Trabajo Práctico 1

Materia	ISO Práctica
Estado	Done
Fecha	

1. Características de GNU/Linux:
    - **Mencione y explique las características más relevantes de GNU/Linux.**
    - Mencione otros sistemas operativos y compárelos con GNU/Linux en cuanto a los puntos mencionados en el inciso a.
    - ¿Qué es GNU?
    - Indique una breve historia sobre la evolución del proyecto
    - Explique qué es la multitarea, e indique si GNU/Linux hace uso de ella.
    - ¿Qué es POSIX?
2. Distribuciones de GNU/Linux:
    - ¿Qué es una distribución de GNU/Linux? Nombre al menos 4 distribuciones de GNU/-Linux y cite diferencias básicas entre ellas.
    - ¿En qué se diferencia una distro de otra?
    - ¿Qué es debian?
3. Estructura de GNU/Linux:
    - Nombre cuales son los 3 componentes fundamentales de GNU/Linux.
    - Mencione y explique la estructura básica del SO GNU/Linux
4. Kernel:
    - ¿Qué es? Indique una breve reseña histórica acerca de la evolución del Kernel de GNU/Linux
    - ¿Cuáles son sus funciones principales?
    - ¿Cuál es la versión actual? ¿Cómo se definía el esquema de versionado del Kernel en versiones anteriores a la 2.4? ¿Qué cambió en el versionado se impuso a partir de la versión 2.6?
    - ¿Es posible tener más de un Kernel de GNU/Linux instalado en la misma máquina?
    - ¿Dónde se encuentra ubicado dentro del File System?
    - ¿El Kernel de GNU/Linux es monolítico? Justifique.
5. Intérprete de comandos Shell
    - ¿Qué es? ¿Cuál es su función?
    - Mencione al menos 3 intérpretes de comandos que posee GNU/Linux y compárelos entre ellos.
    - ¿Dónde se ubican (path) los comandos propios y externos al Shell?
    - ¿Por qué considera que el Shell no es parte del Kernel de GNU/Linux?
    - ¿Es posible definir un intérprete de comandos distinto para cada usuario? ¿Desde dónde se define? ¿Cualquier usuario puede realizar dicha tarea?
6. Sistema de Archivos (File System)
    - ¿Qué es?
    - Mencione sistemas de archivos soportados por GNU/Linux.
    - ¿Es posible visualizar particiones del tipo FAT y NTFS en GNU/Linux?

- ¿Cuál es la estructura básica de los File System en GNU/Linux? Mencione los directorios más importantes e indique qué tipo de información se encuentra en ellos. ¿A qué hace referencia la sigla FHS?

7. Particiones
   - Definición. Tipos de particiones. Ventajas y Desventajas.
   - ¿Cómo se identifican las particiones en GNU/Linux? (Considere discos IDE, SCSI y SATA).
   - ¿Cuántas particiones son necesarias como mínimo para instalar GNU/Linux? Nómbrelas indicando tipo de partición, identificación, tipo de File System y punto de montaje.
   - Ejemplifique diversos casos de particionamiento dependiendo del tipo de tarea que se deba realizar en su sistema operativo.
   - ¿Qué tipo de software para particionar existe? Menciónelos y compare
8. Arranque (bootstrap) de un Sistema Operativo
   - ¿Qué es el BIOS? ¿Qué tarea realiza?
   - ¿Qué es UEFI? ¿Cuál es su función?
   - Diferencias entre UEFI y BIOS
   - ¿Qué es el MBR? ¿Que es el MBC?
   - ¿A qué hacen referencia las siglas GPT? ¿Qué sustituye? Indique cuál es su formato.
   - → En resumen: BIOS usa MBR, UEFI usa GPT.
     Aunque GPT se adapta para BIOS ←
   - ¿Cuál es la funcionalidad de un “Gestor de Arranque”? ¿Qué tipos existen? ¿Dónde se instalan? Cite gestores de arranque conocidos.
   - ¿Cuáles son los pasos que se suceden desde que se prende una computadora hasta que el Sistema Operativo es cargado (proceso de bootstrap)?
   - Analice el proceso de arranque en GNU/Linux y describa sus principales pasos.
   - ¿Cuáles son los pasos que se suceden en el proceso de parada (shutdown) de GNU/Linux?
   - ¿Es posible tener en una PC GNU/Linux y otro Sistema Operativo instalado? Justifique.
9. Archivos
   - ¿Cómo se identifican los archivos en GNU/Linux?
   - Investigue el funcionamiento de los editores vi y mcedit, y los comandos cat y more
   - Cree un archivo llamado “prueba.exe” en su directorio personal usando el vi. El mismo debe contener su número de alumno y su nombre.
   - Investigue el funcionamiento del comando file. Pruébelo con diferentes archivos. ¿Qué diferencia nota?
10. Indique qué comando es necesario utilizar para realizar cada una de las siguientes acciones. Investigue su funcionamiento y parámetros más importantes:
    - Cree la carpeta ISO2017
    - Acceda a la carpeta (cd)
    - Cree dos archivos con los nombres iso2017-1 e iso2017-2 (touch)
    - Liste el contenido del directorio actual (ls)
    - Visualizar la ruta donde estoy situado (pwd)
    - Busque todos los archivos en los que su nombre contiene la cadena “iso*” (find)
    - Informar la cantidad de espacio libre en disco (df)
    - Verifique los usuarios conectado al sistema (who)
    - Acceder a el archivo iso2017-1 e ingresar Nombre y Apellido
    - Mostrar en pantalla las últimas líneas de un archivo (tail).
11. Investigue su funcionamiento y parámetros más importantes:
    - `(a) shutdown`
    - `(b) reboot`
    - `(c) halt`

(d) locate
(e) uname
(f) gmesg
(g) lspci
(h) at → at [OPTION...] runtime
(i) netstat → netstat [opciones]
(j) mount → mount [opciones] [-t tipo] [-a] [-o opc] dispo punto_montaje
(k) umount → umount [opciones]
(l) head → head [opciones]
(m) losetup → losetup [-d][-e <cifrado>][-o <número de traducción>]
(n) write → write usuario [nombre-tty] mensaje x=numero terminal
(ñ) mkfs → mkfs [-V] [-t fstype] [c] nombreParticion
(o) fdisk (con cuidado) → fdisk [opciones]

12. Investigue su funcionamiento y parámetros más importantes
    Indique en qué directorios se almacenan los comandos mencionados en el ejercicio anterior.

