") los permisos que se indicarán en el argumento <permiso>.

El argumento <permiso> indicará que permiso se aplicará: r (lectura), w (escritura) o x (ejecución). Como ya hicimos anteriormente, veamos un ejemplo de cómo se utilizaría esta notación.



chmod ug+w-x fichero.txt chmod o-rw fichero.txt



En esta ocasión hemos necesitado realizar la ejecución del comando "chmod" dos veces para poder realizar todos los ajustes. De todas formas, la utilización de uno u otro modelo dependerá de sus preferencias.

Tenga en cuenta que solo el propietario del archivo o directorio podrá ejecutar este comando de ajuste de permisos.

4.7.2 Comandos chown y chgrp

Además de poder modificar los permisos de lectura, escritura y ejecución de un archivo también podemos cambiar el usuario y grupo propietario del archivo. Esto se hace con la utilización de los comandos `chown` (“change owner” para cambiar el usuario propietario del archivo) y `chgrp` (“change group” cambiar el grupo propietario del archivo).

Para poder cambiar el propietario del archivo solo necesitamos ejecutar el comando:

COMANDO	<code>chown</code>	<code>nombre_nuevo_propietario</code>	<code>nombre_archivo</code>
---------	--------------------	---------------------------------------	-----------------------------

Como ejemplo, fíjese como estaba definido el archivo antes y después de ejecutar el comando `chown`.



Como puede apreciar, hemos indicado como primer argumento el nombre de usuario del nuevo propietario y a continuación, el nombre del fichero sobre el que se aplicará. Pero `chown` también tiene algunas acciones interesantes que debe conocer:

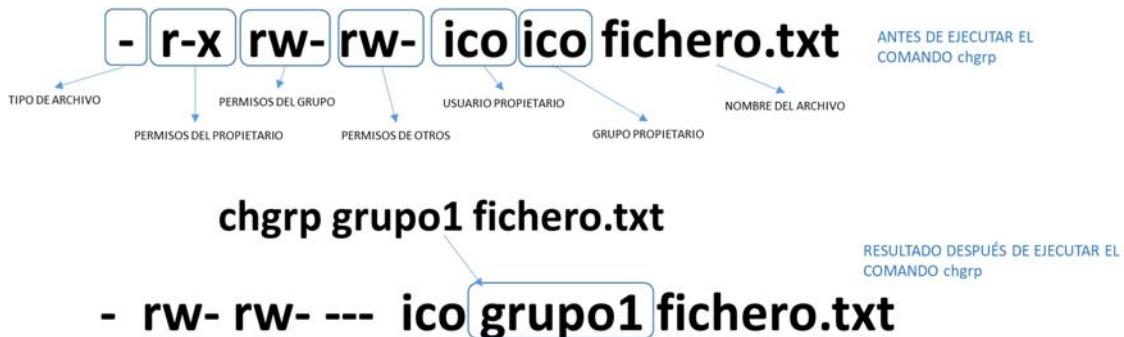
```
chown usuario1 *
```

Hará que el usuario “usuario1” sea el nuevo propietario de todos los archivos y directorios existentes en el directorio actual (ya que el asterisco es un comodín que permite identificar a todos los elementos).

```
chown -R usuario1 *
```

Esta operación pondrá al usuario “usuario1” como nuevo propietario de todos los archivos y directorios existentes en el directorio actual, pero, además, gracias al argumento “-R”, se realizará recursivamente en todos los archivos que se encuentre en dicho directorio. Si se indica un nombre de usuario que no exista en el sistema emitirá un mensaje de error.

Y para el grupo propietario de un archivo tenemos el comando `chgrp` que hace lo mismo que el comando anterior pero aplicado al grupo.



Y también le son aplicables los comentarios que habíamos realizado con respecto a la utilización de comodines (como “*”) para su aplicación a un conjunto de archivos, como la opción “-R” que conseguiría aplicar el comando `chgrp` de manera recursiva.

