Introducción a los sistemas operativos - Práctica 1

1. Características de GNU/Linux:

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

- De libre distribución: La mayoría de las distribuciones de GNU/Linux son gratuitas, lo que permite a los usuarios acceder a un sistema operativo completo sin costo (los S.O comerciales requieren licencia)
- Código abierto: Su código fuente está disponible para cualquiera que quiera verlo, modificarlo y distribuirlo. Esto nos permite estudiarlo, personalizarlo, auditarlo. Aprovechar la documentación, etc.
- Multiusuario: Puede manejar múltiples usuarios simultáneamente sin degradar el rendimiento del sistema.
- Multitarea: Puede manejar múltiples tareas simultáneamente sin degradar el rendimiento del sistema.
- Personalización: Es altamente personalizable, permitiendo a los usuarios ajustar el entorno gráfico, aplicaciones y el comportamiento del sistema.
- Portabilidad. Puede ejecutarse en una amplia variedad de hardware, desde computadoras personales hasta servidores, dispositivos móviles, y sistemas embebidos. (altamente portable)
- Todo es un archivo: Loa dispositivos son representados con archivos también. (y los directorios lo mismo). Permite que un programa interactúe usando comandos y herramientas que usa para leer y escribir archivos.

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

Windows:

- Código cerrado: en windows los usuarios no tienen scceso al código fuente y deben confiar en Microsoft para las actualizaciones y parches.
- Costo: Windows es un software comercial, por lo que los usuarios deben comprar una licencia para usarlo.
- Multiusuario: Lo mismo que GNU/Linux pero más limitado.
- Multitarea: Lo mismo solo que GNU/Linux maneja mejor sus recursos.
- Portabilidad: Es menos portable.
- Todo es un archivo: Windows no tiene esta característica, los dispositivos y recursos del sistema se manejan con un enfoque más centrado a interfaces gráficas y registros de windows.

Mac:

- Código cerrado: sistema operativo de código cerrado, desarrollado por Apple.
- Costo: está diseñado para funcionar únicamente en hardware de Apple, lo que implica un costo asociado tanto con el hardware como con el software.
- Multitarea y Multiusuario: Maneja la multitarea y el Multiusuario de manera eficiente, similar a GNU/Linux, pero con menos flexibilidad en la personalización.
- Portabilidad: Es menos portable ya que solo funciona en hardware diseñado por Apple.
- Todo es un archivo: Mac sigue esta filosofía, similar a Linux. Los dispositivos pueden acceder y manipular como archivos dentro del sistema.

(c) ¿Qué es GNU?

Es un proyecto de software libre iniciado por Richard Stallman en 1983 con el objetivo de crear un sistema operativo completo enteramente por software libre, similar a unix pero sin incluir su código propietario. GNU es un acrónimo recursivo que significa “GNU’S Not Unix” (GNU no es Unix), resaltando que aunque el sistema operativo es similar a Unix, no contiene código Unix.

(d) Indique una breve historia sobre la evolución del proyecto GNU

El proyecto GNU (GNU's Not Unix) fue iniciado por Richard Stallman en 1983 con el objetivo de crear un sistema operativo completamente libre y compatible con Unix. La idea detrás de GNU era

que los usuarios tuvieran la libertad de usar, estudiar, modificar y compartir software, en contraste con las restricciones impuestas por el software propietario. En los primeros años del proyecto, Stallman y la comunidad de desarrolladores contribuyeron al desarrollo de varias herramientas fundamentales para un sistema operativo, como el compilador GCC, el editor de texto Emacs y el depurador GDB. Sin embargo, el núcleo (kernel) de GNU, llamado Hurd, experimentó retrasos significativos. Para llenar este vacío, en 1991 Linus Torvalds lanzó el kernel Linux, que combinado con las herramientas del proyecto GNU permitió crear un sistema operativo completamente funcional, a menudo llamado "GNU/Linux". Este sistema ha sido la base de la mayoría de las distribuciones de software libre y ha jugado un papel central en el desarrollo del movimiento del software libre. Con el tiempo, el proyecto GNU se convirtió en el pilar del movimiento de software libre, promoviendo licencias como la GPL (Licencia Pública General) para asegurar que los programas sigan siendo libres para todos. Aunque el núcleo Hurd nunca alcanzó la popularidad de Linux, el proyecto GNU sigue siendo un referente en la filosofía y el desarrollo del software libre.

