

PROMETEO

Unidad 2





2

Linux: fundamentos y comandos básicos

PROMETEO

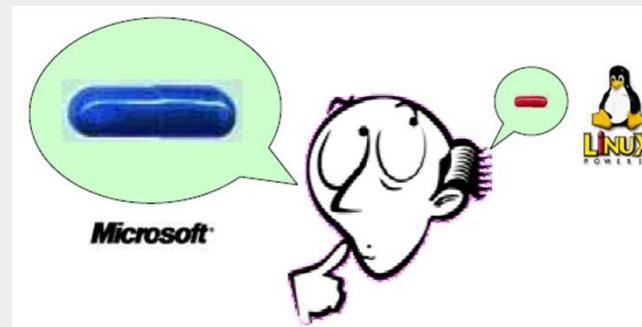
Sesión 4

Kernel y estructura de Linux. Diferencias con
Windows

PROMETEO

¿Por qué Linux?

- “Esta es tu última oportunidad. Despues de esto no hay vuelta atrás.
- Si tomas la pastilla azul la historia termina. Despertarás de tu cama y creerás lo que quieras creer.
- Si tomas la pastilla roja estarás en el País de las Maravillas y te enseñaré cómo de profunda es la madriguera del conejo. Recuerda que todo lo que te estoy ofreciendo es la verdad. Nada más”



Falsos mitos

- Linux es un sistema complicado de usar.
- Linux es un SO solo para desarrolladores.
- Linux tiene poca compatibilidad con el resto de SO.
- Linux es un sistema que carece de aplicaciones cotidianas.
- En definitiva...
- ¿LINUX ESTÁ DESTINADO SOLO PARA EXPERTOS?

Linux, Ubuntu, CentOS, Fedora, ¿qué es cada cosa?

- Cuando se habla de Linux, se habla del NÚCLEO del SO. Normalmente viene dada por un número de versión (xx.yy.zz, por ejemplo 2.4.20).
- Cuando nos instalamos algo referente a Linux, lo que realmente nos estamos instalando es una DISTRIBUCIÓN de Linux que incluye:
 - El núcleo del SO (Linux).
 - Paquetes de aplicaciones (Xgree, apt-get, RPM Package Manager, etc).
- En nuestra querida Wikipedia podéis encontrar las decenas de distribuciones Linux que existen en la actualidad:
https://es.wikipedia.org/wiki/Distribuci%C3%B3n_Linux

Diferencias entre distribuciones

- Genéricas: [Ubuntu](#), [openSUSE](#), [elementaryOS](#), [Debian](#), [Arch Linux](#), [Chrome OS](#), [Linux Mint](#), [Fedora](#), [CentOS](#).
- Básicas: [Lubuntu](#), [Sparky](#), [Bodhi](#).
- Empresas: [SUSE Linux Enterprise Server](#), [Red Hat Enterprise Linux](#).
- Seguridad/Privacidad: [Whonix](#), [Tails](#), [Santoku](#), [DEFT](#), Kali.
- Juegos: [SteamOS](#), [OzonOS](#).
- Accesibilidad (Discapacidad). SONAR, Vinus.
- Otros. Sugar (niños), [Snappy Ubuntu Core](#) (Raspberry Pi).



Licencias

- Software libre: Permite el estudio y modificación del programa para libre distribución y beneficio de los usuarios.
- Software de código abierto: Software en la que los derechos de autor son exclusivos, pero se permite a los usuarios utilizar, cambiar, mejorar y redistribuir el software.
- Kernel de Linux = Software libre y de código abierto.
- Licencias GNU/Linux: Estas licencias deben ser aprobadas por la Open Source Initiative (OSI) y tener denominación GPL.
 - GPL = Licencia que garantiza a los usuarios finales la libertad de usar y modificar el software
- Vídeo explicación: <https://youtu.be/UUJ0dFpj1-M>

Entornos de escritorio en Linux. Unity

Entorno para usuarios novatos (similar a MacOS X y Windows)



PROMETEO

Entornos de escritorio en Linux. Gnome

Entorno modular y ampliable gracias a sus extensiones



PROMETEO

Entornos de escritorio en Linux. KDE

Muy similar a Windows



PROMETEO

Entornos de escritorio en Linux. LXDE

Ofrece una interfaz totalmente básica para el ahorro de recursos.



PROMETEO

Entornos de escritorio en Linux. XFCE

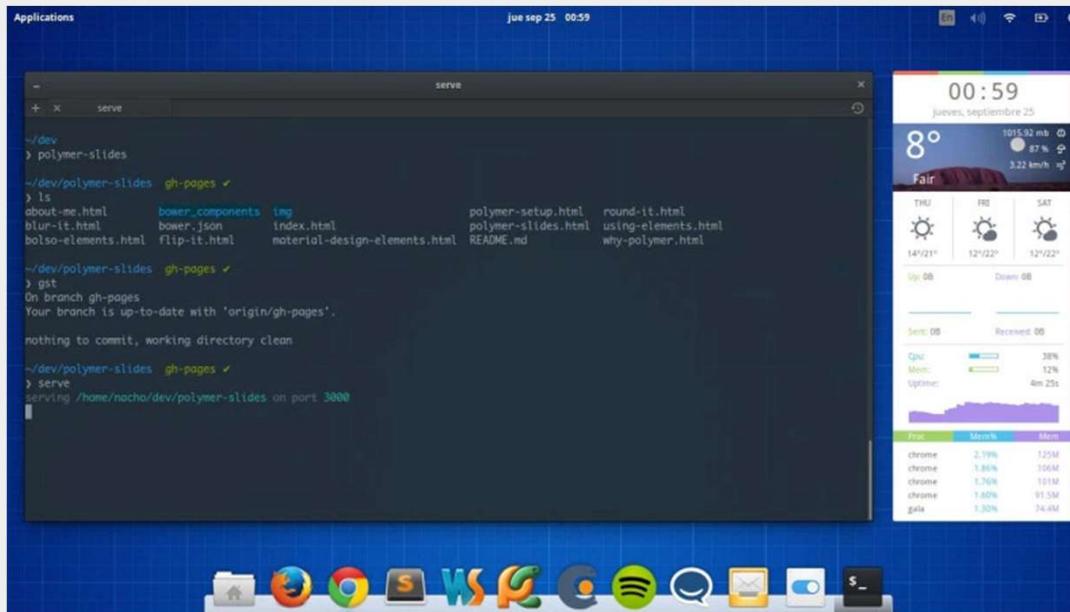
Interfaz para ordenadores con pocos recursos



PROMETEO

Entornos de escritorio en Linux. Pantheon

Similar a MacOS X (Para los frikis de Linux)



PROMETEO

Paquetes en Linux

- Los paquetes en Linux son ficheros comprimidos usados para la construcción de una aplicación en el sistema.
- En otras palabras, es como si comprimes una aplicación al completo y la subes a Internet para que pueda ser utilizada directamente por cualquier usuario. Hay distintos tipos de paquetes:
- DEB (**Ubuntu**, Kubuntu, ZorinOS, Linux Mint). Contiene ejecutables, archivos de configuración, información, copyright y documentación.
- RPM (RedHat y variantes, como por ejemplo, Fedora y openSUSE). Muy potente ante actualizaciones, ya que no hace falta llevarnos toda la aplicación, sino la correspondiente actualización con lo que conlleva reducir de manera drástica la capacidad del paquete.
- TGZ. Paquete con ficheros de código fuente, por lo que el usuario tendrá que complicar estos ficheros.
- Ebuild (Gentoo). Scripts bash.

Ubuntu



ubuntu

PROMETEO

Ubuntu. Concepto y versiones

Ubuntu es una distribución Linux enfocada en usuarios nuevos e intermedios con una gran facilidad de uso.

- La **interfaz** predeterminada de Ubuntu es **Unity** y utiliza en conjunto las aplicaciones de **Gnome**.
- Los paquetes que pueden ser utilizados para esta distribución es **.DEB**.

Principales versiones:

- **Ubuntu Desktop**. Orientado a escritorio.
- **Ubuntu Server**. Orientado a servidores.
- **Ubuntu Phone**. Orientado a smartphones (en desuso).

<https://commons.wikimedia.org/wiki/File:UbuntuFamilyTree1210.svg>

Ubuntu. Principales ventajas

- **Community.** Ubuntu dispone de una **comunidad de desarrolladores**, las cuales pueden escribir código, solución de bugs (vulnerabilidades), e incluso proposición de mejoras e ideas para próximas versiones.
- **Compatibilidad de aplicaciones.** La interfaz Unity permite la **compatibilidad de aplicaciones de otros tipos de interfaces**, como [Gnome](#) (Es un entorno de escritorio e infraestructura de desarrollo para sistemas operativos GNU/Linux, Unix y derivados Unix como BSD o Solaris; compuesto enteramente de software libre.)
- **Facilidad de uso.** Además de disponer de una interfaz muy sencilla en su uso, dispone del Centro de software Ubuntu para la instalación/desinstalación en red de aplicaciones.
- **Seguridad.** Por defecto no incluye un cortafuegos en el sistema debido a que en la propia instalación del SO, no instala servicios que puedan atentar a la seguridad del propio sistema. Además, la potencia de sudo (superusuario) hacen que las distribuciones de Linux sean totalmente estables.