Ejemplos:

1º) Crear el fichero apuntes.txt y mostrar los permisos del fichero.

```
ico@ico-virtual-machine:~$ touch apuntes.txt
ico@ico-virtual-machine:~$ ls -l
total 68
-rw-rw-r-- 1 ico      ico        0 dic 15 07:58 apuntes.txt
-rw-rw-r-- 1 ico      ico        30 dic 14 08:18 bar.txt
drwxrwxr-x 4 ico      ico        4096 oct 10 14:19 capítulo3
drwxr-xr-x 2 ico      ico        4096 dic 13 11:23 Descargas
drwxrwxrwx 2 ico      ico        4096 oct 18 07:59 Directorio2
drwxr-xr-x 3 ico      ico        4096 dic 13 12:56 Documentos
-rw-rwx--- 1 mickey Disney    ico        0 oct 18 07:50 ejemplo.txt
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Escritorio
-rw-rw-r-- 1 ico      ico        13 dic 13 12:53 fich1.txt
-rw-rw-r-- 1 ico      ico        13 dic 13 09:26 fich2.txt
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Imágenes
-rw-rw-r-- 1 ico      ico        19 dic 13 09:26 mifichero.txt
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Música
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Plantillas
-rw-rw-r-- 1 ico      ico        13 oct 10 14:20 prueba.txt
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Público
-r----- 1 ico      ico        0 oct 18 08:01 secret.txt
drwx----- 3 ico      ico        4096 jun  6  2022 snap
drwxrw-rw- 2 mickey marvel   4096 oct 18 07:57 Vengadores
drwxr-xr-x 2 ico      ico        4096 jun  6  2022 Videos
ico@ico-virtual-machine:~$
```

2º) Crear el usuario administrador y cambiar el dueño del fichero apuntes.txt a administrador y muestra los permisos del fichero.

```

ico@ico-virtual-machine:~$ adduser administrador
adduser: Sólo root puede añadir un usuario o un grupo al sistema.
ico@ico-virtual-machine:~$ sudo adduser administrador
[sudo] contraseña para ico:
Añadiendo el usuario `administrador' ...
Añadiendo el nuevo grupo `administrador' (1010) ...
Añadiendo el nuevo usuario `administrador' (1008) con grupo `administrador' ...
Creando el directorio personal `/home/administrador' ...
Copiando los ficheros desde `/etc/skel' ...
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para administrador
Introduzca el nuevo valor, o presione INTRO para el predeterminado
    Nombre completo []: Administrador
    Número de habitación []:
    Teléfono del trabajo []: 112
    Teléfono de casa []:
    Otro []:
¿Es correcta la información? [S/n] s

```

```

ico@ico-virtual-machine:~$ sudo chown administrador apuntes.txt
ico@ico-virtual-machine:~$ ls -l
total 68
-rw-rw-r-- 1 administrador ico      0 dic 15 07:58 apuntes.txt
-rw-rw-r-- 1 ico          ico      30 dic 14 08:18 bar.txt
drwxrwxr-x 4 ico          ico     4096 oct 10 14:19 capítulo3
drwxr-xr-x 2 ico          ico     4096 dic 13 11:23 Descargas
drwxrwxrwx 2 ico          ico     4096 oct 18 07:59 Directorio2
drwxr-xr-x 3 ico          ico     4096 dic 13 12:56 Documentos
-rw-rwx--- 1 mickey      Disney   0 oct 18 07:50 ejemplo.txt
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Escritorio
-rw-rw-r-- 1 ico          ico      13 dic 13 12:53 fich1.txt
-rw-rw-r-- 1 ico          ico      13 dic 13 09:26 fich2.txt
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Imágenes
-rw-rw-r-- 1 ico          ico      19 dic 13 09:26 mifichero.txt
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Música
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Plantillas
-rw-rw-r-- 1 ico          ico      13 oct 10 14:20 prueba.txt
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Público
-r----- 1 ico          ico      0 oct 18 08:01 secret.txt
drwx----- 3 ico          ico     4096 jun  6 2022 snap
drwxrw-rw- 2 mickey      marvel  4096 oct 18 07:57 Vengadores
drwxr-xr-x 2 ico          ico     4096 jun  6 2022 Vídeos
ico@ico-virtual-machine:~$ 

