U3. Introducción al sistema operativo Linux. **Parte I**

Implantación de Sistemas Operativos

Índice

- Introducción Linux
  - Características
  - Interfaz
  - Jerarquía de directorios
- Gestión de ficheros
- Gestión de usuarios

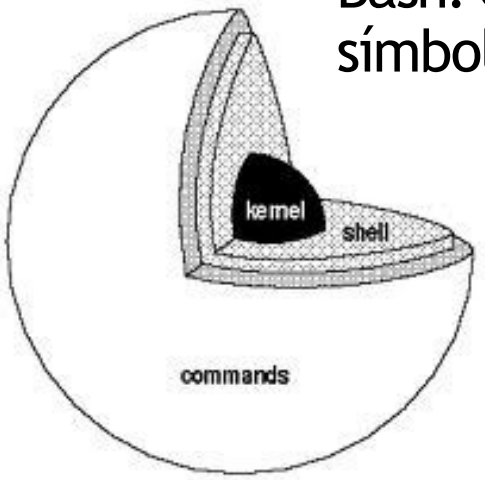
¿Por qué Linux?

- Linux o GNU Linux es un sistema operativo multiusuario de código abierto y gratuito.
  - Se distribuye con licencia GPL.
  - Viene acompañado de software libre (LibreOffice, Brasero, Nano, Vi, etc) que forma parte del proyecto GNU.
- Su creador fue *Linus Torvalds*, inspirado en MINIX, un sistema Unix simplificado y gratuito para estudiantes.
- Características:
  - Es muy **robusto**: menos sensible a errores.
  - Es más **seguro**: está afectado por un menor número de virus en comparación con otros sistemas operativo.
  - Algunas **tareas** son más **sencillas** en Linux que en Windows.
  - Cuenta con numerosas distribuciones o "versiones": http://futurist.se/gldt/

Características linux

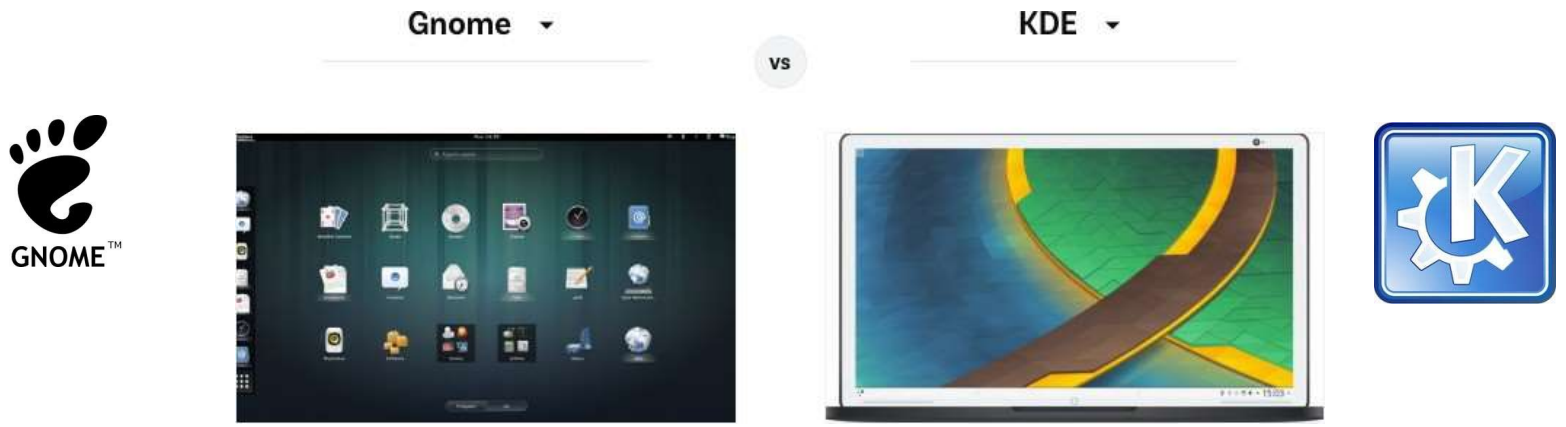
El sistema lo forman el **núcleo del sistema** (kernel) + el **interfaz de comandos** (shell)