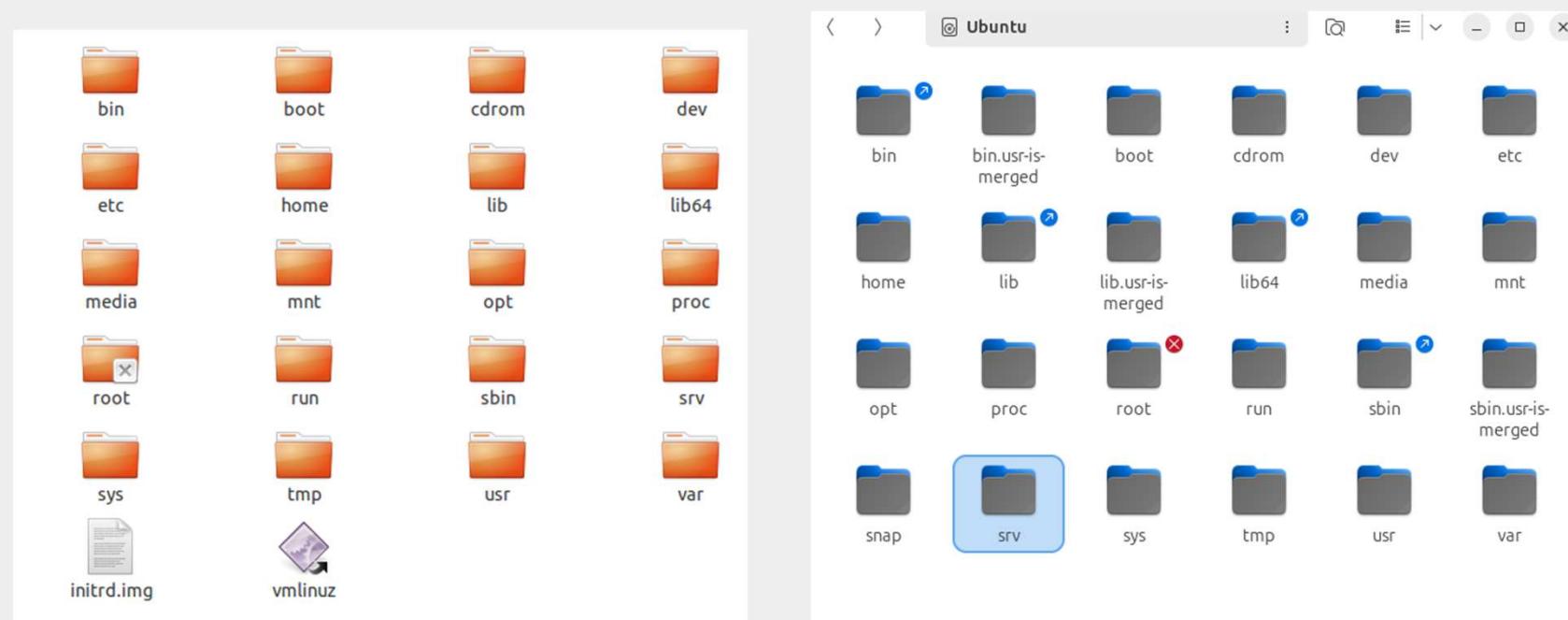
Ubuntu. Requisitos de instalación

Los requisitos varían según la versión de Ubuntu que se vaya a instalar. Para una distribución Ubuntu Server 12.04 LTS, los requisitos recomendados son:

- Procesador de 32 bits a 700 MHz.
- Memoria RAM de 512 MB.
- Disco duro con 5 GB libres.
- Tarjeta gráfica y monitor para soportar resolución de 1024x768.

¿Notáis diferencia con respecto a los requisitos recomendados de cualquiera de los SO de Windows? ;)

Sistema de archivos



Gestión de archivos y dispositivos

- Directorio **/bin**. Contiene todas las órdenes (programas) esenciales de la máquina.
- Directorio **/sbin**. Órdenes o programas para la administración del sistema.
- Directorio **/lib**. Bibliotecas esenciales de la máquina.
- Directorio **/boot**. Información sobre el arranque del sistema.
- Directorio **/etc**. Ficheros de configuración del sistema.
- Directorio **/proc**. Directorio virtual con información sobre programas que se ejecutan, hardware del sistema.
- Directorio **/tmp**. Ficheros temporales.
- Directorio **/var**. Directorio con información que varía con el tiempo (e.j. buzones de correos).

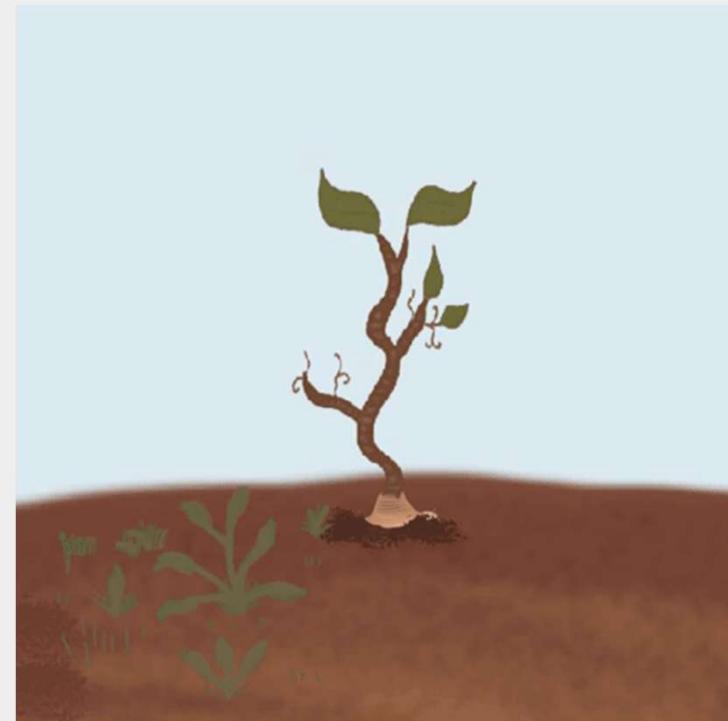
Gestión de archivos y dispositivos

- Directorio **/home**. Carpetas personales de los usuarios.
- Directorio **/dev**. Ficheros de dispositivo.
- Directorio **/mnt**. Carpeta para el montaje de dispositivos o recursos locales o remotos.
- Directorio **/usr**. Programas esenciales de la máquina (se utiliza para redes de ordenadores).
- Directorio **/usr/local**. Contiene todos los programas que se han instalado que no venían con la distribución original.

Aspectos importantes

- Antes de comenzar,
¿qué es **root**?

¿Una raíz?



ROOT

- La cuenta root o de superusuario es la cuenta con los permisos más elevados → similar al Administrador en Windows.
- **Este usuario tiene permisos de lectura, escritura y ejecución** de cualquier aplicación del sistema (acceso administrativo en general), a diferencia de los usuarios normales que tienen acceso limitado a ciertas tareas por razones obvias de seguridad.
- Una instrucción dada con permisos de superusuario puede ser muy útil cuando se usa correctamente, pero terriblemente devastadora si no se tiene el conocimiento preciso de lo que se está haciendo y de las consecuencias que pueden tener nuestras acciones.

ROOT

- Para hacer algo con privilegios de superusuario siempre se utiliza el comando **SU**.
- Esto hace que, al menos psicológicamente hablando, tengamos la noción de que estamos haciendo algo más serio con nuestro sistema.
- SU, del inglés **Substitute User** (cambiar usuario), se utiliza principalmente para cambiar de usuario en un terminal. Generalmente de un usuario normal a root, sin tener que cerrar sesión e iniciar de nuevo.

ROOT

- Para iniciar sesión como superusuario hay dos formas:
- Escribiendo en la terminal: **su**.
- O bien, escribiendo el comando SU seguido del nombre de usuario con el que queremos ingresar: **su [nombre del usuario]**.
- En ambos casos se pedirá de inmediato la contraseña del usuario solicitado y podremos trabajar como si fuéramos ese usuario. Esto es útil para administrar bases de datos, servidores web y otros servicios que disponen de usuarios específicos para realizar diferentes tareas de administración.

ROOT

- En algunas distribuciones de Linux (**Ubuntu**, Kubuntu) la cuenta de superusuario viene desactivada por defecto.
- En estos casos, la cuenta de usuario que creamos al instalar la distribución no es la misma que la cuenta de root o superusuario, sino que **p pertenece al grupo de administradores**.
- Para obtener privilegios de root se utiliza el comando **SUDO**.

ROOT

SUDO permite ejecutar comandos como otro usuario (**incluso como root**) pidiendo **nuestra propia contraseña** en lugar de la del usuario requerido.

La ventaja de utilizar SUDO es que **sólo se ejecuta el comando solicitado simulando ser otro usuario, sin cambiar verdaderamente el usuario actual.**

En el terminal, escribe SUDO justo antes del comando que quieras ejecutar, por ejemplo:

```
sudo apt-get install {nombre-del-paquete}
```

ROOT

- Después de ingresar tu contraseña **tendrás 15 minutos** para seguir utilizando la cuenta como superusuario.
- Pasado este tiempo volverás a tener los privilegios de tu cuenta de usuario normal.
- Pero si queremos tener privilegios de superusuario **de forma permanente** sin necesidad de ingresar SUDO varias veces podemos ejecutar en la terminal: **sudo su**



ROOT → sudo su .