1. Características de GNU/Linux:

Mencione y explique las características más relevantes de GNU/Linux.

Es de **código abierto**, **se redistribuye junto a su código fuente** libre distribución y gratuito. Que sea de código abierto permite estudiarlo, personalizarlo, etc.

Al estar desarrollado por miles de programadores saca actualizaciones más seguido y la corección de errores es rápida.

Es multisuario, multitarea y multiprocesador. Permite el manejo de usuarios y permiso.

En linux **todo es un archivo** y cada directorio puede estar en una partición diferente.

Mencione otros sistemas operativos y compárelos con GNU/Linux en cuanto a los puntos mencionados en el inciso a.

Otros sistemas operativos son Windows y macOS.

Estos códigos son de distribución paga, están diseñados por empresas y hay que pagar para usarlos, **no se redistribuyen junto a su código, es decir, es de código cerrado.** Este tipo de software es patentado y no se redistribuye el código fuente que está cifrado. Debido a esto, la corrección de fallos puede tardar más.

¿Qué es GNU?

Es un **sistema operativo del tipo UNIX**. Se comporta similar a estos aunque no es necesario un certificado de la *Single unix specification.*

Unix es un sistema operativo creado para administrar servidores (inicialmente) cuya interacción era mediante comandos. Diferente a **GNU**, UNIX es software propietario (no libre). Decir que GNU es un sistema *del tipo* UNIX, significa que GNU fue creado para ser usado desde la línea de comandos pero *ser libre.*

El *ser libre* significa que respeta las 4 libertades: libertad para usar, estudiar, copiar y mejorar su código fuente. Esto permite que GNU se use para una gran variedad de plataformas y herramientas que no necesariamente son exclusivas de la administración de servidores. UNIX, en cambio, se ha mantenido en ese rubro.

Indique una breve historia sobre la evolución del proyecto

Iniciado por ***Richard Stallman*** en **1983** con el fin de crear un *Unix libre*. Para asegurarse de que fuese libre, se necesitó crear un marco regulatorio conocido como **GPL** (General Public License de GNU).

En **1875** Stallman crear la **FSF** (Free software foundation) con el fin de financiar el proyecto GNU.

En **1990** GNU ya contaba con un editor de texto (Emacs), un compiladores (GCC) y gran cantidad de bibliotecas que componen un Unix típico. Faltaba el componente principal: el **núcleo Kernel**.

> Se venía trabajando en un núcleo conocido como TRIX, es en **1988** que se decide abandonarlo por su complejidad. En ese momento se adopta como base el núcleo MACH (para crear GNU Hurd que no prosperó). Pero en **1991** *Linux Torvalds* (que venía trabajando desde 1991) crea el núcleo Kernel, el cuál se distribuiría bajo licencia GPL.

En **1992** *Torvalds y Stallman* fusionan ambos proyectos, de ahí nace **GNU/Linux**.

Explique qué es la multitarea, e indique si GNU/Linux hace uso de ella.

Un sistema **multitarea** permite al usuario estar realizando varias tareas al mismo tiempo (no es lo mismo a que tenga varios núcleos, ya que esto habla de la administración a nivel SO y no HW). GNU/Linux **es** multitarea.

¿Qué es POSIX?

POSIX = *Portable Operating System Interface*

Son un conjunto de estándares de la IEEE que buscan facilitar y estandarizar las reglas de la **interoperabilidad** de un SO.

2. Distribuciones de GNU/Linux:

¿Qué es una distribución de GNU/Linux? Nombre al menos 4 distribuciones de GNU/-Linux y cite diferencias básicas entre ellas.

Las distribuciones son SO (software) que están basados en el núcleo de Linux . Estas distribuciones se diferencia en alguno de sus paquetes de software, compuestos (mayoritariamente) de software libre.

Distribuciones de GNU/Linux: Debian, Mint, Astro, Fedora, Ubuntu, Kubuntu.

- Debian: Muy estable y 100% libre. Su sistema de paquetes es `.deb` y su gestión de paquetes es APT . De esta nacen otros SO como Ubuntu, Kubuntu. ç
- Ubuntu: Lo detesto y lo amo (?). Es más fácil de usar que el resto, pero cada dos por tres tiene error en los paquetes. Antes tenía Unity como entorno, pero después se cambio a gnome y ahora es un Gnome raro del tipo Unity.
    - Kubuntu es un ubuntu con escitorio KDE en lugar de gnome.
- Mint: Basado en Ubuntu. Tiene fácil instalación de paquetes. Su escritorio Cinnamon (que igual es un gnome) es muy personalizable.
- Fedora: Tiene gnome y un sistema de paquetes menos útil que ubuntu, pero se ve lindo.

¿En qué se diferencia una distro de otra?

Todas tienen en común el **Kernel** (uno de los puntos en los que gira linux) pero todo lo demás puede ir variando. Varian: las herramientas, la shell, el sistema de archivos, la GUI. Varían por personalizaciones o también porque están creadas desde 0.

¿Qué es debian?

Debian es una distribucion de Linux. Usan un núcleo kernel, pero están trabajando en desarrollar debian para otros (como el Hurd, que trabaja sobre un micronúcleo)

- En 1993 se fundó el proyecto Debian.
- En 1994 el proyecto GNU patrocinó Debian.

3. Estructura de GNU/Linux:

Nombre cuales son los 3 componentes fundamentales de GNU/Linux.

- El **kernel** (núcleo). Es un programa que se ejecuta apenas se prende la CPU, recubre al hardware por completo, gestionándolo y protegiendo a la CPU. Se encarga de planificar, gestionar y administrar los dispositivos.
- La **shell**, proporciona una interfaz para que el **usuario se comunique con el núcleo**, permite interpretar las ordenes del primero y enviarlas al segundo.
- El **sistema de archivos** que organiza el almacenamiento de archivos, que están organizados en directorios.

Mencione y explique la estructura básica del SO GNU/Linux

El **kernel** es el componente principal cuya función es ejecutar programas y gestionar el hardware, administrando la memoria, el CPU y la E/S. La **shell** permite la comunicación entre el usuario y el sistema operativo y el **filesystem** organiza el modo en que se almacena y se accede a los archivos.

4. Kernel:

¿Qué es? Indique una breve reseña histórica acerca de la evolución del Kernel de GNU/Linux

La primera y única capa que cubre al hardware. Es una porción de código que siempre está en memoria principal, ya que es el encargado de la administración y del vector de interrupciones. Arriba del Kernel se encuentra el sistema operativo y es el encargado de ir instalando las otras cosas.