- Kernel o núcleo: es la parte del sistema operativo que interactúa con el hardware. En linux está programado en lenguaje C.
- Shell: Sirve para ejecutar comandos que permiten emplear programas. Los intérpretes de comandos más empleados son:
  - Shell Bourne (sh): su símbolo del sistema es $
  - Bash: es la versión del Shell para el proyecto GNU y su símbolo del sistema es usuario@equipo



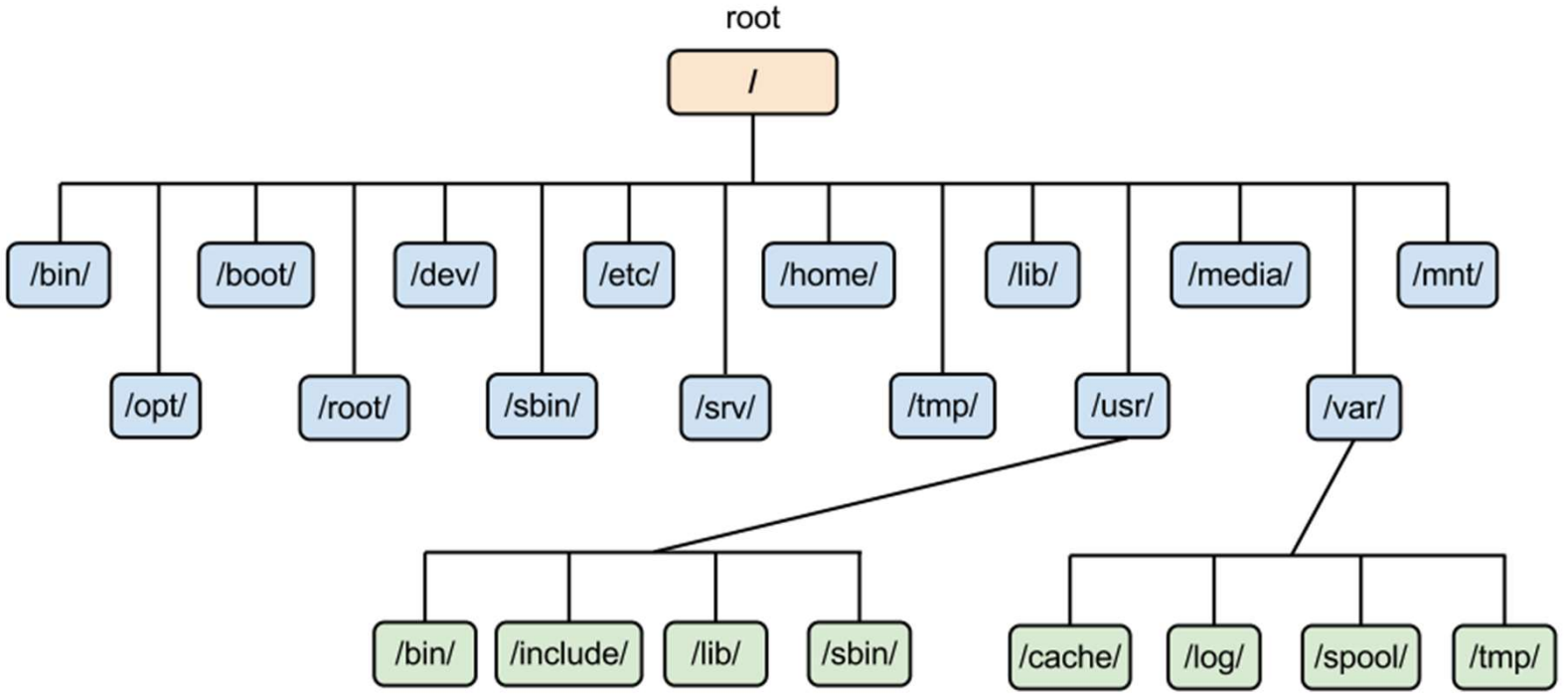
Interfaz gráfica

- Posteriormente se añadió un interfaz gráfico para emplear un entorno gráfico con ratón.
  - Desde el interfaz gráfico tenemos acceso al textual si ejecutamos un terminal.
  - Los entornos gráficos más habituales son: Gnome y KDE.
  - Es posible tener instalado más de un entorno gráfico y escoger cual emplear.



Jerarquía de ficheros

- En *GNU/Linux*, todos los dispositivos de almacenamiento conectados al ordenador se organizan en un mismo árbol de directorios. En él, cada volumen (cada partición) se integra en un punto concreto del árbol.