- Antes de cerrar el terminal, es recomendable volver a nuestro propio usuario, esto se logra con un simple **exit**.
- Para saber con qué usuario estás logueado, basta con observar el símbolo de la línea de comandos.
- **usuario**@usuario-desktop:~**\$** Es la cuenta del **usuario normal (\$)**
- **root**@usuario-desktop:~**#** Es la cuenta de **root (#)**

Cuentas de usuario (I)

- Por tanto: **Únicamente se utiliza root para tareas de administración.**

- **Fichero /etc/passwd:** Define los **usuarios** existentes en el sistema:

- UID: Es el identificador del usuario.
- GID: Es el identificador del grupo.

```
nombre:contraseña:UID:GID:información:directorio_personal:shell
```

- Nombre – nombre del usuario
- Contraseña – antiguamente aparecía la contraseña encriptada del usuario, pero actualmente este campo aparece con una “x”
- UID – identificador de usuario
- GID – identificador de grupo
- Información – información extra del usuario (puede estar en blanco)
- Directorio_personal – directorio personal del usuario
- Shell – shell por defecto que se ejecuta cuando el usuario inicia la sesión

- ¿Cómo vemos el contenido de un fichero en Linux?

- **Hay varias formas, por ahora lo haremos mediante el comando cat:**

cat /etc/passwd

¡ACUÉRDATE DE USAR EL TABULADOR!

Cuentas de usuario (III)

- Fichero **/etc/shadow**. Se almacenan las **contraseñas cifradas** y nos da información sobre caducidad y validez de la cuenta.

Campos separados por ":"

- Nombre de inicio de sesión
- Contraseña encriptada
- Días desde el 1 de enero de 1970 en el la contraseña fue cambiada por última vez
- Días antes de que la contraseña **puede** ser cambiada
- Días después de los cuales la contraseña **debe** de ser cambiada
- Días de antelación antes de la expiración de la contraseña en los que el usuario será avisado
- Días desde que la contraseña expira hasta que se deshabilita la cuenta
- Días desde el 1 de enero de 1970 en el la contraseña expiró
- Campo reservado

Variables de entorno

- “Las variables de entorno forman un conjunto de valores **dinámicos** que normalmente afectan al comportamiento de los procesos en una computadora.” – WIKIPEDIA.
- Una **variable de entorno** es, para que todos podamos entenderlo, **una palabra, un texto fácilmente recordable que nos permitirá acceder a rutas más complejas de forma mucho más sencilla**.
- Para que podamos hacernos una idea, mi equipo dispone de una variable de entorno llamada Path que hace referencia a la ruta:
`C:/ProgramData/Oracle/Java/javapath;%SystemRoot%/system32;%SystemRoot%;`
`%SystemRoot%/System32/Wbem;%SYSTEMROOT%/System32/WindowsPowerShell/`
`v1.0;/C:/Program Files (x86)/ATI Technologies/ATI.ACE/Core-Static` por lo que es fácilmente comprensible que es preferible trabajar con una única palabra que no con esa enorme ruta.

Variables de entorno

- Por tanto, podemos decir que son contenedores de parámetros del sistema.
- **Acceder al valor de una variable:** \$nombre_variable

```
/> echo $SHELL  
/bin/bash
```

- Establecer una variable de entorno:
 - **export VARIABLE=valor**

Ayuda en línea

- Linux proporciona páginas de manuales de ayuda en línea para describir comandos y aplicaciones del sistema con bastante detalle.



Formas de obtener información acerca de un comando:

- `man [sección] <orden>`
- P.e. `man group`
- P.e. `man 5 group`
- Orden `info`
- Opción `-h` ó `-help` en las órdenes (p.e. `passwd -help`)
- Ayuda en formato HTML

Ayuda en línea

```
root@usuario-VirtualBox:/home/usuario# man cat
```

CAT(1) User Commands CAT(1)

NAME
cat - concatenate files and print on the standard output

SYNOPSIS
cat [OPTION]... [FILE]...

DESCRIPTION
Manual page cat(1) line 1 (press h for help or q to quit)


```
root@usuario-VirtualBox:/home/usuario# info cat
```

Next: tac invocation, Up: Output of entire files
3.1 'cat': Concatenate and write files
=====

'cat' copies each FILE ('-' means standard input), or standard input if none are given, to standard output. Synopsis:

-----Info: (coreutils)cat invocation, 72 lines --Top-----
Esto es Info, versión 6.5. Teclee H para ayuda, h para cursillo.


```
root@usuario-VirtualBox:/home/usuario# cat --help
```

Modo de empleo: cat [OPCIÓN]... [FICHERO]...
Concatenate FILE(s) to standard output.

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

-A, --show-all equivalent to -vET
-b, --number-nonblank number nonempty output lines, overrides -n
-e equivalent to -vE
-E, --show-ends display \$ at end of each line

Órdenes (Ejecución multitarea)

PATH

El PATH informa al shell (en la mayoría de los casos BASH) **dónde se encuentran los programas binarios** que podemos ejecutar en el sistema, sin tener que llamarlos por su ruta absoluta.

```
/> echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:,
```

Por ejemplo, el programa “ls” está en /usr/bin/ls algo que puedo saber gracias al comando **whereis**:

```
$ whereis ls  
root@usuario-VirtualBox:/home/usuario# whereis ls  
ls: /bin/ls /usr/share/man/man1/ls.1.gz
```



(me informa dónde reside el binario “ls” así como sus páginas **man**)

Órdenes (Ejecución multitarea)

PATH

Esto significa que puedo llamar al comando “ls” de manera absoluta así:

```
$ /usr/bin/ls (responderá de manera adecuada)
```

pero si llamo a “ls” directamente:

```
$ ls
```

responderá igual, gracias a que tengo definida su ruta en el PATH, como puedo ver así:

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

Ejecución en segundo plano (I)

La ejecución en segundo plano permite ejecutar un proceso (programa en ejecución con sus propios recursos) que seguirá ejecutándose incluso después de cerrar la visualización del mismo.

```
/> gcc programa.c &  
[1] 21457
```

De esta manera se envía el proceso a una ejecución en segundo plano

- El proceso o programa se ejecuta pero el intérprete de órdenes vuelve a tomar el control
- [1] → número de tarea que se asigna
 - Solo se dan a tareas que se ejecutan en segundo plano
- 21457 → PID (identificador de proceso)

Si ejecutamos el siguiente comando, **terminando con un ampersand (“&”)**, lo lanzaremos en segundo plano:

```
find / -iname "*.sh" >  
listadescritps.txt &
```

Lo podemos comprobar con el comando **jobs**

Ejecución en segundo plano (II)

Para visualizar el estado en el que se encuentran los procesos (tanto los de segundo plano como los ejecutados de manera normal)

Orden “ps”

- Sintaxis → `ps [opciones]`
- Opciones
 - -A → selecciona todos los procesos
 - a → selecciona todos los procesos de un terminal, incluidos los de otros usuarios
 - x → información específica sobre la ejecución del proceso

```
root@usuario-VirtualBox:/home/usuario# ps
 PID TTY      TIME CMD
 3111 pts/0    00:00:00 sudo
 3306 pts/0    00:00:00 su
 3325 pts/0    00:00:00 bash
17080 pts/0    00:00:00 ps
```

```
root@usuario-VirtualBox:/home/usuario# ps -a
 PID TTY      TIME CMD
 964 tty1    00:00:00 gnome-session-b
 973 tty1    00:00:05 gnome-shell
1220 tty1    00:00:00 Xwayland
1252 tty1    00:00:00 ibus-daemon
1255 tty1    00:00:00 ibus-dconf
1258 tty1    00:00:00 ibus-x11
1270 tty1    00:00:00 gsd-xsettings
1278 tty1    00:00:00 gsd-a11y-settin
1279 tty1    00:00:00 gsd-clipboard
1285 tty1    00:00:00 gsd-color
1292 tty1    00:00:00 gsd-datetime
1293 tty1    00:00:00 gsd-housekeepin
1300 tty1    00:00:00 gsd-keyboard
```

Ejecución en segundo plano (III)

Para eliminar un proceso en ejecución se utiliza la orden “kill” más el **PID**.

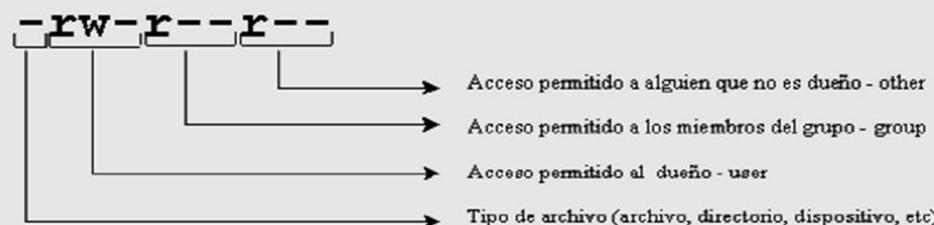
```
/> kill %1  
/> kill 21457
```

% 1 es una forma (¡de muchas!) de especificar a qué trabajo desea enviar la señal.
% 1 se refiere al trabajo en la parte superior de la pila de trabajos en segundo plano.

Propiedad de ficheros y permisos

Los ficheros y directorios están implementados en forma de propiedades y permisos.