Breve reseña histórica

- En 1991 Linus Torvalds inicia la programación de un Kernel Linux basado en Minix (clon de Unix desarrollado por Tenembaum en 1987 con el fin de crear un S.O. de uso didáctico)
- El 5 de octubre de 1991, se anuncia la primera versión “oficial” de Linux (0.02).
- En 1992 se combina su desarrollo con GNU, formando GNU/Linux.
- La versión1.0 apareció el 14 de marzo de 1994.
- Desarrollo continuado por miles de programadores al rededor del mundo
- En mayo de 1996 se decide adoptar a Tux como mascota oficial de Linux
- En julio de 1996 se lanza la versión 2.0 y se define la nomenclatura de versionado. Se desarrollo hasta febrero de 2004 y termino con la 2.0.40
- En enero de 1999 se lanza la versión 2.2, que provee mejoras de portabilidad entre otras y se desarrolla hasta febrero de 2004 terminando en la versión 2.2.26
- En 2001 se lanza la versión 2.4 y se deja de desarrollar a fines del 2010 con la 2.4.37.11
    - La versión 2.4 fue la que catapulto a GNU/Linux como un SO estable y robusto. Durante este perıodo es que se comienza a utilizar Linux mas asiduamente.
- A fines del 2003 se lanza la versión 2.6
    - Esta versión ha tenido muchas mejoras para el SO dentro de las que se destacan soporte de hilos, mejoras en la planificación y soporte de nuevo hardware
- El 3 de agosto de 2011 se lanza la versión 2.6.39.4 anunciándose la misma desde meses previos como la ´ultima en su revisión
- El 17 de julio de 2011 se lanza la versión 3.0
    - No agrega mayores cambios. La decisión del cambio son los 20 años del SO y no superar los 40 números de revisión
    - Totalmente compatible con 2.6
    - La ´ultima versión estable es la 4.7.1 (agosto de 2016)

¿Cuáles son sus funciones principales?

¿Cuál es la versión actual? ¿Cómo se definía el esquema de versionado del Kernel en versiones anteriores a la 2.4? ¿Qué cambió en el versionado se impuso a partir de la versión 2.6?

La última versión del kernel es la 5.15.0-47

- La version 2.4 fue la que catapulto a GNU/Linux como un SO estable y robusto. Durante este periodo es que se comienza a utilizar Linux mas asiduamente

- Antes de la serie de Linux 2.6.x, los números pares indicaban la versión “estable” lanzada. Los números impares, como la serie 2.5.x, son versiones de desarrollo, es decir que no son consideradas de producción.

¿Es posible tener más de un Kernel de GNU/Linux instalado en la misma máquina?

Sí, es posible tener dos kernels en la misma máquina, pero al arrancar la computadora **sólo va a ejecutarse uno.**

Los kernels van a tener que estar en **distintas particiones** del disco duro y, cuándo elijamos desde que partición arrancar, se llevará el kernel correspondiente a memoria para iniciar con todo.

¿Dónde se encuentra ubicado dentro del File System?

Se encuentra en `/boot`

¿El Kernel de GNU/Linux es monolítico? Justifique.

Es un núcleo **monolítico híbrido** dónde los drivers y el código del Kernel se ejecutan en modo privilegiado. Su hibridez es por la **capacidad de cargar y descargar funcionalidad a través de módulos**.

5. Intérprete de comandos Shell

¿Qué es? ¿Cuál es su función?

Es un **programa** que provee la interfaz necesaria para que el usuario acceda a los servicios del **SO**.

Mencione al menos 3 intérpretes de comandos que posee GNU/Linux y compárelos entre ellos.

- **Bash**: Es el mas conocido porque viene por defecto en la mayoría de las distribuciones de GNU/Linux.
    - Permite la edición de líneas de comando.

    - Tiene un tamaño ilimitado del historial de comandos
    - Cumple funciones de control, funciones y cálculos aritméticos.
    - **Soporta completar nombres de archivos y comando con la tecla tab**
    - Su símbolo de sistema es `nombreUsuario@nombreEquipo`
- **BourneShell:** Es el shell estándar.
    - Su símbolo de sistema es el `$`
- **KornShell**: Amplia la shell añadiendo historial de orden y edición en línea de órdenes.
    - Soporta arreglos.
    - Soporta prints

¿Dónde se ubican (path) los comandos propios y externos al Shell?

PATH tiene la lista de directorios separados por `:` en los que la shell busca los comandos externos, cada `:` representa un “lugar”

Los **comandos propios** de la shell son nativos de esta, por lo tanto no corresponde a ningún archivo almacenado en disco.

`cd` `pwd` `echo` `time` `exec` `exit`

`ls` `miscript` `cp` `mv` `rm`

Los **comandos externos** no son comandos que la shell sepa ejecutar, así que para hacerlo tiene que saber su localización en el disco duro. Para esto debe ser un archivo binario ejecutable o un archivo con formato de texto que represente un script de comandos. Los comandos externos se ejecutan por una **shell hijo** que actúa de intermediario. La mayoría de los comandos externos están en `/usr/bin`

¿Por qué considera que el Shell no es parte del Kernel de GNU/Linux?

El **kernel** es el núcleo y trata de ser lo más chico posible, con interfaces mínimas pero **muy complejas** (en resumen: mientras menos gente meta mano porque le parezca difícil, menos chance de hacer cagadas)

Entonces: si el kernel falla lo más probable es que se tenga que reiniciar por completo el equipo. Así que separándolo de la shell, logramos de que si metemos mano en la shell y algo falla no tengamos que rebootear todo, sólo la shell.

¿Es posible definir un intérprete de comandos distinto para cada usuario? ¿Desde dónde se define? ¿Cualquier usuario puede realizar dicha tarea?

Es posible que cada usuario tenga una shell diferente. Desde la terminar podemos ejecutar una serie de comandos que sirven para determinar **qué** shell vamos a usar, así que cada usuario puede determinarlo.

Para cambiar de shell es necesario introducir una contraseña.

6. Sistema de Archivos (File System)

¿Qué es?

El **fileSystem** es el sistema de almacenamiento de un dispositivo de memoria. **Representa qué estructura y organización en la escritura, la búsqueda, la lectura, el almacenamiento, la edición y la eliminación tendrán los archivos**. El objetivo de esta organización es que el usuario pueda identificar los archivos sin lugar a errores y acceder a ellos lo más rápido posible.