/ directorio raíz	**/bin**	Contiene programas ejecutables básicos para el sistema.
	/boot	Contiene los ficheros necesarios para el arranque del sistema.
	/dev	Contiene los ficheros correspondientes a los dispositivos: sonido, impresora, disco duro, lector de cd/dvd, video, etc.
	/etc	Contiene ficheros y directorios de configuración.
	/home	Contiene los directorios de trabajo de los usuarios. Cada usuario tiene su propio directorio en el sistema dentro de /home/.
	/lib	Contiene las librerías compartidas y los módulos del kernel
	/media	Dentro de este directorio se montan los dispositivos como el CD-ROM, memorias USB, discos duros portátiles, etc
	/opt	Directorio reservado para instalar aplicaciones.
	/sbin	Contiene los ficheros binarios ejecutables del sistema operativo.
	/srv	Contiene datos de los servicios proporcionado por el sistema.
	/tmp	Directorio de archivos temporales.
	/usr	Aquí se encuentran la mayoría de los archivos del sistema, aplicaciones, librerías, manuales, juegos… Es un espacio compartido por todos los usuarios.
	/var	Contiene archivos administrativos y datos que cambian con frecuencia: registro de errores, bases de datos, colas de impresión, etc.
	/root	Directorio de trabajo del administrador del sistema (usuario root).
	/proc	Aquí se almacenan datos del kernel e información sobre procesos.

Comandos y entrada/salidas estándar

- Los **Comandos Linux** son palabras reservadas que usa el sistema operativo para ejecutar programas usando una terminal o línea de comandos. La sintaxis general de las órdenes del Shell es:
  - orden [-opciones] argumento1 argumento2
  - Las opciones a veces se pueden agrupar: ls –l –a = ls –la
  - Es importante recordar que en Linux se diferencian las mayúsculas de las minúsculas.
- Todo programa en Linux tendrá:
  - Una entrada estándar (stdin) que por omisión es el teclado (es el tipo 0)
  - Una salida estándar (stdout) que por omisión es la pantalla (es tipo 1)
  - Un salida de error estándar (stderr) que por omisión es la pantalla (es tipo 2)

Comandos y entrada/salidas estándar

- Redirecciones: permiten modificar la entrada o salidas estándar:
  - **>** y **>>** permite redirigir la salida y en lugar de mostrarlo por pantalla, enviarla a un fichero. Si usamos **>** borra el contenido previo y escribe desde 0. En el caso de **>>** añade los nuevos contenidos al final del fichero sin borrar lo que ya había. Si el fichero no existía se **crea uno nuevo con ese nombre**.
    - Ej: ls -a > fichero.txt o ls -la >> fichero.txt
  - Podemos descartar el error de un programa con programa 2> /dev/null
  - **<** permite redirigir la entrada. Ej: wc < fichero.txt (tomando como entrada fichero.txt, aplica el comando wc para contar líneas, palabras y bytes)

Comandos y entrada/salidas estándar

- **Tuberías:** concatenan la entrada y salida de 2 comandos. Hacemos que la salida de un proceso se convierta en la entrada de otro proceso.

```
comando1 | comando2 | comando3 |
```

Ej: cat fichero.txt | grep fichero

- **Ayuda sobre comandos:**
  - man → página de ayuda sobre el comando
  - --help → ayuda rápida sobre el uso del comando

