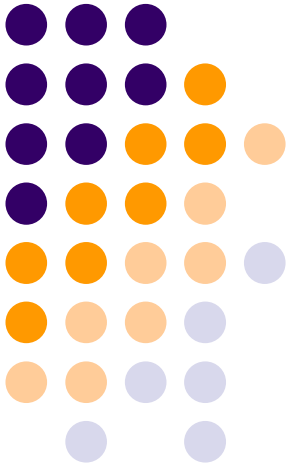
Unidad 6

Comandos básicos GNU/Linux

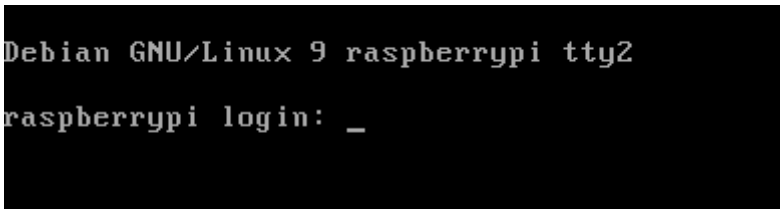


UNIDAD 6
Comandos básicos GNU/Linux

1. Terminales de texto
2. Comandos básicos
3. Redirección y tuberías
4. Permisos
5. Búsquedas
6. Empaquetar y comprimir
7. Montaje de unidades
8. Apagado y reinicio

1. Terminales de texto

- Terminales de texto
  - GNU/Linux cuentan con una interfaz de comandos muy avanzada (facilita muchas tareas, requiere mayor aprendizaje)
  - Cada usuario puede interaccionar con el sistema a través de un terminal de texto o consola
  - Linux dispones de varias consolas
  - Para cambiar de consola:
    CTRL + ALT + F1 … **CTRL + ALT + F8**
  - Cada una de las consolas ofrece el prompt de entrada para poder acceder al sistema



1. Terminales de texto

- Terminales virtuales
  - Aplicación que emula un terminal de texto
  - Integran la interfaz gráfica con la interfaz de texto
  - Accede a un **shell** de Linux
  - Permite ejecutar aplicaciones de terminal

1. Terminales de texto

- Shell
  - Shell: programa que se ejecuta al iniciar una sesión de trabajo y se encarga de atender los mandatos del usuario y ejecutar los programas correspondientes
  - Capa más externa de la estructura de UNIX/Linux
  - Programable
  - Varios shells: bash (*Bourne again shell*, por defecto en Ubuntu), ksh (*Korn shell*), csh (*C shell*), sh,...
  - Prompt → **usuario@maquina:~$**
  - El prompt es personalizable

2. Comandos básicos

- Sintaxis
  - Comando → fichero ejecutable
  - Formato: <comando> [OPCIONES] [ARGUMENTOS]
  - [OPCIONES]:
    - Letras precedidas del símbolo '-'
    - Palabras precedidas de los símbolos '--'
  - [ARGUMENTOS]:
    - Nombres de ficheros y directorios, etc.

2. Comandos básicos

- Ayuda
  - man
    - `$ man <comando>`
    - `$ man <sección> <comando>`
    - `$ whatis <comando>`
  - <comando> --help
  - apropos <término> (ofrece comandos relacionados con <término>)

2. Comandos básicos

- Rutas
  - Al comenzar una sesión, nos encontramos en nuestra carpeta **home** (/home/usuario o ~)
  - pwd (print work directory) → devuelve la ruta actual absoluta dentro del árbol de directorio
    - `$ pwd`

2. Comandos básicos

- Directorios
  - **cd** (change directory) → cambia de directorio

```
$ cd ruta_directorio_destino
$ cd ..
```

  - **ls** (list) → muestra contenido de directorio (lista ficheros)
    - Opciones:
      - -l: ofrece información adicional: tipo de archivos, permisos, propietario y grupo, fecha de creación
      - -a: lista ficheros ocultos
      - -R: lista contenido de forma jerárquica

```
$ ls
$ ls -al
```

2. Comandos básicos

- Ficheros
  - **cat** (catch) → muestra contenido de fichero (y concatena)
    - cat fichero1 … fichero ‘n’

```
$ cat fichero.txt
```

  - **more** → muestra contenido de forma paginada

```
$ more fichero.txt
```

  - **less** → similar a more (uso de cursores)

```
$ less fichero.txt
```

2. Comandos básicos

- Copiar y mover ficheros
  - **cp** (copy) → copia ficheros y directorios

```
$ cp fichero1 fichero2
$ cp -r directorio1 directorio2
```

  - **mv** (move) → mueve ficheros y directorios

```
$ mv fichero1 fichero2 (renombra)
$ mv fichero1 directorio1
```