Características del FyleSistem:

- Hay una **convención** para **nombrar** los archivos
- Los atributos de archivos
- Controles de acceso al mismo

El **fylesistem** es un **muy importante, ya que actúa como una interfaz entre el sistema operativo y los dispositivos conectados al equipo, tanto internos como externos.**

Mencione sistemas de archivos soportados por GNU/Linux.

Los soportados por GNU/Linux son: minix, ext, ext2, ext3, ext4, xia, msdos, umsdos, vfat, proc, nfs, iso9660, hpfs, sysv, smb, ncpfs, FAT, FAT32, NTFS.

¿Es posible visualizar particiones del tipo FAT y NTFS en GNU/Linux?

Cuándo instalamos Linux, lo usual es que tengamos alguna partición en formato **NTFS** o **FAT.** Estas particiones corresponden al sistema operativo windows, por lo que linux no permite ver el contenido de estas particiones (por la limitación en el tipo de archivo) → para verlo tenemos que montar la partición **NTFS o FAT** en la que está windows → Linux lo hace automáticamente al verlo.

NTFS: Es ideal para ser usado en discos duros externos, soporta particiones y el almacenamiento de ficheros grandes. Pero Mac y algunas distos de Linux no lo quieren.

¿Cuál es la estructura básica de los File System en GNU/Linux? Mencione los directorios más importantes e indique qué tipo de información se encuentra en ellos. ¿A qué hace referencia la sigla FHS?

`/` → Esta es **la estructura básica** el tope de los directorios.

`/home` → Acá se almacenan los archivos de usuario

`/var` → Acá está la información que varia de tamaño (archivos de registros, BD)

`/etc` → Archivos de configuración, scripts de arranque, etc

`/bin` → Archivos binarios y ejecutables

`/dev` → Enlace a dispositivos

`/usr` → Aplicaciones de usuario

`/root` → Directorio personal de usuario root (superusuario)

`/boot` → Fichero de configuración del arranque, núcleos y otros ficheros necesarios para el arranque (boot) del equipo.

FHS → FileSystem herarchy standar. Los los sistemas operativos del tipo Unix siguen este estandar para organizar su filesystem.

7. Particiones

Definición. Tipos de particiones. Ventajas y Desventajas.

Una partición es el **nombre que se le da a una división presente en una sola unidad física de almacenamiento de datos**. Tener varias particiones es como tener varios discos duros en uno solo, cada uno con un sistema de archivos y funcionando de manera distinta.

Los tipos de partición son:

- **Primaria**: Depende de la tabla de particiones y son las que **detecta la máquina al arrancar**, por eso mismo ahí se instala el **sistema operativo.** Es la división cruda del disco y, como máximo, puede haber 4 (debido al MBR)
- **Extendida**: Fue la solución para tener más de 4 particiones en un sólo disco, pero acá no se puede instalar un sistema operativo. **Sólo son útiles para guardar datos y sólo puede** haber **una partición extendida,** pero dentro de esta podemos hacer tantas otras como queramos.

→ Hasta acá, si usamos una partición del tipo extendida sólo vamos a poder tener 3 particiones primarias y 1 extendida.

- **Lógica:** Son particiones que se hacen dentro de una partición extendida, sólo se le debe asignar un tamaño y un tipo de FileSystem. Funciona como si fuese un dispositivo diferente.

¿Cómo se identifican las particiones en GNU/Linux? (Considere discos IDE, SCSI y SATA).

En discos **IDE** las particiones empiezan con `/dev/` y luego sigue la identificación dependiendo de la partición que sea:

- **HDA** (primer disco duro IDE)
  - HDA1 (primera partición del primer disco duro IDE)

    - HDA2 (segunda partición del primer disco duro IDE)
    - HDA3 (tercera partición del primer disco duro IDE)
    - HDA4 (cuarta partición del primer disco duro IDE) → si tiene, claro
    - **HDA5** (primera partición **lógica** de una partición extendida del primer disco duro IDE)
- **HDB** (segundo disco duro IDE)
    - Y lo mismo… .

Básicamente en los discos **IDE** se identifica primero los discos duros (HDA, HDB, HDC) y luego si corresponde a una partición primaria (números del 1 al 4) o a una lógica de una partición extendida (5 en adelante, siendo 5 la primera).

En los discos **SCSI y SATA** también empieza con `/dev/` y sigue con la identificación dependiendo de la partición. **SDA, SDB** dependiendo del disco duro y luego con números del 1 al 4 para las particiones primarias y del 5 para las lógicas.

¿Cuántas particiones son necesarias como mínimo para instalar GNU/Linux? Nómbrelas indicando tipo de partición, identificación, tipo de File System y punto de montaje.

Linus necesita **mínimo una partición para funcionar,** que es la `/` , corresponde al área del sistema de archivos:

- es una partición primaria,
- su filesystem es ext4 ,
- el punto de montaje es `/`
- y está en el HDA1 o el SDA1.

Se aconseja utilizar otras dos particiones para el `/home` y para el área de SWAP.

Ejemplifique diversos casos de particionamiento dependiendo del tipo de tarea que se deba realizar en su sistema operativo.

??????????????????

¿Qué tipo de software para particionar existe? Menciónelos y compare

- **Gparted**: para GNOME. Crea, elimina, redimensiona, inspecciona y copia particiones. Es buena para reacomodar los discos.
- **KDE Partition manager**: Hace uso de la biblioteca GNU parted, permite
- **Fdisk**: Está incluido en todas las distros de Linux pero es multiplataforma (así que permite hacerlo con más de 90 sistemas de archivos diferentes). Permite administrar las particiones al igual que los demás.

administrar particiones (crear, borrar, comprobar, redimensionar, copiarla, etc)

- **Cfdisk**: Se usa mediante la línea de comandos, intenta leer tablas de particiones del disco y muestra las encontradas.

8. Arranque (bootstrap) de un Sistema Operativo

¿Qué es el BIOS? ¿Qué tarea realiza?

BIOS significa **Binary Input Output System** y es un chip o circuito integrado que en su interior tiene una rutina de software que pone en funciona el resto del hardware de la placa base. **Es el primer programa que se ejecuta al prender la máquina**. La información para esta rutina se encuentra almacenada en **otro chip del tipo CMOS ubicada en la misma placa base**, para que estos datos no se pierdan tiene una batería para alimentarse.

Su propósito es iniciar, testear y configurar el funcionamiento del hardware (RAM, discos duros y otras placas). Una vez terminado esto, se carga el gestor de arranque para que comience a ejecutarse el sistema operativo de la máquina.