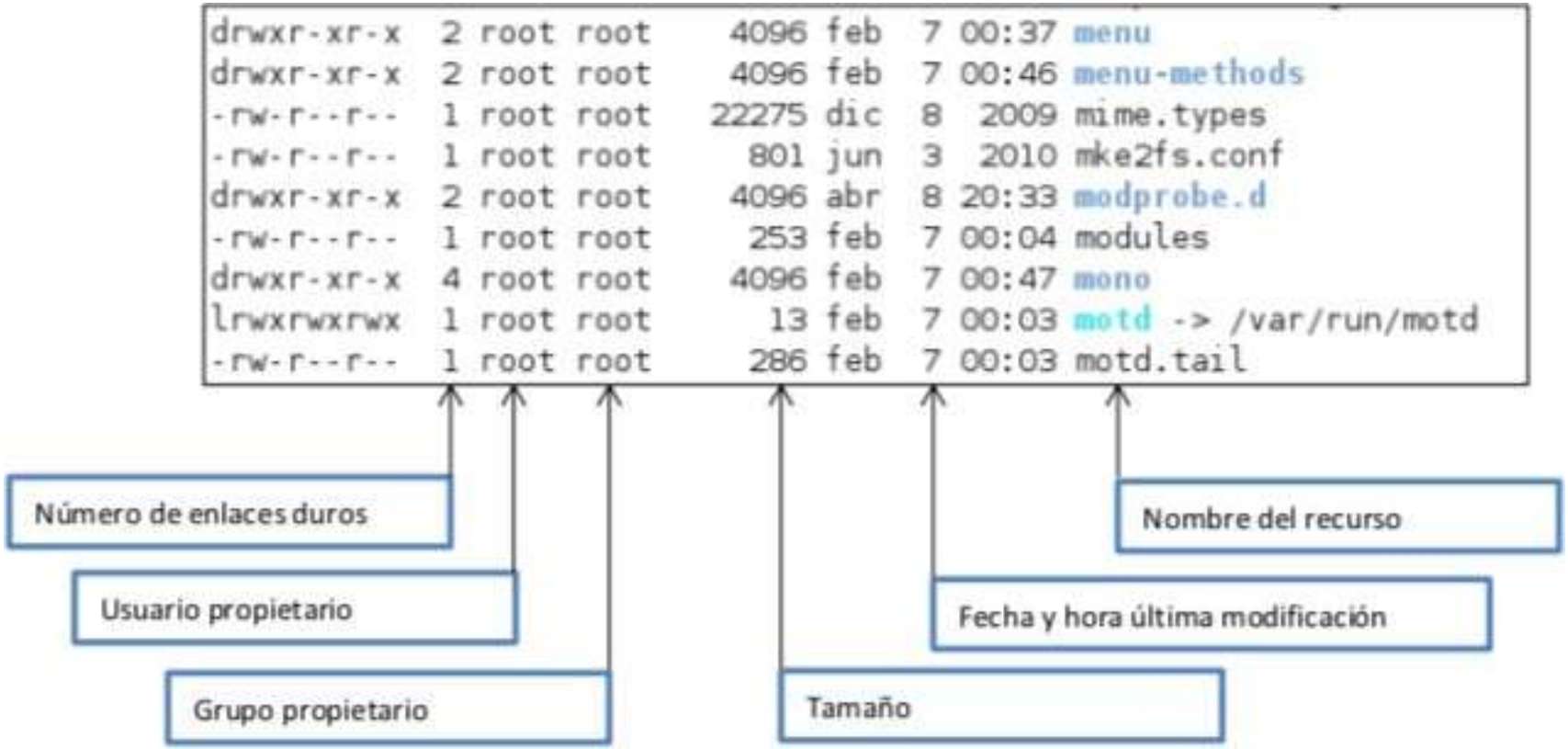
Ficheros y directorios

Comandos:

- pwd: permite conocer el nivel en el que nos encontramos
- cd: permite moverse por los directorios
- ls: lista los ficheros
- mkdir: permite crear directorios
- rm: Borra ficheros y directorios
- rmdir: Borra ficheros y directorios si están vacíos
- mv: permite mover ficheros
- cp: copia archivos o directorios
- cmp: compara dos ficheros byte a byte
- cat: Muestra el contenido de un fichero
- more: Muestra el contenido de un fichero de forma paginada
- head y tail: muestran respectivamente las primeras o últimas líneas de un fichero.

Ls -l

- Además de los permisos sobre ficheros (primera columna), la versión larga del listado de ficheros devuelve los campos siguientes:



Búsqueda de ficheros

- Búsqueda de patrones: grep
  - Sintaxis: grep [opciones] <patrón> [ficheros]
  - Opciones más comunes:
    - -c devuelve el número de líneas
    - -i no distingue mayúsculas de minúsculas
    - -v devuelve las líneas que no contienen el patrón
  - Ejemplos: **grep root /etc/passwd**
- Búsqueda de ficheros: find
  - Sintaxis: find [ruta] [opciones]
  - Opciones más comunes:
    - -name <nombre>
    - -type <tipo> (f es un fichero normal, l es un enlace, d es un directorio)
    - -size +/-<n codificados> (la codificación es c bytes, k kilobytes, etc)
    - -perm +/-<modo> permite buscar los ficheros que tengan unos permisos
    - -mtime +/- N (archivos modificados hasta/desde N días)
    - -exec <comando> "{}" ";" (ejecuta un comando sobre los archivos encontrados con find. {} se sustituye por los archivos automáticamente.
  - Ejemplos: **find -name '*.txt' -perm 644 -type f -size +10c -exec rm -f "{}" ";"**