2. Comandos básicos

- Creación y borrado de archivos y directorios
  - **mkdir** (make directory) → crea directorio

    ```
    $ mkdir directorio
    ```

  - **rm** (remove) → borra ficheros y directorios

    ```
    $ rmdir directorio (solo si está vacío)
    $ rm -r directorio
    $ rm fichero
    ```

2. Comandos básicos

- Caracteres especiales y comodines
  - ***** → representa cualquier cadena de caracteres
  - **?** → representa cualquier carácter simple
  - **[-]** → uno o más caracteres de los incluidos entre corchetes
  - **!** → exclusión de caracteres

2. Comandos básicos

- Cadenas de texto
  - **touch** → crea archivo de texto
  - **grep**→ muestra las líneas del fichero en las que aparece una subcadena indicada como patrón
  - **sort** → ordena líneas del fichero de entrada (muchas opciones)
  - **head** → muestra las primeras ‘n’ líneas de un fichero (10 por defecto)
  - **tail** → muestra las últimas ‘n’ líneas de un fichero

2. Comandos básicos

- Cadenas de texto
  - **wc** → cuenta el número de palabras, líneas o caracteres de un fichero
  - **paste** →concatena línea a línea los ficheros de entrada
  - **cut** → escribe en la salida estándar secciones de cada del fichero de entrada
  - **sed** → editor de texto orientado a “flujo

2. Comandos básicos

- Enlaces
  - **ln** (link) → crea un enlace a un fichero o directorio
  - Hay dos tipos:
    - Físico (hard, duro): asigna un nuevo nombre al archivo (mismo inodo)
      - $ ln <fichero>
    - Simbólico (soft, blando): “acceso directo”
      - $ ln **-s** <fichero>

2. Comandos básicos

- Otros comandos
  - **clear** → borra la pantalla
  - **uname** → muestra información del sistema

    ```
    $ uname -a
    ```

  - **file** → indica naturaleza de un fichero (fichero de texto, binario ejecutable, código C, ...)
  - **who** → muestra los usuarios conectados al sistema
  - **whoami** → nombre del usuario actual
  - **hostname** → muestra el nombre del equipo

3. Redirección y tuberías

- Entrada y salida estándar
  - **stdin** (stardard input) → entrada estándar
  - **stdout** (stardard output) → salida estándar
  - Muchos comandos toman su entrada de stdin y mandan su salida a stdout
  - El terminal virtual por defecto toma como stdin el teclado y como stdout la pantalla

3. Redirección y tuberías

- Redireccionamiento de la salida
  - **>** : envía la salida a un fichero (destruye contenido anterior)
  - **>>** : envía salida (añade al contenido anterior)

```
$ ls > listado.txt
$ cat > lista_compra.txt
  Galletas
  Naranjas
  Lentejas
  Ctrl-D
```

3. Redirección y tuberías

- Redireccionamiento de la entrada
  - `<` y `<<` : toman la entrada de un fichero (en vez de teclado)

```
$ cat < lista_compra.txt
Galletas
Naranjas
Lentejas
```

3. Redirección y tuberías

- Redireccionamiento entre comandos (pipes o tuberías)
  - Permiten tomarla salida de un comando como entrada de otro
  - Se pueden conectar más de dos comandos

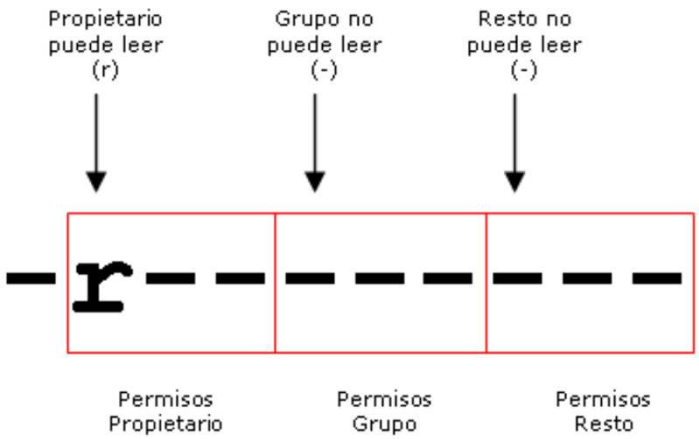
```
$ ls | sort -r
$ ls /home/usuario | less
$ ls /home/usuario | sort -r | less
```

4. Permisos

- Permisos en UNIX/Linux
  - En Unix/Linux todos los archivos pertenecen obligatoriamente a un usuario y a un grupo
  - Cuando un usuario crea un nuevo archivo, el propietario del archivo será el usuario que lo ha creado
  - El grupo del archivo será el grupo principal al que pertenece dicho usuario
  - Hay tres tipos de permisos, aplicables a archivos y carpetas: permiso de **lectura**, permiso de **escritura** y permiso de **ejecución**