> **Si la BIOS** detecta algún problema, no permite que se llegue a la instancia de arranque hasta que este sea solucionado.

Unified extensible firmaware interface, un **BIOS** con esteroides.

¿Qué es UEFI? ¿Cuál es su función?

Es una interfaz especial que se encarga de arrancar la placa base y los componentes de hardware relacionados con esta. Esta interfaz **carga el gestor de arranque** (bootloader) en memoria principal y **este se encarga del resto de la rutina de arranque**.

Para poder usar la interfaz UEFI, la máquina necesita un **firmware especial en la placa base**, ya que al encenderse la máquina el firmware usa la interfaz UEFI como una capa operativo que actúa entre el firmware y el sistema operativo. Para que el UEFI siempre este ahí y arranque antes que la máquina, se implementa de forma permanente en la placa base un chip de memoria.

Diferencias entre UEFI y BIOS

BIOS

- Diseño rustico, sólo funciona el teclado
- Se ejecuta en 16 bits

UEFI

- Se conecta a internet para actualizarse
- Se ejecuta en 32 o 64 bits
- Arranque más rápido
- Mejora la seguridad con su funcionalidad de **secure boot, evitando**

el inicio de so no autenticados.

- Se carga desde cualquier dispositivo de memoria no volátil.
- Se le puede añadir herramientas de tercero.

¿Qué es el MBR? ¿Que es el MBC?

MBR = Master boot record

MBC = Master boot code

El **MBR** es un **sector reservado en el disco físico** (cilindro 0, cabeza 0, sector 1). Está en **todos los discos.** Se encarga de **iniciar el programa de arranque** para el sector de arranque de esa partición, en este sector se identifica dónde se encuentra el sistema operativo y habilita la información de inicio que se cargará en la RAM.

Pesa 512 bytes de los cuáles:

- Los primeros 446 bytes es el **MBC**
- La tabla de particiones, esto ocupa 64 bytes.
- Hay 2 bytes libres o que se ocupan con la firma del MBR.

El **MBC** es una parte del MBR y **realiza el primer conjunto de funciones en el proceso de arranque.** Arranca el sistema operativo:

1. Examina la tabla de particiones
2. En el *primary disk* hay una partición marcada como **booteable** (es decir que funciona para el arranque de la PC)
3. Ejecuta lo que está en el primer sector de la partición booteable
4. Transfiere el control al código ejecutable en el sector de arranque.

¿A qué hacen referencia las siglas GPT? ¿Qué sustituye? Indique cuál es su formato.

Global unique identifier → proviene de que el sistema le asocia un identificador global única a cada partición.

GPT no es más que una tabla de particiones implementada por los sistemas **UEFI**. Las limitaciones del **GPT** son las del disco duro y el sistema operativo, ya que trata de ponerle un identificador único a cada partición.

Por ejemplo, Windows tiene un límite de 128 particiones primarias GPT.

Su **estructura** es nomenclada de la siguiente manera.

- **LBA0** → acá hay un trozo de código MBR para proporcionar compatibilidad con el sistema **BIOS**.

- **LBA1** → se almacena información sobre los bloques de disco que los usuarios pueden usar, además del número y el tamaño de particiones que existen. Acá está el **GUID** del disco y dónde está la tabla secundaria (**backup**). A lo último contiene una comprobación CRC32 para que el EFI verifique que todo está bien y proceder con el arranque,
- **LBA2 al LBA33** se almacenan las entradas de particiones correspondientes. En total cada bloque lógico tiene 512 bytes, por lo tanto puede almacenar información de hasta 4 particiones. En cada una se almacena:
  - el tipo de partición (16 bytes)
  - el GUID único de partición (16 bytes)
  - otra información de hasta 128 bytes.

→ En resumen: BIOS usa MBR, UEFI usa GPT. Aunque GPT se adapta para BIOS ←

¿Cuál es la funcionalidad de un "Gestor de Arranque"? ¿Qué tipos existen? ¿Dónde se instalan? Cite gestores de arranque conocidos.

Gestores conocidos: GNU GRUB, LILO, BURGS, Syslinux.

Un gestor de arranque (**bootloader**) es un software que carga en memoria el sistema operativo. Para esto, el bootloader se ejecuta al arrancar un dispositivo usando algún medio que sea booteable. Este medio que es booteable recibe la información desde el firmware de la máquina (BIOS o UEFI).

El bootalder puede alocarse en un medio extraíble (CD, DVD, USB, disco duro) y, dependiendo de configuración de nuestra BIOS o UEFI, estos dispositivos van a ser dónde intente encontrarlo la primera vez. Después continúa al disco duro.

El **MBR** se encuentra en el disco duro, dónde también está la tabla de particiones del medio de almacenamiento. Su lugar en el disco puede ser cualquiera de estos dos:

- El famoso espacio 0, la pista 0 del disco, blabla.
- En una partición específica del medio de arranque.

¿Cuáles son los pasos que se suceden desde que se prende una computadora hasta que el Sistema Operativo es cargado (proceso de bootstrap)?

1. La computadora se prende, lanza el **boostrap** de la BIOS o el UEFI. El boostrap está en el chip de la BIOS, así que se denomina **control del BIOS**.
   a. Se prueba el hardware
   b. El BIOS se desplaza la dirección de inicio al controlador **DMA** y **luego carga** el **MBR**.
2. En este paso el **bootstrap** necesita localizar y copiar el sistema operativo a la RAM.

a. El orden que el boostrap lo busca depende de cómo tengamos configurada la BIOS (primero diskette, después el disco duro y después los CD)
b. Encuentro el registro de inicio del SO y copia dicho registro a la RAM.
c. Pasa el control del proceso de inicio al registro de inicio.
d. El registro busca los archivos en el disco duro y localiza el resto del SO.
e. A medida que los archivos se van encontrando y cargando en la RAM, el registro de inicio ya no es necesario.
f. el SO pasa de estar en el disco duro a controlar el proceso de inicio.

3. El SO busca los archivos de configuración del hardware que son específicos para la computadora. Se cargan los drivers.

Analice el proceso de arranque en GNU/Linux y describa sus principales pasos.