- *Lectura*
 - Ver el contenido del fichero
 - Listar el contenido de un directorio
- *Escritura*
 - Modificar el contenido de un fichero
 - Añadir o borrar ficheros o directorios en un directorio
- *Ejecución*
 - Si es un programa, el usuario puede ejecutarlo
 - Si es un directorio el usuario puede consultar la información de los ficheros que contiene



¿A quién se puede otorgar permisos?

Los permisos solamente pueden ser otorgados a tres tipos o grupos de usuarios:

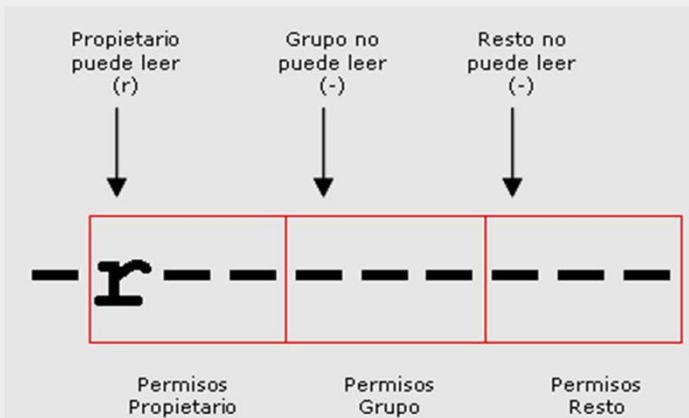
- Al **usuario** propietario del archivo
- Al **grupo** propietario del archivo
- Al **resto** de usuarios del sistema (**todos menos el propietario**)

Se pueden dar permisos de lectura, escritura, ejecución ó **combinación** de ambos **al usuario** propietario del archivo, **al grupo** propietario del archivo o **al resto** de usuarios del sistema.

En Unix no existe la posibilidad de asignar permisos a usuarios concretos ni a grupos concretos, tan solo se puede asignar permisos al usuario propietario, al grupo propietario o al resto de usuarios.

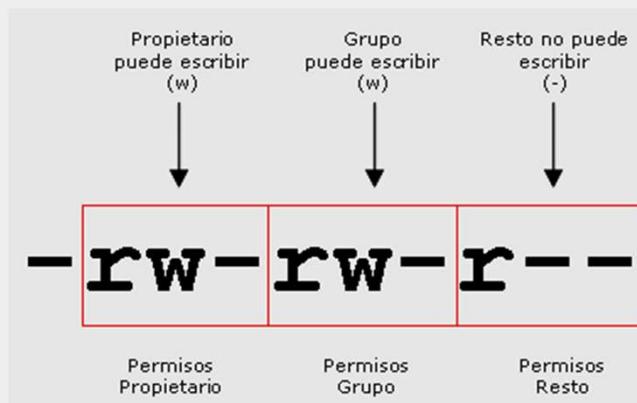
Permiso de lectura

- ☐ Cuando un usuario tiene permiso de **lectura** de un archivo significa que puede leerlo o visualizarlo, bien sea con una aplicación o mediante comandos.
- ☐ Cuando un usuario tiene permiso de lectura de una carpeta, significa que puede visualizar el contenido de la carpeta, es decir, puede ver los archivos y carpetas que contiene, bien sea con el comando 'ls' o con un explorador de archivos como Konqueror.
- ☐ El permiso de lectura se simboliza con la letra '**r**' del inglés 'read'.



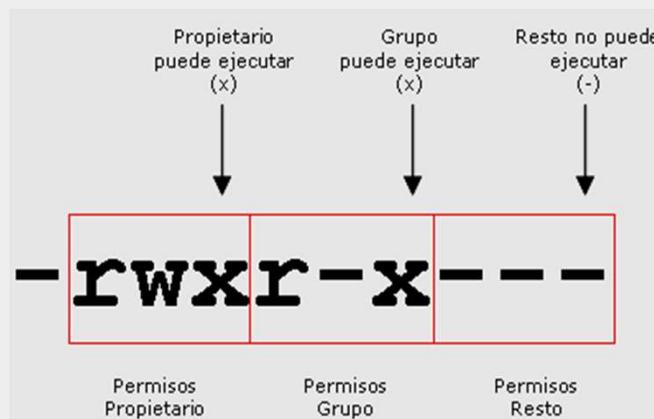
Permiso de escritura

- ☐ Cuando un usuario tiene permiso de **escritura** sobre un archivo significa que puede modificar su contenido, e incluso borrarlo. También le da derecho a cambiar los permisos del archivo mediante el comando **chmod** así como cambiar su propietario y el grupo propietario mediante el comando **chown**.
- ☐ Cuando un usuario tiene permiso de escritura sobre una carpeta, significa que puede modificar el contenido de la carpeta, es decir, puede crear y eliminar archivos y otras carpetas dentro de ella.
- ☐ El permiso de escritura se simboliza con la letra '**w**' del inglés '**write**'.



Permiso de ejecución

- ❑ Cuando un usuario tiene permiso de ejecución de un archivo significa que puede ejecutarlo.
- ❑ Los únicos archivos ejecutables son las aplicaciones y los archivos de comandos (scripts). Si tratamos de ejecutar un archivo no ejecutable, dará errores.
- ❑ Cuando un usuario tiene permiso de ejecución sobre una carpeta, significa que puede entrar en ella, bien sea con el comando 'cd' o con un explorador de archivos como Konqueror.
- ❑ El permiso de ejecución se simboliza con la letra '**x**' del inglés '**eXecute**'.



Permisos de archivo

- ❑ Con el comando **ls -l** podemos visualizar los permisos de los archivos o carpetas. Al ejecutar el comando aparecen todos los archivos, uno por línea.
- ❑ El primer carácter indica de qué tipo de archivo se trata.
 - Si es un guión '-' significa que se trata de un **archivo normal**
 - La letra '**d**' significa que se trata de una carpeta (**directory**)
 - La letra '**l**' significa que se trata de un enlace (**link**).
 - Otros valores son s, p, b que se refieren a sockets, tuberías (pipe) y dispositivos de bloque respectivamente.

Tipo de archivo:		
(-)	para archivos normales	
(d)	para carpetas (directory)	
(l)	para enlaces (link)	
(s)=(p)=tubería (pipe), (b)=dispositivo de bloque.		
	-rw-	r-----
Permisos Propietario	Permisos Grupo	Permisos Resto

Cambio de permisos

- Para cambiar los permisos de un archivo o una carpeta es necesario disponer del permiso de escritura (w) sobre dicho archivo o carpeta. Para hacerlo, se utiliza el comando **chmod**. La sintaxis del comando chmod es la siguiente:

```
chmod [opciones] permiso nombre_archivo_o_carpeta
```

- Los permisos se pueden representar de dos formas.

La primera es mediante las iniciales de a quién va dirigido el permiso (**usuario=u**, **grupo=g**, **resto=o** (other)), seguido de un signo **+** si se quiere **añadir** permiso o un signo **-** si se quiere **quitar** y seguido del tipo de permiso (**lectura=r**, **escritura=w** y **ejecución=x**).

Ejemplo: **chmod ug+rwx notas.txt**

Manipulación de permisos

A veces es necesario realizar algunos ajustes de forma manual en algunos ficheros determinados, sobretodo si estamos hablando de entornos de servidores.

Cambio de permisos → orden chmod

- Sintaxis → `chmod [opciones] modo fichero`
- Opciones más comunes
 - `-R` → cambia recursivamente los permisos
 - `-f` → no muestra mensajes de error
- Modo
 - Se puede especificar mediante `+/-<permiso>`
 - `/> chmod +rw prueba`
 - Añade permisos de lectura y escritura al usuario, grupo y otros
 - `/> chmod ug+rw prueba`
 - Añade permisos de lectura y escritura al usuario y grupo
 - Númeroicamente
 - `/> chmod 700 prueba`
 - Añade permisos de lectura, escritura y ejecución únicamente al usuario, al resto no le da ningún permiso



Permisos mediante un código numérico

- ❑ El código numérico está compuesto por **tres cifras** entre **0 y 7**. La primera de ellas representaría los permisos del usuario propietario, la segunda los del grupo propietario y la tercera los del resto de usuarios.
- ❑ En binario, las combinaciones representan el tipo de permisos. El bit más a la derecha (menos significativo) se refiere al permiso de ejecución (**1=activar y 0=desactivar**). El bit central se refiere al permiso de escritura y el bit más a la izquierda se refiere al permiso de lectura. La siguiente tabla muestra las 8 combinaciones posibles:

Cód Binario	Permisos efectivos		
	r	w	x
0 0 0	---		
1 0 1	--x		
2 0 1	-w-		
3 0 1	-wx		
4 1 0	r--		
5 1 0	r-x		
6 1 1	r w-		
7 1 1	r wx		

Permisos mediante un código numérico

- Si deseamos **otorgar** sólo permiso de **lectura**, el código a utilizar es el **4**.
- Si deseamos **otorgar** sólo permiso de **lectura y ejecución**, el código es el **5**.
- Si deseamos **otorgar** sólo permiso de **lectura y escritura**, el código es el **6**.
- Si deseamos **otorgar todos los permisos**, el código es el **7**.
- Si deseamos **quitar todos los permisos**, el código es el **0**.

```
// Dar todos los permisos al usuario y ninguno ni al grupo ni al resto  
chmod 700 examenSI.txt  
// Dar al usuario y al grupo permisos de lectura y ejecución y ninguno al resto  
chmod 550 examenSI.txt  
// Dar todos los permisos al usuario y lectura y ejecución al grupo y al resto  
chmod 755 /usr/bin/games/csgo  
// Dar todos los permisos al usuario y de lectura al resto, sobre todos los archivos  
chmod 744 *
```

Cód	Binario	Permisos efectivos
0	0 0 0	---
1	0 0 1	--x
2	0 1 0	-w-
3	0 1 1	-wx
4	1 0 0	r--
5	1 0 1	r-x
6	1 1 0	r w-
7	1 1 1	rwx

Cambio de propietarios

Es posible **cambiar el usuario y grupo propietario** tanto de un fichero como de un directorio.