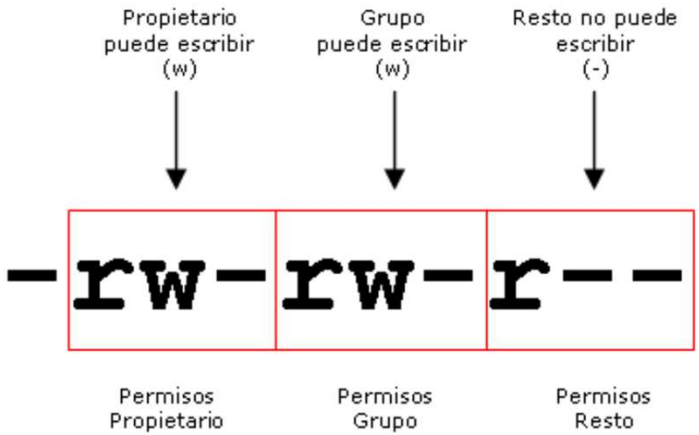
4. Permisos

- Permiso de lectura (r, read)
  - Sobre un archivo significa que se puede leer o visualizar
  - Sobre una carpeta significa que se puede visualizar su contenido



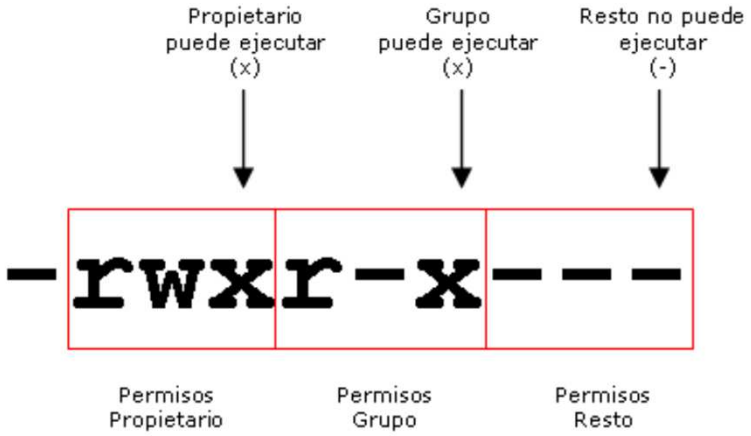
4. Permisos

- Permiso de escritura (w, write)
  - Sobre un archivo significa que se puede modificar su contenido e incluso borrarlo; también permite cambiar los permisos, así como cambiar su propietario y grupo
  - Sobre una carpeta significa que se puede modificar su contenido: puede crear y eliminar archivos y otras carpetas dentro de ella



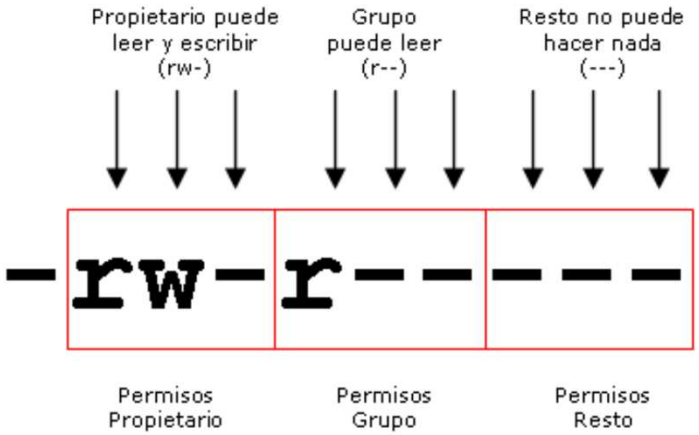
4. Permisos

- Permiso de ejecución (x, eXecute)
  - Sobre un archivo significa que se puede ejecutar
  - Los archivos ejecutables son las aplicaciones y los archivos de comandos (scripts)
  - Sobre una carpeta significa que se puede entrar en ella



4. Permisos

- Asignación de permisos
  - Los permisos pueden ser otorgados a:
    - El **usuario** propietario del archivo
    - El **grupo** propietario del archivo
    - El **resto** de usuarios del sistema
  - No es posible asignar permisos a usuarios o grupos concretos



4. Permisos

- Asignación de permisos
  - Al hacer **ls -l** sobre una carpeta, se visualizan los permisos de los archivos y carpetas que contiene
    - El primer carácter indica el tipo de archivo
    - Los 9 siguientes indican los permisos de lectura, escritura y ejecución para el propietario, el grupo y el resto de usuarios