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

La multitarea es la capacidad de un sistema operativo para ejecutar varios procesos al mismo tiempo. Esto se logra distribuyendo los recursos del procesador entre diferentes tareas, lo que permite que varias aplicaciones o programas funcionen simultáneamente sin interferir entre sí. GNU/Linux utiliza la multitarea, permitiendo que múltiples procesos se ejecuten de manera concurrente. Gestiona el uso del procesador mediante el escalonamiento de tareas, lo que garantiza que cada proceso reciba tiempo de ejecución adecuado, optimizando así el rendimiento del sistema.

(f) ¿Qué es POSIX?

POSIX (Portable Operating System Interface) es un conjunto de estándares definidos por IEEE para garantizar la compatibilidad entre sistemas operativos. Su objetivo es facilitar la portabilidad de software entre diferentes plataformas, asegurando que programas escritos para un sistema operativo compatible con POSIX puedan ejecutarse en otros sin cambios significativos. POSIX define interfaces de programación y servicios básicos, como la gestión de archivos, procesos y la interacción con dispositivos. Muchos sistemas operativos, incluidos GNU/Linux, macOS y algunas versiones de Windows, son compatibles con POSIX o implementan partes de este estándar.

2. Distribuciones de GNU/Linux:

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

Una distribución de GNU/Linux es un sistema operativo basado en el núcleo Linux y las herramientas del proyecto GNU, al cual se le añaden otros componentes como software, gestores de paquetes y entornos gráficos para crear un sistema completo y funcional. Cada distribución está adaptada para diferentes usos o preferencias de usuarios, como servidores, desarrollo, o usuarios de escritorio.

Ejemplos de distribuciones y diferencias básicas:

1. Ubuntu:
2. Debian
3. Fedora:
4. Arch Linux

(b) ¿En qué se diferencia una distribución de otra?
Las distribuciones de GNU/Linux se diferencian entre sí en varios aspectos, entre ellos:

1. **Gestor de paquetes**: Cada distribución usa un sistema de gestión de paquetes diferente para instalar, actualizar y gestionar software. Ejemplos incluyen:
    - **Debian/Ubuntu**: APT (Advanced Package Tool)
    - **Fedora**: DNF (anteriormente YUM)
    - **Arch Linux**: Pacman
2. **Enfoque del usuario**: Algunas distribuciones están diseñadas para principiantes, como Ubuntu, mientras que otras están orientadas a usuarios avanzados o desarrolladores, como Arch Linux o Gentoo.
3. **Ciclo de actualizaciones**:
    - **Rolling release**: Las distribuciones como Arch Linux actualizan continuamente los paquetes sin necesidad de una versión nueva del sistema.
    - **Versiones fijas**: Distribuciones como Ubuntu y Fedora lanzan versiones estables periódicamente con un conjunto fijo de características.
4. **Entornos de escritorio**: Aunque se pueden instalar múltiples entornos de escritorio en cualquier distribución, algunas vienen con uno predeterminado. Por ejemplo:
    - **Ubuntu**: GNOME (anteriormente Unity)
    - **Kubuntu**: KDE Plasma
    - **Fedora**: GNOME
5. **Estabilidad vs. innovación**: Algunas distribuciones priorizan la estabilidad, como Debian, mientras que otras prefieren incorporar las últimas tecnologías y características, como Fedora.
6. **Compatibilidad y soporte**: Las distribuciones varían en cuanto a soporte de hardware, facilidad de instalación, documentación y tamaño de la comunidad.

Estas diferencias permiten a los usuarios elegir la distribución que mejor se adapte a sus necesidades y preferencias.

(c) ¿Qué es Debian? Acceda al sitio 1 e indique cuáles son los objetivos del proyecto y una breve cronología del mismo

Debian es una distribución de GNU/Linux creada en agosto de 1993 por Ian Murdock. Su objetivo principal es desarrollar y mantener un sistema operativo basado completamente en software libre. Debian se distingue por su compromiso con la comunidad y su transparencia en los procesos de desarrollo, al estar organizada por voluntarios a nivel mundial. Es la única gran distribución de Linux que no está controlada por una entidad comercial, y sigue un estricto contrato social que prioriza la libertad del software.

Entre los objetivos del proyecto Debian destacan la creación de un sistema operativo robusto, estable y completamente libre, accesible para todo el mundo. Para ello, se desarrolla un sistema de gestión de paquetes que garantiza la instalación segura y eficiente de software.