```

3º) Añadir el permiso de ejecución a otros usuarios para el fichero apuntes.txt utilizando notación simbólica.

```

ico@ico-virtual-machine:~$ sudo chmod o+x apuntes.txt
ico@ico-virtual-machine:~$ ls -l
total 68
-rw-rw-r-x 1 ico      ico      0 dic 15 08:15 apuntes.txt
-rw-rw-r-- 1 ico      ico      30 dic 14 08:18 bar.txt
drwxrwxr-x 4 ico      ico      4096 oct 10 14:19 capitulo3
drwxr-xr-x 2 ico      ico      4096 dic 13 11:23 Descargas
drwxrwxrwx 2 ico      ico      4096 oct 18 07:59 Directorio2
drwxr-xr-x 3 ico      ico      4096 dic 13 12:56 Documentos
-rw-rwx--- 1 mickey Disney   ico      0 oct 18 07:50 ejemplo.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Escritorio
-rw-rw-r-- 1 ico      ico      13 dic 13 12:53 fich1.txt
-rw-rw-r-- 1 ico      ico      13 dic 13 09:26 fich2.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Imágenes
-rw-rw-r-- 1 ico      ico      19 dic 13 09:26 mifichero.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Música
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Plantillas
-rw-rw-r-- 1 ico      ico      13 oct 10 14:20 prueba.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Público
-r----- 1 ico      ico      0 oct 18 08:01 secret.txt
drwx----- 3 ico      ico      4096 jun  6 2022 snap
drwxrw-rw- 2 mickey marvel   ico      4096 oct 18 07:57 Vengadores
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Vídeos
ico@ico-virtual-machine:~$
```

4º) Quitar el permiso de ejecución a otros usuarios y dar permiso de ejecución a los usuarios del mismo grupo usando notación simbólica.

```

ico@ico-virtual-machine:~$ sudo chmod o-x apuntes.txt
ico@ico-virtual-machine:~$ sudo chmod g+x apuntes.txt
ico@ico-virtual-machine:~$ ls -l
total 68
-rw-rwxr-- 1 ico      ico      0 dic 15 08:15 apuntes.txt
-rw-rw-r-- 1 ico      ico      35 dic 15 08:18 bar.txt
drwxrwxr-x 4 ico      ico      4096 oct 10 14:19 capitulo3
drwxr-xr-x 2 ico      ico      4096 dic 13 11:23 Descargas
drwxrwxrwx 2 ico      ico      4096 oct 18 07:59 Directorio2
drwxr-xr-x 3 ico      ico      4096 dic 13 12:56 Documentos
-rw-rwx--- 1 Mickey Disney   ico      0 oct 18 07:50 ejemplo.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Escritorio
-rw-rw-r-- 1 ico      ico      13 dic 13 12:53 fich1.txt
-rw-rw-r-- 1 ico      ico      13 dic 13 09:26 fich2.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Imágenes
-rw-rw-r-- 1 ico      ico      19 dic 13 09:26 mifichero.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Música
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Plantillas
-rw-rw-r-- 1 ico      ico      13 oct 10 14:20 prueba.txt
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Público
-r----- 1 ico      ico      0 oct 18 08:01 secret.txt
drwx----- 3 ico      ico      4096 jun  6 2022 snap
drwxrw-rw- 2 Mickey marvel   ico      4096 oct 18 07:57 Vengadores
drwxr-xr-x 2 ico      ico      4096 jun  6 2022 Vídeos
ico@ico-virtual-machine:~$
```

5º) Dado el siguiente comando en notación octal, escribe los comandos equivalentes en notación simbólica.