1. El paso de la **BIOS**: busca el cargador de arranque, lo pasa a la memoria y le da el control →
2. le pasa el control al **MBR,** que estaba en el primer sector de la unidad de arranque (normalmente en: `/dev/sda` o `/dev/had` dependiendo de nuestro disco). El *MBR* busca y ejecuta el **gestor de arranque** (GRUB) →
3. el **GRUB** en el caso de que tengamos más de un SO, las entradas están en este archivo y nos ofrece la posibilidad de ejecutar el que querramos. Vemos la pantalla del GRUB en el arranque del sistema y tenemos que elegir (normalmente hay seteada una opción por default para el tiempo, cuya opción se encuentra en `grub.cfg`)
4. Bien, arranca el SO así que **arranca el Kernel**, al iniciarse se monta el sistema de archivos raíz y se ejecuta el `/sbin/init`, como `init` es lo primero que ejecuta el Kernel de linux, siempre le corresponde el número “1” como ID de proceso (PID) (se verifica con el comando `ps-eflgrep init`
5. El `init` comprueba el archivo e indica el nivel de ejecución, esto puede variar dependiendo de las circunstancias y como se cargó el sistema. Además carga un par de herramientas. Hay 7 niveles de ejecución
   a. 0 - parar
   b. 1 - usuario exclusivo
   c. 2 - modo multiusuario sin acceso NFS (sistema de archivos de red)
   d. 3 - modo multiusuario sin restricción
   e. 4- reservado (no se usa casi)
   f. 5 - X11 (sistema de ventanas X)
   g. 6 - reiniciar

6. **RUNLEVEL,** dependiendo de su nivel de ejecución, el sistema iniciará las aplicaciones desde los siguientes directorios:

- Run level 0 – `/etc/rc.d/rc0.d/`
- Run level 1 – `/etc/rc.d/rc1.d/`
- Run level 2 – `/etc/rc.d/rc2.d/`
- Run level 3 – `/etc/rc.d/rc3.d/`
- Run level 4 – `/etc/rc.d/rc4.d/`
- Run level 5 – `/etc/rc.d/rc5.d/`
- Run level 6 – `/etc/rc.d/rc6.d/`

¿Cuáles son los pasos que se suceden en el proceso de parada (shutdown) de GNU/Linux?

`shutdown` es el comando para apagar, suspender o reiniciar el SO en Linux.

1. Se **desmontan todos los sistemas de archivos** (excepto el raíz `/`)
2. Se finaliza (`kill`) los procesos de los usuarios
3. Se desmonta el sistema de archivos raíz (`/`)
4. `INIT` muestra un mensaje indicaidno que se puede apagar la máquina, ahí es que se baja el interruptor del suministro eléctrico.

¿Es posible tener en una PC GNU/Linux y otro Sistema Operativo instalado? Justifique.

Sí, porque se puede tener dos SO en un mismo disco gracias al poder místico de las ✨ p a r t i c i o n e s ✨, las cuáles son tomadas como discos diferentes y sobre las que se pueden poner distintos SO, ya que cada una va a tener su MBR, bootloader y kernel.

9. Archivos

¿Cómo se identifican los archivos en GNU/Linux?

Se identifican con su nombre, ya que en Linux no hay ningún formato estándar para esto, mientras no tenga un `/` y tenga menos de 256 caracteres, va joya.

Investigue el funcionamiento de los editores vi y mcedit, y los

El **editor VI** es un editor de texto que maneja en memoria el texto entero de un

comandos cat y more

archivo. Clásico de UNIX, se puede usar en cualquier tipo de terminal.

El editor **VI** tiene tres estados:

- **Modo comando.** Se encuentra en este modo siempre que se inicia, las teclas ejecutan acciones (comandos) que permiten mover el cursor, ejecutar comandos, salir y guardar cambios.
- **Modo Inserción.** Con comandos se inserta texto.
- **Modo linea.** Se escriben comandos en la última línea al final de la pantalla.

El **editor mcedir** es un administrador de archivos para Linux (aunque también está para windows). Es una aplicación para la terminal que permite: copiar, mover, eliminar archivos y directorios, buscar archivos, ejecutar comandos en la subshell, tener un visor internos y editor incluido.

El comando `cat` deriva de la palabra con*cat*enar y permite crear, fusionar e imprimir archivos en pantalla.

```
cat > filename.txt #crea un archivo, para agregar lineas de texto solo hay que apretar enter. Al
terminar se apreta CNTRL + D para salir.
cat filenamte.txt #lee el contenido de un archivo y lo muestra en consola.
cat filename.txt | more #para desplazarse por archivos grandes
cat *.txt # para mostrar el contenido de mas de un archivo
cat source.txt > destination.txt #se usa el ">" para redirigir la salida a otro archivo, si exis
te lo sobreescribe y si no existe lo crea.
cat source.txt >> destination.txt # agrega el contenido del archivo destino " >> "
cat source1.txt source2.txt > destination.txt # concatena todos en el destino
cat -s filename.txt #suprime las líneas vacías repetidas y ahorra espacio en pantalla
tac filename.txt # el archivo aparece en orden inverso
cat -v filename.txt # se muestran los carácteres no imprimibles
```

El comando `more` permite mostrar el resultado de la ejecución de un comando en la terminal de a una página a la vez. útil para comandos como `ls` o `du` que causan un gran desplazamiento.
ps -ef | more #la pantalla se llena con una lista de datos, pero para pasar a la siguiente tenes que apretar la barra espaciadora y la q para salir. Al final muestra un - - more - -

```
more <nombre del archivo> #  para leer de a una pagina a la vez
more -p <nombre del archivo> # para que la pantalla se borre y pase a la pagina siguiente, no ha
y desplazamiento
more -c <nombre del archivo> # borra las líneas a medida que se muestran.
```

Cree un archivo llamado “prueba.exe” en su directorio personal usando el vi. El mismo debe contener su número de alumno y su

nombre.

Investigue el funcionamiento del comando file. Pruébelo con diferentes archivos. ¿Qué diferencia nota?

El comando `file` dice si un objeto es un directorio o un archivo. La sintaxis es:

`file -OPCION nombre_archivo/directorio`

Y las opciones son:

- `-c` → se fija si tiene errores de formato
- `-h` → ni sigue enlances simbólicos
- `-m` → usa mfile como archivo alternativo
- `-f` → ffile tiene una lista de archivos a examinar.