Ordenación de ficheros

- El comando **sort** ordena las líneas de un fichero mostrándolas por la salida estándar.
- Sintaxis: sort [opciones] [fichero]
- Algunas opciones:
  - **-r** : ordena al revés (alfabéticamente)
  - **-f** : trata las mayúsculas y minúsculas por igual.
  - **-n** : ordena de forma numérica, de modo que no es necesario que los números se rellenen con ceros por la izquierda.
  - **-t** permite escoger un separador
  - **-u** permite eliminar líneas repetidas
- Ejemplo:
  - **$ sort -f /etc/passwd**
  - **$ cat /etc/passwd | sort -t":" -k4n (**Con –k4 le indicamos a sort que queremos ordenar por la columna 4. Y, al añadir la opción -n le indicamos que ordene por orden numérico.**)**

Variables de entorno

- En *GNU/Linux*, tenemos una serie de variables de entorno que podemos utilizar para obtener información relativa a la instalación del sistema.
  - La instrucción **env** lista las variables de entorno
  - Ejemplo: para saber la ruta en las que el sistema busca los archivos ejecutables empleamos $PATH
    - Podemos consultar sólo esa variable con **echo $PATH**
    - Podemos modificar con **export Variable = Valor**. Ej: export PATH = /Home/Scripts:$PATH

Variable	Descripción
DISPLAY	Donde aparecen la salidas de X-Windows.
HOME	Directorio personal.
HOSTNAME	Nombre de la máquina.
MAIL	Archivo de correo.
PATH	Lista de directorios donde buscar los programas.
PS1	Prompt.
SHELL	Intérprete de comandos por defecto.
TERM	Tipo de terminal.
USER	Nombre del usuario.

Grupos y usuarios linux

Comandos:

- **Id**: permite averiguar el id del usuario (uid) y de su grupo (gid) entre otras cosas.
- **Logname y whoami**: devuelve el login del usuario activo
- **Adduser**: permite añadir el usuario especificado.
  - Ej: sudo adduser manolito
- **Passwd**: permite cambiar la contraseña del usuario especificado
  - Ej: sudo passwd manolito
- **Userdel**: permite borrar la cuenta del usuario especificado, si ponemos el parámetro –r además borramos los ficheros asociados a ese usuario.
  - Ej: sudo userdel –r manolito
- **Deluser:** permite borrar un usuario y permite configurar con parámetros borrado de ficheros, copias de seguridad, etc.
- **Groupadd**: permite crear un grupo que podemos vincular al usuario con **addgroup**
- **Addgroup:** vincula grupo y usuario
- **Delgroup**: elimina un grupo del sistema (con –only-if-empty nos aseguramos de que no tiene usuarios)