`chmod 755 prueba.txt`

sería equivalente a estos tres comandos:

`chmod u+rwx prueba.txt`

`chmod g+rx-w prueba.txt`

`chmod o+rx-w prueba.txt`

Los permisos de los directorios se pueden cambiar de la misma forma que los ficheros, aunque el significado es algo diferente. Si un directorio tiene el permiso de lectura quiere decir que se puede ver su contenido. Si tiene permiso de escritura, quiere decir que se pueden crear ficheros dentro y si tiene permiso de ejecución quiere decir que se puede entrar dentro.

→ Hacer Actividad 4.6

4.8 CRON

Cron permite planificar tareas para ser ejecutadas en un momento específico. A estas tareas se les suele llamar “Cron Jobs”. Suele ser usado típicamente para tareas de mantenimiento como hacer backups periódicos, eliminar ficheros de log que ya no son necesarios, ejecutar tareas de mantenimiento, etc...

El formato típico de un Cron Job es:

```
Minute(0-59) Hour(0-24) Day_of_month(1-31) Month(1-12) Day_of_week(0-6) Command_to_execute
```

```
# min (0 - 59)
#   hour (0 - 23)
#     day of month (1 - 31)
#       month (1 - 12)
#         day of week (0 - 6) (0 to 6 are Sunday to
#           Saturday, or use names; 7 is also Sunday)
#
#           command to execute
```

Vamos a probarlo, para ello vamos a hacer que nuestro comando sea escribir “hola” en un fichero de texto. Para ello, primero creamos el fichero de texto en nuestro home:

```
$ touch bar.txt
```

El comando que vamos a utilizar va a ser echo y vamos a hacer un append (>>) del echo religiéndolo al comando:

```
$ echo hola >> /home/ico/bar.txt
```

Para ver cómo va escribiendo en el fichero, podemos usar:

```
$ cat /home/ico/bar.txt
```

```
ico@ico-virtual-machine:~$ touch bar.txt
ico@ico-virtual-machine:~$ echo hola >> /home/ico/bar.txt
ico@ico-virtual-machine:~$ cat /home/ico/bar.txt
```

Ahora que ya conocemos cómo funciona el comando echo, vamos a hacer que haga el comando `echo hola >> /home/ico/bar.txt` cada cierto tiempo.

Mostrar la crontab del usuario actual:

Crontab es un simple archivo de texto que guarda una lista de comandos a ejecutar en un tiempo especificado por el usuario. Crontab verificará la fecha y hora en que se debe ejecutar el script o el comando, los permisos de ejecución y lo realizará en el background.

```
$ crontab -l
```

Como es la primera vez, no hay crontab que mostrar y hay que crearla.

```
no crontab for aitor - using an empty one
```

```
Select an editor. To change later, run 'select-editor'.
```

1. /bin/nano <---- easiest
2. /usr/bin/vim.basic
3. /usr/bin/vim.tiny
4. /bin/ed

```
Choose 1-4 [1]:
```

Elegimos nano

Crear una crontab

```
$ crontab -e
```

The screenshot shows a terminal window titled "aitor@aitor-virtual-machine: ~". The title bar also displays "File Edit View Search Terminal Help" and "GNU nano 2.9.3 /tmp/crontab.uzh84C/crontab". The main area of the terminal contains the crontab configuration file. It includes comments explaining the syntax and fields, and an example backup command. A status bar at the bottom shows keyboard shortcuts for various operations like Get Help, Write Out, Where Is, Cut Text, Justify, Exit, Read File, Replace, Uncut Text, and To Spell. A message "[Read 22 lines]" is displayed above the status bar.