Al usarlo retorna “ASCII text”

10. Indique qué comando es necesario utilizar para realizar cada una de las siguientes acciones. Investigue su funcionamiento y parámetros más importantes:

Cree la carpeta ISO2017

`mkdir` → `mkdir/ruta/nuevoDirectorio`

`mkdir` con la la ruta y el nombre que quiero ponerle a la carpeta. Por ejemplo:

```
mkdir /home/letizia/Documentos/ISO2017
```

Acceda a la carpeta (cd)

`cd` es el comando para acceder a las carpetas. Y es `cd directorio`

Cree dos archivos con los nombres iso2017-1 e iso2017-2 (touch)

El comando, justamente `touch`. Particularmente el

```
-touch iso2017-1 # cuando se trata de un sólo parámetro
-touch iso2017-1 iso2017-2 # y así para múltiples archivos
-touch iso2017{1..2} # esto va a generar automáticamente los nombres de los archivos con el núme
ro. Si quisiera hacer del 1 al 20, pondría {1..20}
```

Liste el contenido del directorio actual (ls)

`ls {param} {directorio}`

Y los parámetros son:

- `ls -a` → Lista todos los ficheros, incluyendo los ocultos.
- `ls -l` → Lista todos los ficheros en formato de una columna.
- `ls -t` → Lista todos los ficheros por orden de la fecha de modificación (o cualquiera sea la marca de tiempo)
- `ls -h` → Añade una letra indicando el tamaño para facilitar la lectura.

Visualizar la ruta donde estoy situado (pwd)

`pwd {param}`

Busque todos los archivos en los que su nombre contiene la cadena “iso*” (find)

`find / -name "iso"`

A ver, la sintaxis básica de esto es `find <directorioDondeArrancar> <options> <TerminoAbUSCAR>`, dónde:

- `<directorioDondeArrancar>` es el punto de origen dónde se desea iniciar la búsqueda, también puede ser remplazado por `/` para buscar por todo el sistema, `.` para buscar en la carpeta dónde se está parado, `~` para buscar desde el home.
- `<options>` se usa para indicar que voy a usar para buscar del (nombre, tipo, fecha, etc)
- `<terminoABuscar>` es el término de la búsqueda relevante, si busco una partecita que está concatenada, lo pongo entre comillas.

Informar la cantidad de espacio libre en disco (df)

`df -h`

Tiene diferentes parámetros

```
df -h# te mostrará el resultado en un formato legible por humanos.
df -m# esta línea de comando se utiliza para mostrar la información del uso del sistema de archi
vos en MB.
df -k# para mostrar el uso del sistema en KB.
df -T# esta opción mostrará el tipo de sistema de archivos (aparecerá una nueva columna).
df –ht /home# te permite ver información de un sistema de archivos específico en un formato legi
ble (en este caso el sistema de archivos /home).
df –help# listará otros comandos útiles que puedes usar, con sus descripciones.
```

Verifique los usuarios conectado al sistema (who)

`who`

Acceder a el archivo iso2017-1 e ingresar Nombre y Apellido

Para esto se tiene que usar el vim

```
-vi iso2017-1 # para entrar al vim
-i # para acceder al modo edición
Ballestero Letizia
```

```
Esc # para salir del modo edición
:wq! # hace el guardado del archivo y después sale de la Vim
```

`tail nombreArchivo`

Mostrar en pantalla las últimas líneas de un archivo (tail).

Tail va a imprimirme las últimas 10 líneas del archivo, para pedirle que sea otro el número tengo que poner: `tail -n* archivo` dónde el * representas el número de líneas que quiero escribir.

11. Investigue su funcionamiento y parámetros más importantes:

`(a) shutdown` Apaga, reinicia y detiene el sistema.

`shutdown options time wall`, dónde **options** puede ser una cadena de caracteres, time los detalles de la hora y wall es el mensaje para notificar a los usuarios que se apagará el sistema.

`(b) reboot` Diseñado para reiniciar el equipo

`reboot [-d | -f | -i | -n | -w]`

- -n → no sincroniza antes de reiniciar
- -w → no reinicia, pero escribe el registro wtmp (`/var/log/wtmp`)
- -d → no escribe en este registro
- -f → fuera el reboot pero no llaman al shutdown
- -i → apaga todas las interfaces de red, junto antes de reiniciar el sistema
- -h → pone todos los discos en modo de espera antes de detenerse o de apagarse
- -p → halt es llamado como powerOff

`(c) halt` También se usa para apagar la máquina

`halt [-d | -f | -h | -n | -i | -p | -w]`

- -n → no sincroniza antes de reiniciar
- -w → no reinicia, pero escribe el registro wtmp (`/var/log/wtmp`)
- -d → no escribe en este registro
- -i → apaga todas las interfaces de red, junto antes de reiniciar el sistema
- -h → pone todos los discos en modo de espera antes de detenerse o de apagarse

`(d) locate`

Es la forma más práctica de buscar un archivo en Linux. Hace las búsquedas haciendo uso de una base de datos dónde se encuentra toda la información y las rutas de archivos en el sistema. No va partición por partición.

- Es case sensitive
- se instala con `sudo apt install locate`
- se actualiza la BD con `sudo updatedb`
- se busca con: `locate nombreArchivo` y nos aparece la ruta.

`(e) uname`

Muestra la info del kernel y del SO. Su sintaxis es `uname [parametros]`

```
Parámetros
-s: nombre del kernel
-r: version del kernel
-v: version del SO
-o: nombre del SO
-n: nombre del host (nombre del sistema en la red)
-m: tipo de arquitectura
-a: toda la info de arriba junta
```

`(f) gmesg`

Escribe los mensajes del kernel en una forma bonita. Obtiene los datos leyendo el buffer del anillo del kernel. Su sintaxis es `gmseg [parametros].` **Parametros**:

```
-C, --clear Borra el «buffer» circular del núcleo.
-c, --read-clear Lee y borra todos los mensajes.
-D, --console-off Desactiva la impresión de mensajes por consola.
-E, --console-on Activa la impresión de mensajes por consola.
-F, --file <fichero> Utiliza el fichero en lugar del «buffer» de registro del núcleo.
-f, --facility <lista> Restringe la salida a los recursos definidos.
-H, --human Salida legible para humanos.
-k, --kernel Muestra los mensajes del núcleo.
-L, --color[=<cuándo>] Colorea los mensajes (auto, siempre o nunca).
-l, --level <lista> Restringe la salida a los niveles definidos.
-n, --console-level <nivel> Establece el nivel de los mensajes imprimidos por la consola.
-P, --nopager No redirige la salida a un busca.
-r, --raw Imprime el «buffer» de mensajes en bruto.
-S, --syslog Fuerza a utilizar syslog(2) en lugar de /dev/kmsg.
-s, --buffer-size <tamaño> Tamaño de «buffer» para consultar el «buffer» circular del núcleo.
-u, --userspace Muestra los mensajes del espacio d usuario.
-w, --follow Espera por mensajes nuevos.
-x, --decode Descodifica recurso y nivel en una cadena legible.
-d, --show-delta Muestra la diferencia de tiempos entre los mensajes imprimidos.
-e, --reltime Muestra la hora local y la diferencia de tiempo en formato legible.
-T, --ctime Fechas legibles por humanos.
-t, --notime No imprime la marca de tiempo de los mensajes.
--time-format <formato> Muestra la marca de tiempo con el formato: [delta|reltime|ctime|notime|i
so].
NOTA: Suspender / Reanudar volverá inexactas las marcas de tiempo de ctime e iso.
-h, --help Muestra esta ayuda y sale.
```

```
-V, --version Muestra información de versión y sale.
si no pongo parametros muestra toda la informacion
```