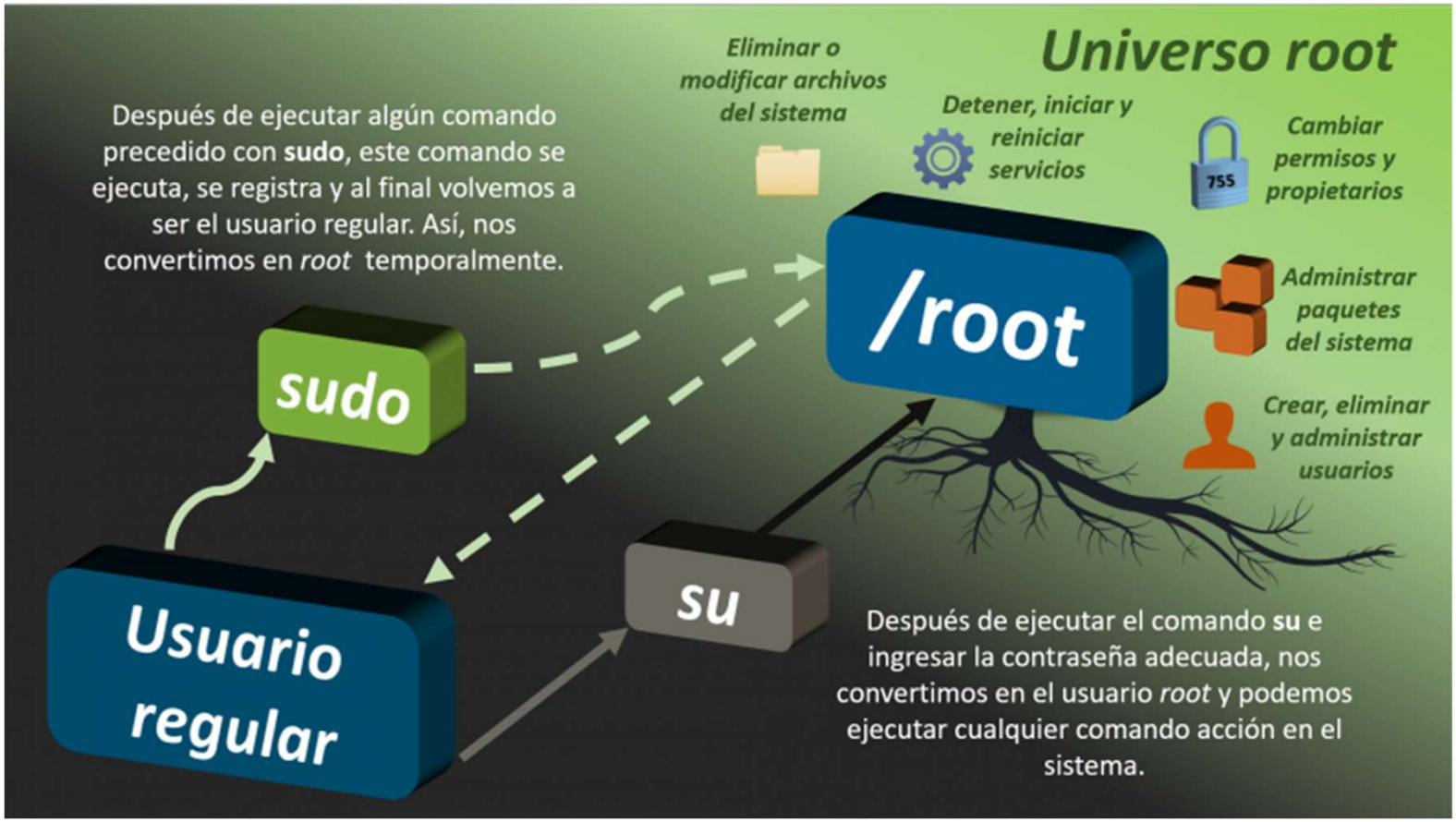
Grupos y usuarios linux

- La identificación de usuario se hace mediante: nombre de usuario (login) + contraseña (password).
- Cada usuario
  - Pertenece a uno o más **grupos** de usuarios
  - Tiene unos permisos de lectura(**r**), escritura(**w**) y ejecución(**x**) sobre programas, ficheros y directorios.
  - Tiene su propio espacio en el directorio **/home.**
- Hay un tipo especial de usuario llamado root o superusuario (**su**) que tiene privilegios para realizar algunas tareas.

Comandos

- **su:** permite convertirnos en otro usuario
  - su miusuario (exit: permite salir de una cuenta de usuario).
  - su, o su – invoca al usuario root
  - su - miusuario inicia un nuevo Shell con las preferencias del usuario (home, Shell, logname, path, etc).
- **sudo:** permite ejecutar una instrucción como superusuario. Requiere de contraseña la primera vez que ejecutamos o pasado un tiempo.

Diferencias sudo y su



Grupos locales

- En *GNU/Linux*, **todas las cuentas** de usuario necesitan pertenecer a un **grupo principal** (también suele llamarse *grupo primario*).
- Cada vez que creamos **una cuenta de usuario**, de forma transparente se **crea también un nuevo grupo** con el mismo nombre.
- Además existen **grupos predeterminados**, que se utilizan para la propia gestión del sistema. Por ejemplo, el uso de algunos servicios (como *mail* o *news*) o de dispositivos (como *disk* o *cdrom*).
- Después de éstos, se encontrarán los grupos que hayamos creado nosotros (su gid > 999).
- En /etc/group encontraremos separados por puntos el nombre del grupo, su contraseña (si no la tiene es ! y si está cifrada x), el identificador y los grupos secundarios.
- El comando groups muestra los grupos del usuario.