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
[ Read 22 lines ]
^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text    ^J Justify
^X Exit         ^R Read File   ^\ Replace      ^U Uncut Text  ^T To Spell
```

A continuación, vamos a escribir nuestro comando en nuestro crontab.

The screenshot shows a terminal window titled "ico@ico-virtual-machine: ~". The title bar also displays "File Edit View Search Terminal Help" and "GNU nano 4.8 /tmp/crontab.erHxc4/crontab". The main area of the terminal contains the crontab configuration file, identical to the one in the previous screenshot, but with an additional line added at the bottom: "* * * * echo hola >> /home/ico/bar.txt". This line specifies that the command "echo hola" should be run every minute. A status bar at the bottom shows keyboard shortcuts for various operations like Get Help, Write Out, Where Is, Cut Text, Justify, Exit, Read File, Replace, Uncut Text, and To Spell.

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * echo hola >> /home/ico/bar.txt
```

He elegido que se ejecute cada minuto.

Para comprobar que se está realizando...

```
ico@ico-virtual-machine:~$ cat /home/ico/bar.txt
hola
hola
hola
hola
hola
```

Prueba a cambiar la periodicidad, te puede ayudar la información siguiente.

1. Para ejecutar un cronjob cada minuto

```
* * * * * <command-to-execute>
```

2. Cada 5 minutos

```
*/5 * * * * <command-to-execute>
```

3. Cada 15 minutos

```
*/15 * * * * <command-to-execute>
```

4. También se pueden definir múltiples intervalos de tiempo separados por comas. Por ejemplo, si queremos ejecutar el cron job en los minutos 0, 5 y 10 de cada hora:

```
0,5,10 * * * * <command-to-execute>
```

5. Cada hora en el minuto 0:

```
0 * * * * <command-to-execute>
```

6. Cada dos horas:

```
0 */2 * * * <command-to-execute>
```

7. Cada día:

```
0 0 * * * <command-to-execute>
```

8. Cada día a las 3 am

```
0 3 * * * <command-to-execute>
```

9. Los domingos:

```
0 0 * * SUN <command-to-execute>
```

O

```
0 0 * * 0 <command-to-execute>
```

10. Los días laborables:

```
0 0 * * 1-5 <command-to-execute>
```

12. Cada mes:

```
0 0 1 * * <command-to-execute>
```

13. A las 16:15 el primer día del mes:

```
15 16 1 * * <command-to-execute>
```

También se pueden usar los siguientes strings para definir periodos específicos:

@reboot	Run once, at startup.
@yearly	Run once a year.
@annually	(same as @yearly).
@monthly	Run once a month.
@weekly	Run once a week.
@daily	Run once a day.
@midnight	(same as @daily).
@hourly	Run once an hour.

Por ejemplo, para ejecutar un cron job cada vez que el ordenador se reinicia:

```
@reboot <command-to-execute>
```

Finalmente, si queremos borrar todos los cron jobs del usuario:

```
$ crontab -r
```

Si no queréis acordaros de esto, hay editores que dejan configurarlo de manera fácil:

<https://crontab-generator.org/>

Y páginas para probar lo que harían la configuración:

<https://crontab.guru/>

```
ico@ico-virtual-machine:~$ curl -v https://www.facebook.com
*   Trying 31.13.83.36:443...
* TCP_NODELAY set
* Connected to www.facebook.com (31.13.83.36) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crt
*   CPath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_CHACHA20_POLY1305_SHA256
* ALPN, server accepted to use h2
* Server certificate:
*   subject: C=US; ST=California; L=Menlo Park; O=Facebook, Inc.; CN=*.facebook.com
*   start date: Sep 11 00:00:00 2020 GMT
*   expire date: Dec 10 12:00:00 2020 GMT
*   subjectAltName: host "www.facebook.com" matched cert's "*.facebook.com"
*   issuer: C=US; O=DigiCert Inc; OU=www.digicert.com; CN=DigiCert SHA2 High Assurance Server CA
*   SSL certificate verify ok.
```

4.9 GREP

Grep (global regular expression print) permite buscar expresiones regulares en texto.