(g) lspci

Muestra la información acerca de las ranuras PCI del sistema. Su sintaxis es `lspci [opciones]`

```
-t mostrar un diagrama que incluye a todas las ranuras PCI, puentes, dispositivos y conexiones e
ntre éstos
-v, -vv o -vvv: para ver tres diferentes niveles de detalle. La salida será muy extensa en todos
los casos.
-n: mostrar los código de dispositivo como números en lugar de mostrar la lista de identidades P
CI
```

(h) at → **at [OPTION...] runtime**

Podemos leer comandos de entrada estándar o scripts que deseamos que se ejecuten más tarde. Es útil para apagar el sistema a una hora en la que no vamos a estar.

Se instala con `sudo apt update` , `sudo apt install at`

```
-f: Para leer comandos de un archivo en lugar de la entrada estándar, siguiendo la ruta del arch
ivo. Por ejemplo, para crear un trabajo que ejecutará el script /home/script.sh --> at 09:00 -f
 /home/script.sh

-M: De forma predeterminada, si el comando produce una salida, enviará un mail con la salida al
 usuario una vez que se complete el trabajo. Invocar at la -M opción para suprimir la notificaci
ón por mail

-m: para enviar un correo electrónico incluso si no hay salida.
```

(i) netstat → **netstat [opciones]**

Supervisa las conexiones de red tanto entrantes como salientes, así como las tablas de enrutamiento, estadísticas de la interfaz, etc.

```
-a: Listar todas la conexiones
-i,-ie: Identificar las interfaces de red
-at: Listar las conexiones TCP
-au: Listar las conexiones UDP
-tnl: Puertos abiertos que escuchan un servicio
-s: Estadísticas de la red
```

(j) mount → **mount [opciones] [-t tipo] [-a] [-o opc] dispo punto_montaje**

Sirve para montar dispositivos. Esto solo lo puede hacer un superusuario

```
-a Permite montar todos los sistemas de ficheros especificados en el fichero /etc/fstab
-f Realiza un montaje ficticio. Sirve para comprobar si el montaje se realizaría correctamente
-n Monta el dispositivo sin escribirlo en el fichero /etc/mtab
```

```
-r Monta el sistema de ficheros como sólo lectura
-w Monta el sistema de ficheros para lectura/escritura (opción por defecto)
```

(k) umount → **umount [opciones]**

Desmonta

```
–V: Permite realizar pruebas

–a: Desmonta varios sistemas de archivos a la vez. Si se incluyen puntos de montaje con la opció
n –a, los sistemas de archivos se desmontan. Si no hay puntos de montaje incluidos, se realiza u
n intento de desmontar todos los sistemas de archivos que aparecen en /etc/mnttab, excepto los s
istemas de archivos "necesarios", como /, /usr, /var, /proc, /dev/fd y /tmp. Como el sistema de
 archivos ya está montado y debe tener una entrada en /etc/mnttab, no tiene que incluir un indic
ador para el tipo de sistema de archivos.

–f: Fuerza un sistema de archivos ocupado para que se desmonte. Puede utilizar esta opción para
 desbloquear un cliente bloqueado cuando intenta montar un sistema de archivos desmontable.
```

(l) head → **head [opciones]**

Se utiliza para mostrar información, si lo usamos sin parámetros, de las 10 primeras líneas de un archivo en la salida estándar. También se puede utilizar para mostrar información de varios ficheros a la vez.

```
-n* --> * numero de lineas que queremos que nos muestre
-c* --> los primeros * bytes
```

(m) losetup → **losetup [-d][-e <cifrado>][-o <número de traducción>]**

Se utiliza para fijar el dispositivo de bucle

```
-d: dispositivo extraíble.
-e <cifrado>: Iniciar cifrado codificación.
-o <número de traducción>: Establecer el número de conversión de datos.
```

(n) write → **write usuario [nombre-tty] mensaje x=numero terminal**

Sirve para enviar un mensaje a un usuario en particular. Para ello primero debemos saber cuales son los usuarios conectados, utilizando el comando “`who`“.

(ñ) mkfs → **mkfs [-V] [-t fstype] [c] nombreParticion**

Formatea una partición y crea el marco necesario para manejar y almacenar metadatos de archivos.
Dispositivo: examen preliminar de la partición del disco duro, por ejemplo: / dev / sda1

> -V: Modo de presentación detallada
> -t: para un determinado tipo de sistema de archivos, por defecto de Linux es ext2
> -c: Al hacer el sistema de archivos, inspeccionar la partición de mala pista

`(o) fdisk (con cuidado)` → `fdisk [opciones]`

Nos permite realizar acciones sobre los discos duros, tales como crear y editar nuevas particiones, eliminar las viejas y modificar el sistema de archivos.

```
-l: Para listar las particiones del disco duro actual. Podemos agregar el nombre del disco para
 mostrar solo las particiones asociadas a el. fdisk + nombreDisco: ingresar al modo de comando p
ara ese disco luego de entrar al modo comando tenemos varias opciones:
d: borrar una particion
m: despliega opciones del menu
n: crea una nueva particion
p: despliega particiones actuales
q: sale de fdisk sin guardar modificaciones
t: cambio tipo de particion seleccionada
v: analiza tabla de particion
w: guarda cambios y sale de fdisk
```

12. Investigue su funcionamiento y parámetros más importantes

Indique en qué directorios se almacenan los comandos mencionados en el ejercicio anterior.

Al ejecutar el comando `sudo find / -name NOMBRECOMANDO` aparece en consola el directorio de cada uno en particular. La mayoría se encuentran o bien en `/bin` o en `/sbin.`