Grupos

- En /**etc/group** encontraremos separados por puntos el nombre del grupo, su contraseña (si no la tiene es ! y si está cifrada x), el identificador y los miembros para los que el grupo no es su grupo primario.
- Descripción **/etc/group**:
  - **Group**: nombre de grupo.
  - **Password**: contraseña de grupo.
  - **GID**: identificador de grupo.
  - **Usuario1,usuario2…**: usuarios que pertenecen al grupo.

```
Adm:x:4:syslog,p
Hola:x:1002:manolo
```

- No podemos conectarnos como grupo.
- El password es utilizado para los casos en los que queramos cambiar el grupo principal de pertenencia del usuario sin pertenecer al grupo.
  - gpasswd nomgrupo (nos añadirá nomgrupo como grupo ppal)
- En el caso en el que el usuario quiera poner un grupo secundario al cual pertenece como grupo principal (sólo durante la sesión) utilizaríamos:
  - newgrp grupoSecundario

Usuarios

- El contenido del fichero **/etc/passwd** determina quien puede acceder al sistema de manera legítima y que se puede hacer una vez dentro del sistema.
- Ejemplo línea del fichero:

usuario1:FXWUuZ.vwXttg:500:501:usuario pepito:/home/usuario1:/bin/bash

usuario1:	Nombrede la cuenta (Login)
FXWUuZ.vwXttg:	Clave de acceso encriptada (password)
500:	UID de esta cuenta (si es 0, el usuario es root)
501:	GID del grupo principal al que pertenece la cuenta
usuario pepito:	Nombre del usuario
/home/usuario1:	Directorio de trabajo de usuario1
/bin/bash:	Intérprete de comando (shell) de usuario pepito

Grupos y usuarios linux

Usuario root:

- Por defecto, la distribución Ubuntu bloquea la cuenta root (superusuario).
- La /home de root es el directorio /root
- Podemos activar la cuenta root de la siguiente manera:
  - **De forma puntual: $ sudo su** comando
  - **Podemos desactivarla con dos comandos:**
    - **$ sudo passwd -l root**
    - **$ sudo usermod -p '!' root**
- Por seguridad, es preferible utilizar sudo para la administración del sistema sin activar la cuenta root.

- Podemos ampliar el número de usuarios que tienen permiso como root (sudoers) empleando **visudo** (es un editor para cambiar el fichero /etc/sudoers.d).

Permisos directorios y ficheros

- Los directorios o carpetas tienen una serie de atributos para lectura (r), escritura (w) y ejecución (x) para el usuario (u), el grupo (g) y otros (o).

PERMISO	IDENTIFICA
-	*Sin permiso.*
r	*Permiso de lectura.*
w	*Permiso de escritura.*
x	*Permiso de ejecución.*

Tabla.12.9. *Tipos de permiso.*

Permisos directorios y ficheros

- **CHMOD**: permite cambiar los valores de los permisos para el usuario, el grupo y otros usuarios que no pertenezcan al grupo.
- Mediante máscara. Ejemplo:

  Chmod 775 carpeta

  - 7 en binario es 111 por lo que activamos la lectura, escritura y ejecución
  - 5 en binario es 101 por lo que activamos la lectura y ejecución y desactivamos la escritura
- Cambio de permisos mediante caracteres. Ejemplo:

  Chmod ug=rwxrwx,o=rx carpeta

  - Permite al usuario, grupo y a los otros leer y ejecutar y al usuario y grupo escribir.
  - Si queremos que los permisos sean distintos para el usuario, grupo y los otros, es necesario que separemos con comas y sin espacios.
  - Es equivalente a chmod 775 carpeta

CARÁCTER	ACCIÓN
-	*Elimina derechos.*
+	*Añade derechos.*
=	*Asigna permisos especificados.*

PERMISO	ACTÚA SOBRE
u	*Propietario.*
g	*Grupo al que pertenece el usuario.*
o	*Resto de usuarios.*

Permisos directorios y ficheros

- Además de u, g y o si queremos aplicar algo a todos los usuarios podemos emplear **a** (viene de all)
  - Ej: chmod a+x carpeta, permite a todos ejecutar.
- Ejemplos instrucción CHMOD

ORDEN	RESULTADO
$chmod g+x doc1	*Con esta orden lo que estamos haciendo es otorgar privilegios de ejecución al grupo al que pertenece el archivo llamado doc1.*
$chmod go-wx doc1	*Se quitan los privilegios de escritura y ejecución al grupo y al resto de usuarios del archivo doc1.*
$chmod =x doc1	*Asigna a todos los usuarios el permiso de ejecución. Esta misma orden se podría poner* $chmod ugo+x doc1.
$chmod = doc1	*Quita todos los privilegios a todos los usuarios del archivo doc1.*