Creamos un fichero test_grep.txt con el siguiente contenido con el editor nano:

```
nano test_grep.txt
```

The screenshot shows a terminal window with the title bar 'ico@ico-virtual-machine: ~'. Below it, the nano editor interface is visible. The status bar at the top right says 'test_grep.txt'. The main area of the editor contains the following text:

```
GNU nano 4.8
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
123 456
esto es una prueba de grep
```

Para comprobar que se ha guardado correctamente:

The screenshot shows a terminal window with the title bar 'ico@ico-virtual-machine: ~\$'. Below it, the command 'cat test_grep.txt' has been run, and its output is displayed:

```
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
123 456
esto es una prueba de grep
```

Usando grep, podemos hacer búsquedas en el fichero. Podemos buscar un string:

```
grep ordenador test_grep.txt
```

```
ico@ico-virtual-machine:~$ grep ordenador test_grep.txt  
ordenador
```

Lo que hará una búsqueda y devolverá los resultados exactos que coincidan con el string, teniendo en cuenta mayúsculas y minúsculas. Con el parámetro -i podemos ignorar las mayúsculas solo buscando coincidencia de letras:

```
ico@ico-virtual-machine:~$ grep -i ordenador test_grep.txt  
ordenador  
Ordenador
```

Con el parámetro -w podemos buscar por palabras completas:

```
ico@ico-virtual-machine:~$ grep -w es test_grep.txt  
esto es una prueba de grep  
ico@ico-virtual-machine:~$ grep es test_grep.txt  
esto es una prueba de grep
```

Por ejemplo, sin usar -w la búsqueda también nos devuelve “estos”.

También es posible hacer búsquedas invertidas con el parámetro -v, es decir, que no devuelva el patrón que indicamos:

```
ico@ico-virtual-machine:~$ grep -v es test_grep.txt  
ordenador  
Ordenador  
o&denador  
linux  
linus  
unix  
foo bar  
FOO bar  
123 456
```

Con los parámetros -A y -B podemos indicar que nos muestre las X líneas siguientes o las X anteriores.

```
ico@ico-virtual-machine:~$ grep -i -A3 ordenador test_grep.txt  
ordenador  
Ordenador  
o&denador  
linux  
linus  
ico@ico-virtual-machine:~$ grep -i -B1 linus test_grep.txt  
linux  
linus
```

Con -C podemos hacer las dos cosas a la vez

```
ico@ico-virtual-machine:~$ grep -i -C1 linus test_grep.txt  
linux  
linus  
unix
```

Con -n podemos ver la línea de fichero donde se ha encontrado el patrón:

```
ico@ico-virtual-machine:~$ grep -n a test_grep.txt
1:ordenador
2:Ordenador
3:o&denador
7:foo bar
8:FOO bar
10:esto es una prueba de grep
```

Con `-r` podemos hacer una búsqueda recursiva en todos los ficheros que cuelguen de un directorio.

```
lco@lco-virtual-machine:~$ grep -r "Network" /var/log/*
grep: /var/log/boot.log: Permission denied
grep: /var/log/boot.log.1: Permission denied
grep: /var/log/boot.log.2: Permission denied
grep: /var/log/boot.log.3: Permission denied
grep: /var/log/boot.log.4: Permission denied
grep: /var/log/boot.log.5: Permission denied
grep: /var/log/boot.log.6: Permission denied
grep: /var/log/btmp: Permission denied
grep: /var/log/btmp.1: Permission denied
[...]
[ 2.535150] kernel: e1000: Intel(R) PRO/1000 Network Driver - version 7.3.21-k8-NAPI
[ 2.968104] kernel: e1000 0000:02:01.0 eth0: Intel(R) PRO/1000 Network Connection
[ 10.491688] kernel: audit: type=1400 audit(1603101519.849:7): apparmor="STATUS" operati
e_load" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-client.action" pid=647 comm="apparmor_
[ 10.491691] kernel: audit: type=1400 audit(1603101519.849:8): apparmor="STATUS" operati
```