En cuanto a la cronología, Debian fue patrocinada por el Proyecto GNU de la FSF durante su primer año (1994-1995). Desde entonces, ha crecido enormemente, lanzando versiones estables periódicas y adoptando innovaciones técnicas y organizativas que han establecido un estándar en el mundo del software libre

3. Estructura de GNU/Linux:
(a) Nombre cuales son los 3 componentes fundamentales de GNU/Linux.
Los tres componentes fundamentales de un sistema GNU/Linux son:

- El Kernel (núcleo): El Kernel de Linux se podría definir como el corazón de este sistema operativo. Es, a grandes rasgos, el encargado de que el software y el hardware de una computadora puedan trabajar juntos.
- El Shell (intérprete de comandos): Es el programa que recibe lo que se escribe en la terminal y lo convierte en instrucciones para el sistema operativo.
- El FileSystem (sistema de archivos): Es la forma en que dentro de un SO se organizan y se administran los archivos.

4. Kernel:
(a) ¿Qué es? Indique una breve reseña histórica acerca de la evolución del Kernel de GNU/Linux.
El Kernel es el núcleo de cualquier SO, es software y desde un punto de vista abstracto se encuentra inmediatamente después del hardware.

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 SO de uso didáctico). El 5 de octubre de 1991, se anuncia la primera versión “oficial” de Linux (0.02).

(b) ¿Cuáles son sus funciones principales?
El kernel controla la memoria, los dispositivos periféricos y los procesos de la CPU. Actúa como enlace entre recursos y procesis, asigna la memoria a los procesos y, si algún proceso requiere acceso a algún componente de hardware, el Kernel se lo adjudica. Ademásde impedir que algún proceso realice tareas malicionas sobre la CPU.

(c) ¿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 versión actual del Kernel de Linux es 6.2.16

Antes de la versión 2.6, los números impares indicaban desarrollo, los pares producción.

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.

(d) ¿Es posible tener más de un Kernel de GNU/Linux instalado en la misma máquina?
Puedes tener varios instalados en Linux y eliges uno de ellos cada vez que inicias tu máquina.

(e) ¿Dónde se encuentra ubicado dentro del File System?
El Kernel reside *boot, pero en usr/src/linux se almacena el archivo del código fuente del núcleo de Linux*

(f) ¿El Kernel de GNU/Linux es monolítico? Justifique.
El Kernel de Linux es monolítico, debido a que todas las funciones se encuentran en un solo lugar, conteniendo todo lo necesario para administrar el funcionamiento del sistema.

5. Intérprete de comandos (Shell)):

(a) ¿Qué es?

El shell-también conocido como el intérprete de comandos, línea de comandos, terminal o consola- es un programa que actúa como interfaz para comunicar al usuario con el sistema operativo mediante una ventana que espera comandos textuales ingresados por el usuario en el teclado, los interpreta y los entrega al SO para su ejecución.

(b) ¿Cuáles son sus funciones?

El intérprete de comandos (Shell) tiene las siguientes funciones principales:

1. **Ejecución de comandos**: Interpreta y ejecuta comandos que el usuario ingresa.
2. **Automatización con scripts**: Permite crear scripts para automatizar tareas repetitivas.
3. **Gestión de procesos**: Inicia, detiene y controla procesos del sistema.
4. **Redirección de entradas y salidas**: Permite redirigir la salida o entrada de comandos y combinar comandos.
5. **Expansión de comodines**: Usa caracteres especiales para operar sobre múltiples archivos o directorios.
6. **Variables de entorno**: Maneja configuraciones del sistema.
7. **Control de permisos**: Administra permisos de acceso a archivos y procesos.
8. **Historial de comandos**: Permite acceder a comandos usados previamente.
9. **Alias**: Crea atajos para comandos largos o complejos.
10. **Operadores de control**: Facilita el uso de condicionales y bucles en scripts.

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

1. Korn-Shell(ksh)
2. Bourne-Shell(sh)
3. C-Shell(csh)

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

Los comandos propios son reconocidos y ejecutados por el shell directamente y sin ayuda de ningún otro ejecutable

Los comandos externos estan en */bin*, */usr*bin, */usr*local/bin o cualquier otra ubicación si se la agrega a la variable PATH

(e) ¿Por qué se considera que el Shell no es parte del Kernel de GNU/Linux)

Porque no parte del software creado para administrar los recursos y el hardware de la computadora, sino que es una extensi´pn que permite el diálogo del usuario con el SO y, por ende, el mismo Kernel

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

Si es posible definir un intérprete distinto y se puede definir al crear o modificar usuarios en el sistema, y en el archivo */etc/*passwd

6. Sistema de Archivos (File System):

(a) ¿Qué es?