Orden chown

- Sintaxis → `chown [opciones] usuario[:grupo] fichero`
- Opciones más usuales → las mismas que en la orden `chmod`
- Ejemplo
 - `>/ chown aitor.users prueba`
 - Establece como usuario y grupo propietarios del fichero `prueba` a `aitor` y `users` respectivamente

Ficheros de arranque

- En el directorio personal de cada usuario existen unos ficheros de arranque que en ocasiones son útiles para ciertas tareas como definir un nuevo PATH o exportar algunas variables de entorno.
- Los ficheros más importantes son:
 - `.bash_profile`: Se lanza en cada inicio de sesión que realiza el usuario.
 - `.bashrc`.

Sistemas de ficheros

Información extra

Olga M.Moreno

PROMETEO

Introducción

- Un **sistema de ficheros** es un dispositivo que se formatea para guardar información.
Linux soporta una gran cantidad de sistemas de ficheros.
- En Linux para poder acceder a cualquier sistema de ficheros es necesario **montarlo**.
- El lugar donde se montan se llama **punto de montaje** y es un directorio donde se va a proyectar el sistema de ficheros. Al desmontar el sistema de ficheros en el punto de montaje contendrá la información que tenía con anterioridad.

Montaje de sistemas de ficheros (I)

- Para acceder a cualquier sistema de ficheros es imprescindible que esté montado previamente.
- Se puede ver cuántos sistemas de ficheros hay montados y sus puntos de montaje con el comando **mount**.
- Montar cualquier sistema de ficheros requiere tener privilegios de root y especificar cierta información como el tipo de sistema de ficheros, dispositivo donde se encuentra y punto de montaje.

```
/> mount -t ext2 /dev/hda2 /mnt/server  
/> mount -t MSDOS /dev/fd0 /mnt/floppy
```

Montaje de sistemas de ficheros (II)

Para desmontar un sistema de ficheros se utiliza la orden **umount**

```
/> umount /mnt/floppy
```

- Hay que especificar o el dispositivo o el punto de montaje

Durante el arranque del sistema se montan varios sistemas de ficheros

- Especificado en el fichero **/etc/fstab**
- Formato

```
dispositivo punto_de_montaje tipo parámetros
```

Para montar cualquier sistema de ficheros que venga especificado en **/etc/fstab** basta con

```
/> mount <punto_de_montaje>
/> mount /mnt/floppy
```

La opción **-a** de **mount** provoca el montaje de todos los sistemas de ficheros especificados en **/etc/fstab**

```
/> mount -a
```

- Los sistemas de ficheros que tengan especificado el parámetro **noauto** no serán montados

La opción de montaje **user**, permite a un usuario normal del sistema el montaje de un sistema de ficheros

Automontaje de dispositivos

El automontador es un añadido al núcleo de Linux que permite el montaje automático de un sistema de ficheros cuando se accede a él. Los ficheros que suelen estar implicados son:

- **/etc/auto.misc**
 - Archivo de mapas donde se especifica el punto de montaje, dispositivo y opciones de montaje
- **/etc/auto.master**
 - Lista los directorios bajo los cuales el automontador puede colgar los puntos de montaje
- **/etc/init.d/autofs**
 - Aplicación de montaje automático

Comprobación y reparación de sistemas de ficheros

Se utiliza la orden fsck

```
fsck -t tipo dispositivo
tipo → tipo del sistema de ficheros
dispositivo → dispositivo que se quiere
verificar o reparar
```

Sesión 5-6-7

Comandos básicos
para la administración de Linux

PROMETEO

Información del sistema

1. **arch**: mostrar la arquitectura de la máquina (1).
2. **uname -m**: mostrar la arquitectura de la máquina (2).
3. **uname -r**: mostrar la versión del kernel usado.
4. **dmidecode -q**: mostrar los componentes (hardware) del sistema.
5. ~~hdparm -i /dev/hda~~: mostrar las características de un disco duro.
6. ~~hdparm -T /dev/sda~~: realizar prueba de lectura en un disco duro.
7. **cat /proc/cpuinfo**: mostrar información de la CPU.
8. **cat /proc/interrupts**: mostrar las interrupciones.
9. **cat /proc/meminfo**: verificar el uso de memoria.
10. **cat /proc/swaps**: mostrar ficheros swap.
11. **cat /proc/version**: mostrar la versión del kernel.
12. **cat /proc/net/dev**: mostrar adaptadores de red y estadísticas.
13. **cat /proc/mounts**: mostrar el sistema de ficheros montado.
14. **lspci -tv**: mostrar los dispositivos PCI.
15. **lsusb -tv**: mostrar los dispositivos USB.
16. **date**: mostrar la fecha del sistema.
17. **cal 2011**: mostrar el almanaque de 2011.
18. **cal 07 2011**: mostrar el almanaque para el mes julio de 2011.
19. **date 041217002011.00**: colocar (declarar, ajustar) fecha y hora.
20. **clock -w**: guardar los cambios de fecha en la BIOS.

Actualizador de paquetes APT

1. **apt-get install package_name**: instalar / actualizar un paquete deb.
2. **apt-cdrom install package_name**: instalar / actualizar un paquete deb desde un cdrom.
3. **apt-get update**: actualizar la lista de paquetes.
4. **apt-get upgrade**: actualizar todos los paquetes instalados.
5. **apt-get remove package_name**: eliminar un paquete deb del sistema.
6. **apt-get check**: verificar la correcta resolución de las dependencias.
7. **apt-get clean**: limpiar cache desde los paquetes descargados.
8. **apt-cache search searched-package**: retorna lista de paquetes que corresponde a la serie «paquetes buscados».

Apagar, reiniciar sistema o cerrar sesión

1. **shutdown -h now**: apagar el sistema (1).
2. **init 0**: apagar el sistema (2).
3. **telinit 0**: apagar el sistema (3).
4. **halt**: apagar el sistema (4).
5. **shutdown -h hours:minutes &**: apagado planificado del sistema.
6. **shutdown -c**: cancelar un apagado planificado del sistema.
7. **shutdown -r now**: reiniciar (1).
8. **reboot**: reiniciar (2).
9. **logout**: cerrar sesión.

Archivos y directorios (I)

- **TODOS
IMPORTANTES!**

1. **cd /home**: entrar en el directorio "home".
2. **cd ..**: retroceder un nivel.
3. **cd ../../**: retroceder 2 niveles.
4. **cd**: ir al directorio raíz.
5. **cd ~user1**: ir al directorio user1.
6. **cd -**: ir (regresar) al directorio anterior.
7. **pwd**: mostrar el camino del directorio de trabajo.
8. **ls**: ver los ficheros de un directorio.
9. **ls -F**: ver los ficheros de un directorio.
10. **ls -l**: mostrar los detalles de ficheros y carpetas de un directorio.
11. **ls -a**: mostrar los ficheros ocultos.
12. **ls *[0-9]***: mostrar los ficheros y carpetas que contienen números.
13. **tree**: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(1)
14. **lmtree**: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(2)
15. **mkdir dir1**: crear una carpeta o directorio con nombre 'dir1'.
16. **mkdir dir1 dir2**: crear dos carpetas o directorios simultáneamente (Crear dos directorios a la vez).
17. **mkdir -p /tmp/dir1/dir2**: crear un árbol de directorios.
18. **rm -f file1**: borrar el fichero llamado 'file1'.
19. **rmdir dir1**: borrar la carpeta llamada 'dir1'.

Archivos y directorios (II)

20. **rm -rf dir1**: eliminar una carpeta llamada 'dir1' con su contenido de forma recursiva. (Si lo borro recursivo estoy diciendo que es con su contenido).
21. **rm -rf dir1 dir2**: borrar dos carpetas (directorios) con su contenido de forma recursiva.
22. **mv dir1 new_dir**: renombrar o mover un fichero o carpeta (directorio).
23. **cp file1**: copiar un fichero.
24. **cp file1 file2**: copiar dos ficheros al unísono.
25. **cp dir /* ..**: copiar todos los ficheros de un directorio dentro del directorio de trabajo actual.
26. **cp -a /tmp/dir1 ..**: copiar un directorio dentro del directorio actual de trabajo.
27. **cp -a dir1**: copiar un directorio.
28. **cp -a dir1 dir2**: copiar dos directorio al unísono.
29. **ln -s file1 lnk1**: crear un enlace simbólico al fichero o directorio.
30. **ln file1 lnk1**: crear un enlace físico al fichero o directorio.
31. **touch -t 0712250000 file1**: modificar el tiempo real (tiempo de creación) de un fichero o directorio.
32. **file file1**: salida (volcado en pantalla) del tipo mime de un fichero texto.
33. **iconv -l**: listas de códigos conocidos.
34. **iconv -f fromEncoding -t toEncoding inputFile > outputFile**: crea una nueva forma del fichero de entrada asumiendo que está codificado en fromEncoding y convirtiéndolo a ToEncoding.
35. **find . -maxdepth 1 -name *.jpg -print -exec convert "{}" -resize 80x60 "thumbs/{}" \;**: agrupar ficheros redimensionados en el directorio actual y enviarlos a directorios en vistas de miniaturas (requiere convertir desde Imagemagick).