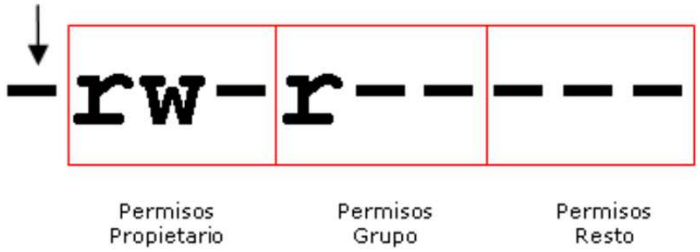
Tipo de archivo:

(-) para archivos normales

(d) para carpetas (directory)

(l) para enlaces (link)

(s)=socket, (p)=tubería (pipe), (b)=dispositivo de bloque.



4. Permisos

- Asignación de permisos
  - **chmod** → comando para cambiar los permisos de un archivo o una carpeta
    - chmod [opciones] permiso *archivo_o_carpeta*
  - Los permisos se pueden representar de dos formas:
    1) Mediante las iniciales de a quién va dirigido el permiso [usuario=**u**, grupo=**g**, resto=**o** (other)], un signo **+** si se quiere añadir permiso o **-** si se quiere quitar, y el tipo de permiso (lectura=**r**, **escritura=w** y ejecución=**x**)
    2) Mediante un código octal (binario con 3 bits) que representaría la activación o desactivación de los 3 tipos de permisos para propietario, grupo y el resto de usuarios

4. Permisos

- Asignación de permisos
  - Ejemplos:

```
$ chmod u+w control.txt
$ chmod o-w control.txt
$ chmod g+x usr/bin/games/tetris
$ chmod ug+w control.txt
$ chmod u+w,g-r,o-r control.txt
```

```
$ chmod 700 nomina.txt
$ chmod 740 prueba.txt
$ chmod 777 script.sh
```

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

4. Permisos

- Asignación de permisos
  - Cambiar propietario:
    - **chown** <usuario> <archivo>
    - Sirve tanto para usuarios como grupos (en este caso antecediendo el nombre del grupo de :)
  - Cambiar grupo principal:
    - **chgrp** <grupo> <archivo>

```
$ chown ernesto ejercicios.txt
$ chgrp alumnos ejercicios.txt
$ chown ernesto:alumnos ejercicios.txt
```

5. Búsqueda

- Buscar archivos y carpetas
  - **find** → comando para realizar búsquedas
    - find <ruta> <expresión_de_búsqueda> <acción>
      - <ruta> → indica el directorio a partir del cual se hará la búsqueda (por defecto directorio actual (.)
      - <expresión> → acción a realizar una vez localizado el archivo/archivos (por defecto -print)
      - <acción> → comando a ejecutar sobre el resultado

```
$ find /Imagenes *.png
```

6. Empaquetar y comprimir

- Empaquetar archivos
  - **tar** → comando para empaquetar archivos
    - tar [opciones] <archivo>
      - -c: “empaqueta” el archivo
      - -x: “desempaqueta” el archivo
      - -v: muestra lo que va empaquetando/desempaquetando
      - -f: indica que el siguiente argumento es el nombre del fichero a empaquetar/desempaquetar

```
$ tar cvf curso.tar /Documentos/archivos_curso
$ tar xvf curso.tar
```

6. Empaquetar y comprimir

- Comprimir archivos
  - **gzip** → comprime archivos en fomato .gz
    - gzip [opciones] <archivo>
      - -d: descomprimir

```
$ gzip  curso.gz /Documentos/archivos_curso
$ gzip -d curso.gz
```

  - A tar se le puede añadir la opción **-z** para que, además, comprima el archivo

```
$ tar cvfz curso.tar.gz /Documentos/archivos_curso
$ tar xvfz curso.tar.gz
```

  - Otros formatos: **zip**/**unzip**, **rar**/**unrar**, **bzip**/**bzip2**

7. Montaje de unidades

- Montar y desmontar unidades
  - **mount** → permite montar sistemas de ficheros y anclarlos a puntos de montaje (directorios)
    - mount [-t sistema de archivos] [dispositivo] punto_de_montaje

```
$ mount /dev/sdb1 /media/pendrive
```

```
$ mount -t ntfs /dev/sda2 /mnt/carpeta_Windows
```

  - **umount** → desmonta la unidad

```
$ umount /media/pendrive
```

8. Apagado y reinicio

- Cierre de sesión, apagado y reinicio del equipo
  - logout → cerrar sesión de usuario

    ```
    $ logout
    ```

  - shutdown → apagado del equipo

    ```
    $ shutdown -h now
    ```

  - reboot → reinicio

    ```
    $ reboot
    ```