Tabla.12.12. *Ejemplos sobre el uso de chmod.*

Umask

- Cuando creamos un fichero se emplean los permisos por defecto 644.
- Cuando creamos un directorio los permisos por defecto son 755
- Para configurar los permisos por defecto empleamos **umask** que por defecto tiene el valor **022** y se emplea de la siguiente manera:

Tipo de ficheros	Operación	Desarrollo	Resultado
Ficheros normales	666 - máscara	(666 – 022 = 644)	644
Directorios y Ficheros ejecutables	777 - máscara	(777 – 022 = 755)	755

- Ejemplos de sintaxis:
  - Umask (consultamos el valor)
  - Umask –S (consultamos el valor del complemento de la máscara)
  - Umask 077 (asignamos valor a la máscara durante la sesión)
  - Umask –S u=rwx, g=rwx, o=rx (los directorios y ejecutables se crearán con permisos 775 y el resto 664)

Cambiar de propietario y grupo

- Sólo root puede cambiar el propietario de un fichero, un usuario puede cambiar el grupo de un fichero si pertenece al nuevo grupo.

- Para cambiar de propietario a un fichero o ficheros:
  - **chown** usuario o UID fich1 [fich2 fcih3...]
- Para cambiar el grupo de un fichero o ficheros:
  - **chgrp** grupo o GID fich1 [fich2 fich3 .....]
- No se modificarán los permisos para los ficheros ni su ubicación.

- El parámetro –R cambia la propiedad de forma recursiva.

- Podemos cambiar usuario y grupo con la misma orden
  - **chown** usuario:grupo fich1 [fich2 fich3...]

Enlaces simbólicos vs Lsicos

- Un **enlace simbólico** es un nuevo nombre asociado a un archivo pero a diferencia de los enlaces físicos, **el enlace simbólico no contiene los datos del archivo**, simplemente apunta al registro del sistema de archivos donde se encuentran los datos. Tiene mucha similitud a un *acceso directo* en Windows.
- Un **enlace físico** es una forma de identificar el mismo contenido con diferentes nombres. Éste enlace **no es una copia separada del archivo anterior** sino un nombre diferente para exactamente el mismo contenido. Es decir, tienen el mismo inodo.
  - Los enlaces físicos o duros sólo se pueden hacer entre ficheros (no directorios), del mismo sistema de ficheros.
- Si borramos el fichero, el enlace físico sigue funcionando mientras que el simbólico no.

- Comandos:
  - Enlace físico: **ln** archivo.txt nuevo_nombre.txt
  - Enlace simbólico: **ln -s** archivo.txt nuevo_nombre.txt

Empaquetar y Comprimir

Comando **empaquetar**:

- tar -cvf archivo.tar /dir/a/comprimir/
- -c : indica a tar que cree un archivo.
- -v : indica a tar que muestre lo que va empaquetando.
- -f : indica a tar que el siguiente argumento es el nombre del fichero.tar.

Comando **desempaquetar** los ficheros .tar:

- tar -xvf archivo.tar
- -x : indica a tar que desempaquete el fichero.tar.
- -v : indica a tar que muestre lo que va desempaquetando.
- -f : indica a tar que el siguiente argumento es el nombre del fichero a desempaquetar.

Comprimir: para comprimir en formato tar.gz, se utiliza el siguiente comando:

- tar -czfv archivo.tar.gz ficheros
- -z : indica que use el compresor gzip

Para **descomprimir** empleamos tar -xzvf archivo.tar.gz

Compresión/descompresión gzip

Los comandos gzip y gunzip permiten comprimir y descomprimir respectivamente uno o varios ficheros.

- Sintaxis:
  - gzip [opciones] <ficheros/directorio>
  - gunzip [opciones] <ficheros/directorio>
- Algunas opciones:
  - -r : dado un directorio comprime todos los ficheros presentes en él recursivamente.
  - -1 a -9 : especifica el grado de la compresión (-1 menor y más rápida -9 mayor y más lenta).
  - -S <sufijo> : permite especificar la extensión (por defecto es gz).
- Ejemplos:
  - **$ gzip -9 *** # Comprime todos los ficheros del directorio actual (su extensión cambia a .gz)
  - **$ gunzip big-file.gz** # descomprime el fichero big-file.gz
- También existen los pares de comandos zip y unzip, y compress y uncompress.