Búsqueda de archivos

1. **find / -name file1**: buscar fichero y directorio a partir de la raíz del sistema.
2. **find / -user user1**: buscar ficheros y directorios pertenecientes al usuario 'user1'.
3. **find /home/user1 -name *.bin**: buscar ficheros con extensión '. bin' dentro del directorio '/ home/user1'.
4. **find /usr/bin -type f -atime +100**: buscar ficheros binarios no usados en los últimos 100 días.
5. **find /usr/bin -type f -mtime -10**: buscar ficheros creados o cambiados dentro de los últimos 10 días.
6. **find / -name *.rpm -exec chmod 755 '{}' \;**: buscar ficheros con extensión '.rpm' y modificar permisos.
7. **find / -xdev -name *.rpm**: Buscar ficheros con extensión '.rpm' ignorando los dispositivos removibles como cdrom, pen-drive, etc....
8. **locate *.ps**: encuentra ficheros con extensión '.ps' ejecutados primeramente con el command 'updatedb'.
9. **whereis halt**: mostrar la ubicación de un fichero binario, de ayuda o fuente. En este caso pregunta dónde está el comando 'halt'.
10. **which halt**: mostrar la senda completa (el camino completo) a un binario / ejecutable.

Espacio de disco

1. **df -h**: mostrar una lista de las particiones montadas.
2. **ls -lSr |more**: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
3. **du -sh dir1**: Estimar el espacio usado por el directorio 'dir1'.
4. **du -sk * | sort -rn**: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
5. **rpm -q -a -qf '%10{SIZE}t%{NAME}n' | sort -k1,1n**: mostrar el espacio usado por los paquetes rpm instalados organizados por tamaño (Fedora, Redhat y otros).
6. **dpkg-query -W -f='\${Installed-Size;10}t\${Package}n' | sort -k1,1n**: mostrar el espacio usado por los paquetes instalados, organizados por tamaño (Ubuntu, Debian y otros).

Usuarios y grupos

Siguiente diapositiva más completa

Asociar un usuario a un grupo existente:
usermod -G nombregrupo nombreusuario

Ver los grupos asociados a un usuario:
groups nombreusuario

1. **groupadd nombre_del_grupo**: crear un nuevo grupo.
2. **groupdel nombre_del_grupo**: borrar un grupo.
3. **groupmod -n nuevo_nombre_del_grupo viejo_nombre_del_grupo**: renombrar un grupo.
4. **useradd -c "Name Surname" -g admin -d /home/user1 -s /bin/bash user1**: Crear un nuevo usuario perteneciente al grupo "admin".
5. **useradd user1**: crear un nuevo usuario.
6. **userdel -r user1**: borrar un usuario ('-r' elimina el directorio Home).
7. **usermod -c "User FTP" -g system -d /ftp/user1 -s /bin/nologin user1**: cambiar los atributos del usuario.
8. **passwd**: cambiar contraseña.
9. **passwd user1**: cambiar la contraseña de un usuario (solamente por root).
10. **chage -E 2011-12-31 user1**: colocar un plazo para la contraseña del usuario. En este caso dice que la clave expira el 31 de diciembre de 2011.
11. **pwck**: chequear la sintaxis correcta el formato de fichero de '/etc/passwd' y la existencia de usuarios.
12. **grpck**: chequear la sintaxis correcta y el formato del fichero '/etc/group' y la existencia de grupos.
13. **newgrp group_name**: registra a un nuevo grupo para cambiar el grupo predeterminado de los ficheros creados recientemente.

Usuarios y grupos

Crear grupo: **groupadd Usuarios**

Asociar un usuario a un grupo existente:

- **usermod -G nombregrupo nombreusuario**

Ver los grupos asociados a un usuario:

- **groups nombreusuario**

Crear un usuario y asignarle un directorio:

- **sudo useradd -m -g Usuarios -s /bin/bash manolito**

Explicación:

- -m: Crear automáticamente la carpeta del usuario en la carpeta /Home/<NombreUsuario>
- -g: grupo principal al que será agregado
- -G: Grupos secundarios al que pertenecerá.
- -s: Shell que utilizará por defecto el usuario.
- manolito: Nombre del usuario.

Después de este comando, queda asignar la contraseña, para ello utilizamos el comando «**passwd**».

Permisos en ficheros

Código	Significado
u	Usuario (propietario)
g	Grupo
o	Otros
a	Todos (u+g+o)

Permiso	Letra	Valor	Significado
Lectura	r	4	Ver contenido
Escritura	w	2	Modificar/eliminar
Ejecución	x	1	Ejecutar o acceder

Para cambiar permisos de forma recursiva: **-R**

Ver permisos

ls -l Ejemplo → **-rwxr-xr--**

Cambiar permisos (chmod)

• Modo simbólico:

chmod u+x script.sh # Añade ejecución al usuario

chmod go-r archivo.txt # Quita lectura a grupo y otros

• Modo numérico:

chmod 755 archivo.txt # u=rwx, g=rx, o=rx

chmod 700 archivo.txt # Solo el dueño tiene acceso

Permisos en ficheros

Cambiar propietario (chown)

chown juan archivo.txt # Cambia el propietario chown :ventas archivo.txt # Cambia el grupo

chown -R juan:ventas /carpeta # Cambia todo recursivamente

Permisos especiales

Ejemplo:

chmod 1777 /tmp # Sticky bit

Ejemplos comunes

Objetivo	Comando
Script ejecutable	chmod +x script.sh
Solo dueño accede	chmod 700 archivo
Compartir carpeta en grupo	chmod 2775 carpeta
Servidor web	chown -R www-data:www-data /var/www/html chmod -R 755 /var/www/html

Permisos en ficheros

1. **ls -lh**: Mostrar permisos.
2. **ls /tmp | pr -T5 -W\$COLUMNS**: dividir la terminal en 5 columnas.
3. **chmod ugo+rwx directory1**: colocar permisos de lectura (@), escritura (w) y ejecución(x) al propietario (u), al grupo (g) y a otros (o) sobre el directorio 'directory1'.
4. **chmod go-rwx directory1**: quitar permiso de lectura (@), escritura (w) y (x) ejecución al grupo (g) y otros (o) sobre el directorio 'directory1'.
5. **chown user1 file1**: cambiar el dueño de un fichero.
6. **chown -R user1 directory1**: cambiar el propietario de un directorio y de todos los ficheros y directorios contenidos dentro.
7. **chgrp group1 file1**: cambiar grupo de ficheros.
8. **chown user1:group1 file1**: cambiar usuario y el grupo propietario de un fichero.
9. **find / -perm -u+s**: visualizar todos los ficheros del sistema con SUID configurado.
10. **chmod u+s /bin/file1**: colocar el bit SUID en un fichero binario. El usuario que corriendo ese fichero adquiere los mismos privilegios como dueño.
11. **chmod u-s /bin/file1**: deshabilitar el bit SUID en un fichero binario.
12. **chmod g+s /home/public**: colocar un bit SGID en un directorio –similar al SUID pero por directorio.
13. **chmod g-s /home/public**: desabilitar un bit SGID en un directorio.
14. **chmod o+t /home/public**: colocar un bit STIKY en un directorio. Permite el borrado de ficheros solamente a los dueños legítimos.
15. **chmod o-t /home/public**: desabilitar un bit STIKY en un directorio.

Olga M.Moreno

PROMETEO

Atributos especiales en ficheros

1. **chattr +a file1**: permite escribir abriendo un fichero solamente modo append.
2. **chattr +c file1**: permite que un fichero sea comprimido / descomprimido automaticamente.
3. **chattr +d file1**: asegura que el programa ignore borrar los ficheros durante la copia de seguridad.
4. **chattr +i file1**: convierte el fichero en invariable, por lo que no puede ser eliminado, alterado, renombrado, ni enlazado.
5. **chattr +s file1**: permite que un fichero sea borrado de forma segura.
6. **chattr +S file1**: asegura que un fichero sea modificado, los cambios son escritos en modo synchronous como con sync.
7. **chattr +u file1**: te permite recuperar el contenido de un fichero aún si este está cancelado.
8. **lsattr**: mostrar atributos especiales.