Combinándolo con `-l`, nos devolverá los ficheros, sin mostrar el contenido:

```
ico@ico-virtual-machine:~$ grep -r -l "Network" /var/log/*
grep: /var/log/boot.log: Permission denied
grep: /var/log/boot.log.1: Permission denied
grep: /var/log/boot.log.2: Permission denied
grep: /var/log/boot.log.3: Permission denied
grep: /var/log/boot.log.4: Permission denied
grep: /var/log/boot.log.5: Permission denied
grep: /var/log/boot.log.6: Permission denied
grep: /var/log/btmp: Permission denied
grep: /var/log/btmp.1: Permission denied
/var/log/dmesg
/var/log/dmesg.0
grep: /var/log/gdm3: Permission denied
grep: /var/log/installer/syslog: Permission denied
grep: /var/log/installer/version: Permission denied
grep: /var/log/installer/casper.log: Permission denied
grep: /var/log/installer/debug: Permission denied
grep: /var/log/installer/partman: Permission denied
/var/log/journal/bfe7b4f43ae347559766a74f3c4c6557/system@c9d0870b2b9
22e511...[redacted]
```

Si lo combinamos con -L nos mostrará los ficheros que **no** contengan el patrón:

```
ico@ico-virtual-machine:~$ grep -r -L "Network" /var/log/*
/var/log/alternatives.log
/var/log/alternatives.log.1
/var/log/apt/eipp.log.xz
/var/log/apt/term.log.1.gz
/var/log/apt/term.log
/var/log/apt/history.log
/var/log/apt/history.log.1.gz
/var/log/auth.log
/var/log/auth.log.1
/var/log/auth.log.2.gz
```

Las búsquedas con GREP no se limitan a usar simplemente strings, es posible usar expresiones regulares. Aprender a usar expresiones regulares queda fuera del alcance de esta introducción, pero vamos a ver unos ejemplos.

Por ejemplo, podemos usar el punto como comodín, sustituyendo un carácter:

```
ico@ico-virtual-machine:~$ grep 's.o' test_grep.txt
esto es una prueba de grep
```

Si quisiéramos usar el punto tendríamos que usar el backlash \:

```
ico@ico-virtual-machine:~$ grep 's\.o' test_grep.txt
ico@ico-virtual-machine:~$
```

Podemos usar ^ y \$ para buscar palabras que empiecen o acaben de cierta manera:

```
ico@ico-virtual-machine:~$ grep '^o' test_grep.txt
ordenador
o&denador
ico@ico-virtual-machine:~$ grep 'ar$' test_grep.txt
foo bar
FOO bar
```

Con las “character” clases podemos buscar rangos o grupos de letras. Por ejemplo, todas las palabras que tengan una letra minúscula:

```
ico@ico-virtual-machine:~$ grep '[a-z]' test_grep.txt
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
esto es una prueba de grep
```

Algunas de estas clases ya están definidas, [:lower:] sería la clase definida equivalente a [a-z].

```
ico@ico-virtual-machine:~$ grep '[[[:lower:]]]' test_grep.txt
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
esto es una prueba de grep
```

Algunas de las clases definidas son: [:lower:] y [:upper:] para caracteres en minúsculas y mayúsculas respectivamente. [:alpha:], que sería la suma de [:lower:] y [:upper:] (es decir, cualquier letra):

```
ico@ico-virtual-machine:~$ grep '[[[:alpha:]]]' test_grep.txt
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
esto es una prueba de grep
```

[:digit:] para los números:

```
ico@ico-virtual-machine:~$ grep '[[[:digit:]]]' test_grep.txt
123 456
```

Y [:alnum:] que sería la suma de [:alpha:] y [:digit:].

```
ico@ico-virtual-machine:~$ grep '[[[:alnum:]]]' test_grep.txt
ordenador
Ordenador
o&denador
linux
linus
unix
foo bar
FOO bar
123 456
esto es una prueba de grep
```

Los brackets podemos combinarlos con strings para buscar palabras que formen un patrón:

```
ico@ico-virtual-machine:~$ grep 'linu[sx]' test_grep.txt
linux
linus
```