Archivos y ficheros comprimidos

1. **bunzip2 file1.bz2**: descomprime un fichero llamado 'file1.bz2'.
2. **bzip2 file1**: comprime un fichero llamado 'file1'.
3. **gunzip file1.gz**: descomprime un fichero llamado 'file1.gz'.
4. **gzip file1**: comprime un fichero llamado 'file1'.
5. **gzip -9 file1**: comprime con compresión máxima.
6. **rar a file1.rar test_file**: crear un fichero rar llamado 'file1.rar'.
7. **rar a file1.rar file1 file2 dir1**: comprimir 'file1', 'file2' y 'dir1' simultáneamente.
8. **rar x file1.rar**: descomprimir archivo rar.
9. **unrar x file1.rar**: descomprimir archivo rar.
10. **tar -cvf archive.tar file1**: crear un tarball descomprimido.
11. **tar -cvf archive.tar file1 file2 dir1**: crear un archivo contenido 'file1', 'file2' y 'dir1'.
12. **tar -tf archive.tar**: mostrar los contenidos de un archivo.
13. **tar -xvf archive.tar**: extraer un tarball.
14. **tar -xvf archive.tar -C /tmp**: extraer un tarball en / tmp.
15. **tar -cvfj archive.tar.bz2 dir1**: crear un tarball comprimido dentro de bzip2.
16. **tar -xvfj archive.tar.bz2**: descomprimir un archivo tar comprimido en bzip2
17. **tar -cvfz archive.tar.gz dir1**: crear un tarball comprimido en gzip.
18. **tar -xvfz archive.tar.gz**: descomprimir un archive tar comprimido en gzip.
19. **zip file1.zip file1**: crear un archivo comprimido en zip.
20. **zip -r file1.zip file1 file2 dir1**: comprimir, en zip, varios archivos y directorios de forma simultánea.
21. **unzip file1.zip**: descomprimir un archivo zip.

Paquetes DEB(Debian)

1. **dpkg -i package.deb**: instalar / actualizar un paquete deb.
2. **dpkg -r package_name**: eliminar un paquete deb del sistema.
3. **dpkg -l**: mostrar todos los paquetes deb instalados en el sistema.
4. **dpkg -l | grep httpd**: mostrar todos los paquetes deb con el nombre "httpd"
5. **dpkg -s package_name**: obtener información en un paquete específico instalado en el sistema.
6. **dpkg -L package_name**: mostar lista de ficheros dados por un paquete instalado en el sistema.
7. **dpkg --contents package.deb**: mostrar lista de ficheros dados por un paquete no instalado todavía.
8. **dpkg -S /bin/ping**: verificar cuál paquete pertenece a un fichero dado.

Ver el contenido de un fichero

1. **cat file1**: ver los contenidos de un fichero comenzando desde la primera hilera.
2. **tac file1**: ver los contenidos de un fichero comenzando desde la última línea.
3. **more file1**: ver el contenido a lo largo de un fichero.
4. **less file1**: parecido al comando 'more' pero permite salvar el movimiento en el fichero así como el movimiento hacia atrás.
5. **head -2 file1**: ver las dos primeras líneas de un fichero.
6. **tail -2 file1**: ver las dos últimas líneas de un fichero.
7. **tail -f /var/log/messages**: ver en tiempo real qué ha sido añadido al fichero.

Manipulación de texto

¿Cómo sería?

Para **buscar en Ejemplo1 y Ejemplo2** todas las palabras que contengan “**pru**” y guardar el resultado en **salida.txt**:

1. **cat file1 file2 .. | command <> file1_in.txt_or_file1_out.txt**: sintaxis general para la manipulación de texto utilizando PIPE, STDIN y STDOUT.
2. **cat file1 | command(sed, grep, awk, grep, etc...) > result.txt**: sintaxis general para manipular un texto de un fichero y escribir el resultado en un fichero nuevo.
3. **cat file1 | command(sed, grep, awk, grep, etc...) » result.txt**: sintaxis general para manipular un texto de un fichero y añadir resultado en un fichero existente.
4. **grep Aug /var/log/messages**: buscar palabras “Aug” en el fichero ‘/var/log/messages’.
5. **grep ^Aug /var/log/messages**: buscar palabras que comienzan con “Aug” en fichero ‘/var/log/messages’
6. **grep [0-9] /var/log/messages**: seleccionar todas las líneas del fichero ‘/var/log/messages’ que contienen números.
7. **grep Aug -R /var/log/***: buscar la cadena “Aug” en el directorio ‘/var/log’ y debajo.
8. **sed 's/stringa1/stringa2/g' example.txt**: reubicar “string1” con “string2” en ejemplo.txt
9. **sed '/^\$/d' example.txt**: eliminar todas las líneas en blanco desde el ejemplo.txt
10. **sed '/^#/d; /^\$/d' example.txt**: eliminar comentarios y líneas en blanco de ejemplo.txt
11. **echo 'esempio' | tr '[:lower:]' '[:upper:]'**: convertir minúsculas en mayúsculas.
12. **sed -e '1d' result.txt**: elimina la primera línea del fichero ejemplo.txt
13. **sed -n '/stringa1/p'**: visualizar solamente las líneas que contienen la palabra “string1”.

cat Ejemplo1.txt Ejemplo2.txt | grep pru > salida.txt

Establecer carácter y conversión de ficheros

1. **dos2unix filedos.txt fileunix.txt**: convertir un formato de fichero texto desde MSDOS a UNIX.
2. **unix2dos fileunix.txt filedos.txt**: convertir un formato de fichero de texto desde UNIX a MSDOS.
3. **recode ..HTML < page.txt > page.html**: convertir un fichero de texto en html.
4. **recode -l | more**: mostrar todas las conversiones de formato disponibles.

Montaje de sistemas de archivos

- Ubuntu, para facilitar esta tarea que en versiones anteriores no era automático, “automonta” las unidades extraíbles.
- Si los directorios de montaje no existen, hay que crearlos (con privilegios).
- **df -h** para ver todos los sistemas de archivos montados junto con algunos atributos

1. **mount /dev/hda2 /mnt/hda2**: montar un disco llamado hda2. Verifique primero la existencia del directorio '/mnt/hda2'; si no está, debe crearlo.
2. **umount /dev/hda2**: desmontar un disco llamado hda2. Salir primero desde el punto '/mnt/hda2'.
3. **fuser -km /mnt/hda2**: forzar el desmontaje cuando el dispositivo está ocupado.
4. **umount -n /mnt/hda2**: correr el desmontaje sin leer el fichero /etc/mtab. Útil cuando el fichero es de solo lectura o el disco duro está lleno.
5. **mount /dev/fd0 /mnt/floppy**: montar un disco flexible (floppy).
6. **mount /dev/cdrom /mnt/cdrom**: montar un cdrom / dvdrom.
7. **mount /dev/hdc /mnt/cdrecorder**: montar un cd regrabable o un dvdrom.
8. **mount /dev/hdb /mnt/cdrecorder**: montar un cd regrabable / dvdrom (un dvd).
9. **mount -o loop file.iso /mnt/cdrom**: montar un fichero o una imagen iso.
10. **mount -t vfat /dev/hda5 /mnt/hda5**: montar un sistema de ficheros FAT32.
11. **mount /dev/sda1 /mnt/usbdisk**: montar un usb pen-drive o una memoria (sin especificar el tipo de sistema de ficheros).

Análisis del sistema de ficheros

1. **badblocks -v /dev/hda1**: Chequear los bloques defectuosos en el disco hda1.
2. **fsck /dev/hda1**: reparar / chequear la integridad del fichero del sistema Linux en el disco hda1.
3. **fsck.ext2 /dev/hda1**: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
4. **e2fsck /dev/hda1**: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
5. **e2fsck -j /dev/hda1**: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
6. **fsck.ext3 /dev/hda1**: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
7. **fsck.vfat /dev/hda1**: reparar / chequear la integridad del fichero sistema fat en el disco hda1.
8. **fsck.msdos /dev/hda1**: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.
9. **dosfsck /dev/hda1**: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.

Formatear un sistema de ficheros

1. **mkfs /dev/hda1**: crear un fichero de sistema tipo Linux en la partición hda1.
2. **mke2fs /dev/hda1**: crear un fichero de sistema tipo Linux ext 2 en hda1.
3. **mke2fs -j /dev/hda1**: crear un fichero de sistema tipo Linux ext3 (periódico) en la partición hda1.
4. **mkfs -t vfat 32 -F /dev/hda1**: crear un fichero de sistema FAT32 en hda1.
5. **fdformat -n /dev/fd0**: formatear un disco flooply.
6. **mkswap /dev/hda3**: crear un fichero de sistema swap.

Para formatear en nuestros sistemas Ubuntu un PenDrive en el formato FAT32:

mkfs.vfat -F32 /dev/sdb2

Trabajando con un PenDrive

1. Como sabemos, en el directorio /dev podemos ver todos los dispositivos disponibles. Podemos listar el directorio mediante “ls /dev”.
2. Para verlo mejor, ejecutamos “**fdisk /dev/sdb**”, accediendo al gestor de particiones de Linux.
 - Pulsaremos “m” para obtener la ayuda.
 - Para imprimir la tabla de particiones, pulsaremos “p”.
 - Para salir sin guardar, pulsaremos “q”.
3. ¿Cómo vemos el contenido del PenDrive que se encuentra en /dev/sdb2?
 - Si probamos con “ls /dev/sdb2” no obtendremos ningún listado. ¿Porqué?
 - Porque es una partición, y la única forma de ver su contenido es **asociándolo a un directorio**.
 - Creamos en /mnt una carpeta llamada, por ejemplo, “pendrive”.
 - MONTAMOS la partición en el directorio recién creado.
 - mount /dev/sdb2 /mnt/pendrive
 - Para extraer el dispositivo montado CON SEGURIDAD debemos desmontarlo:
 - umount /dev/sdb2

Backups

1. **dump -0aj -f /tmp/home0.bak /home**: hacer una salva completa del directorio '/home'.
2. **dump -1aj -f /tmp/home0.bak /home**: hacer una salva incremental del directorio '/home'.
3. **restore -if /tmp/home0.bak**: restaurando una salva interactivamente.
4. **rsync -rogpav -delete /home /tmp**: sincronización entre directorios.
5. **rsync -rogpav -e ssh -delete /home ip_address:/tmp**: rsync a través del túnel SSH.
6. **rsync -az -e ssh -delete ip_addr:/home/public /home/local**: sincronizar un directorio local con un directorio remoto a través de ssh y de compresión.
7. **rsync -az -e ssh -delete /home/local ip_addr:/home/public**: sincronizar un directorio remoto con un directorio local a través de ssh y de compresión.
8. **dd bs=1M if=/dev/hda | gzip | ssh user@ip_addr 'dd of=hda.gz'**: hacer una salva de un disco duro en un host remoto a través de ssh.
9. **dd if=/dev/sda of=/tmp/file1**: salvar el contenido de un disco duro a un fichero. (En este caso el disco duro es "sda" y el fichero "file1").
10. **tar -Puf backup.tar /home/user**: hacer una salva incremental del directorio '/home/user'.

Redes (I)

1. **ifconfig eth0**: mostrar la configuración de una tarjeta de red Ethernet.
2. **ifup eth0**: activar una interface 'eth0'.
3. **ifdown eth0**: deshabilitar una interface 'eth0'.
4. **ifconfig eth0 192.168.1.1 netmask 255.255.255.0**: configurar una dirección IP.
5. **ifconfig eth0 promisc**: configurar 'eth0'en modo común para obtener los paquetes (sniffing).
6. **dhclient eth0**: activar la interface 'eth0' en modo dhcp.
7. **route -n**: mostrar mesa de recorrido.
8. **route add -net 0/0 gw IP_Gateway**: configurar entrada predeterminada.
9. **route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1**: configurar ruta estática para buscar la red '192.168.0.0/16'.
10. **route del 0/0 gw IP_gateway**: eliminar la ruta estática.
11. **echo "1" > /proc/sys/net/ipv4/ip_forward**: activar el recorrido ip.
12. **hostname**: mostrar el nombre del host del sistema.

Redes (II)

13. **host www.example.com**: buscar el nombre del host para resolver el nombre a una dirección ip(1).
14. **nslookup www.example.com**: buscar el nombre del host para resolver el nombre a una dirección ip y viceversa(2).
15. **ip link show**: mostrar el estado de enlace de todas las interfaces.
16. **mii-tool eth0**: mostrar el estado de enlace de 'eth0'.
17. **ethtool eth0**: mostrar las estadísticas de tarjeta de red 'eth0'.
18. **netstat -tup**: mostrar todas las conexiones de red activas y sus PID.
19. **netstat -tupl**: mostrar todos los servicios de escucha de red en el sistema y sus PID.
20. **tcpdump tcp port 80**: mostrar todo el tráfico HTTP.
21. **iwlist scan**: mostrar las redes inalámbricas.
22. **iwconfig eth1**: mostrar la configuración de una tarjeta de red inalámbrica.
23. **whois www.example.com**: buscar en base de datos Whois.

IPTables – NFTables (Firewall)

1. **iptables -t filter -L**: mostrar todas las cadenas de la tabla de filtro.
2. **iptables -t nat -L**: mostrar todas las cadenas de la tabla nat.
3. **iptables -t filter -F**: limpiar todas las reglas de la tabla de filtro.
4. **iptables -t nat -F**: limpiar todas las reglas de la tabla nat.
5. **iptables -t filter -X**: borrar cualquier cadena creada por el usuario.
6. **iptables -t filter -A INPUT -p tcp -dport telnet -j ACCEPT**: permitir las conexiones telnet para entrar.
7. **iptables -t filter -A OUTPUT -p tcp -dport http -j DROP**: bloquear las conexiones HTTP para salir.
8. **iptables -t filter -A FORWARD -p tcp -dport pop3 -j ACCEPT**: permitir las conexiones POP a una cadena delantera.
9. **iptables -t filter -A INPUT -j LOG -log-prefix "DROP INPUT"**: registrando una cadena de entrada.
10. **iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE**: configurar un PAT (Puerto de traducción de dirección) en eth0, ocultando los paquetes de salida forzada.
11. **iptables -t nat -A PREROUTING -d 192.168.0.1 -p tcp -m tcp -dport 22 -j DNAT -to-destination 10.0.0.2:22**: redireccionar los paquetes dirigidos de un host a otro.

Monitorización y depuración (I)

1. **top**: mostrar las tareas de linux usando la mayoría cpu.
2. **ps -eafw**: muestra las tareas Linux.
3. **ps -e -o pid,args -forest**: muestra las tareas Linux en un modo jerárquico.
4. **pstree**: mostrar un árbol sistema de procesos.
5. **kill -9 ID_Proceso**: forzar el cierre de un proceso y terminarlo.
6. **kill -1 ID_Proceso**: forzar un proceso para recargar la configuración.
7. **lsof -p \$\$**: mostrar una lista de ficheros abiertos por procesos.
8. **lsof /home/user1**: muestra una lista de ficheros abiertos en un camino dado del sistema.
9. **strace -c ls >/dev/null**: mostrar las llamadas del sistema hechas y recibidas por un proceso.

Monitorización y depuración (II)

10. **strace -f -e open ls >/dev/null**: mostrar las llamadas a la biblioteca.
11. **watch -n1 'cat /proc/interrupts'**: mostrar interrupciones en tiempo real.
12. **last reboot**: mostrar historial de reinicio.
13. **lsmod**: mostrar el kernel cargado.
14. **free -m**: muestra el estado de la RAM en megabytes.
15. **smartctl -A /dev/hda**: monitorear la fiabilidad de un disco duro a través de SMART.
16. **smartctl -i /dev/hda**: chequear si SMART está activado en un disco duro.
17. **tail /var/log/dmesg**: mostrar eventos inherentes al proceso de carga del kernel.
18. **tail /var/log/messages**: mostrar los eventos del sistema.

Otros comandos

apropos

1. **apropos ...keyword**: mostrar una lista de comandos que pertenecen a las palabras claves de un programa; son útiles cuando tú sabes qué hace tu programa, pero de sconoces el nombre del comando.
2. **man ping**: mostrar las páginas del manual on-line; por ejemplo, en un comando ping, usar la opción '-k' para encontrar cualquier comando relacionado.
3. **whatis ...keyword**: muestra la descripción de lo que hace el programa.
4. **mkbootdisk -device /dev/fd0 `uname -r`**: crear un floppy boteable.
5. **gpg -c file1**: codificar un fichero con guardia de seguridad GNU.
6. **gpg file1.gpg**: decodificar un fichero con Guardia de seguridad GNU.
7. **wget -r www.example.com**: descargar un sitio web completo.
8. **wget -c www.example.com/file.iso**: descargar un fichero con la posibilidad de parar la descargar y reanudar más tarde.
9. **echo 'wget -c www.example.com/files.iso' | at 09:00**: Comenzar una descarga a cualquier hora. En este caso empezaría a las 9 horas.
10. **ldd /usr/bin/ssh**: mostrar las bibliotecas compartidas requeridas por el programa ssh.
11. **alias hh='history'**: colocar un alias para un commando -hh= Historial.
12. **chsh**: cambiar el comando Shell.
13. **chsh -list-shells**: es un comando adecuado para saber si tienes que hacer remoto en otra terminal.
14. **who -a**: mostrar quien está registrado, e imprimir hora del último sistema de importación, procesos muertos, procesos de registro de sistema, procesos activos producidos por init, funcionamiento actual y últimos cambios del reloj del sistema.

Resumen básico

Comando	Función principal	Ejemplo
ls	Lista archivos y carpetas	ls -l
pwd	Muestra la ruta actual	pwd
cd	Cambia de directorio	cd /home/usuario
cat	Muestra contenido de un archivo	cat archivo.txt
less	Visualiza archivos por páginas	less archivo.log
cp	Copia archivos o carpetas	cp origen destino
mv	Mueve o renombra archivos	mv viejo.txt nuevo.txt
rm	Elimina archivos o carpetas	rm archivo.txt
mkdir	Crea un directorio nuevo	mkdir carpeta
tar	Comprime o descomprime archivos	tar -czvf archivo.tar.gz carpeta/
ps	Muestra procesos activos	ps aux
top	Monitorea recursos en tiempo real	top
htop	Versión visual de top	htop
pstree	Muestra procesos en forma de árbol	pstree
apt	Gestor de paquetes	apt-get install /update/upgrade

Resumen intermedio

Comando	Función principal	Ejemplo
man	Muestra el manual de un comando	man ls
whoami	Muestra el usuario actual	whoami
sudo	Ejecuta comandos con privilegios de administrador	sudo apt update
history	Muestra el historial de comandos	history
clear	Limpia la pantalla de la terminal	clear
echo	Imprime texto o variables	echo "Hola Mundo"
touch	Crea archivos vacíos o actualiza fechas	touch nuevo.txt
find	Busca archivos y carpetas	find /home -name "*.txt"
grep	Busca texto dentro de archivos	grep "error" log.txt
head	Muestra las primeras líneas de un archivo	head -n 10 archivo.txt
tail	Muestra las últimas líneas (útil para logs)	tail -f /var/log/syslog
df	Muestra espacio libre en discos	df -h
du	Muestra el tamaño de archivos o carpetas	du -sh carpeta/
chmod	Cambia permisos de archivos	chmod 755 script.sh



PROMETEO