Es la forma en la que el SO organiza y administra los archivos.

(b) Mencione sistemas de archivos soportados por GNU/Linux
Linux soporta varios sistemas de archivos: ext2, ext3, ReiserFS, XFS, JFS, UFS, ISO9660, FAT, FAT32 o NTFS

(c) ¿Es posible visualizar particiones del tipo FAT y NTFS en GNU/Linux?
Si, es posible visualizar y acceder a particiones de tipo FAT y NTFS en un sistema GNU/Linux
(d) ¿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?
La estructura básica de los sistemas de archivos en GNU/Linux es:

- /: Es el directorio raiz
- /home: Contiene los directorios "home" de los usuarios.
- /var: Información que varıa de tamaño
- /etc: Almacena configuración del sistema
- /bin: Almacena ejecutables o binarios.
- /dev: Almacena controladores de dispositivo
- /usr: Aplicaciones de usuarios
- /sbin: Almacena programas esenciales del sistema
- /lib: Contiene las imágenes de las librerías compartidas.
- /proc: Es un "sistema de ficheros virtual"
- /root: Directorio home de root
- /tmp: Almacena archivos temporales.

FHS significa Filesystem Hierarchy Standard o Jerarquía del Sistema de Archivos Estandar.

7. Particiones
(a) Definición. Tipos de particiones. Ventajas y Desventajas.
Es una forma de dividir lógicamente el disco físico. Los tipos son:
- Partición primaria: División cruda del disco (puede haber 4 por disco). Se almacena información de la misma en el MBR.

- Partición extendida: Sirve para contener unidades lógicas en su interior. Solo puede existir una partición de este tipo por disco. No se define un tipo de FS directamente sobre ella.

-Partición lógica: Ocupa la totalidad o parte de la partición extendida y se le define un tipo de FS. Las particiones de este tipo se conectan como una lista enlazada.

Ventajas:
- Una unidad bien organizada puede ahorrarte tiempo en la gestión.

- Crea copias de seguridad fácilmente.
- Permite aplicar sistema de archivos múltiple y estilo de partición
- Facilita la instalación en múltiples sistemas operativos.
- Colocar los archivos importantes en particiones diferentes puede reducir el riesgo de pérdida y corrupción de archivos.

Desventajas:
- Error de operación de partición al tener demasiadas particiones.
- No es necesario para gente sin experiencia.
- Todas las particiones permanecen en un disco duro. Si tu disco está dañado, desaparecerán todos los datos que contenga.

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

Las particiones en cada disco son representadas añadiendo un número al nombre del disco. En la primera unidad SCSI o SATA: /*dev/*sda1, /*dev*/sda2, … En la primera unidad IDE: /*dev*/hda1, *devhda2,…*

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

Como mínimo es necesaria una sola partición primaria, para la raíz (/).

(e) ¿Qué tipo de software para particionar existe? Menciónelos y compare.

Particionadores destructivos: Permiten crear y eliminar particiones.

Particionadores no destructivos: Permiten crear, eliminar y modificar particiones, generalmente las distribuciones permiten hacerlo desde la interfaz de instalación.

8. Arranque (bootstrap) de un Sistema Operativo:

(a) ¿Qué es el BIOS? ¿Qué tarea realiza?

La BIOS es un software en la placa base que inicializa y prueba el hardware al encender la computadora. Sus principales tareas son:

1. Autoprueba de encendido (POST) para verificar componentes.
2. Cargar el sistema operativo desde el disco.
3. Configurar hardware básico como la secuencia de arranque.
4. Gestionar dispositivos de entrada/salida (teclado, puertos).
5. Ajustar parámetros del sistema, como velocidad de reloj y energía.

Es esencial para que el sistema arranque y funcione correctamente.

(b) ¿Qué es UEFI? ¿Cuál es su función?

La UEFI (Unified Extensible Firmware Interface) es una versión moderna de la BIOS tradicional, diseñada para reemplazarla. Su función principal es la misma: inicializar el hardware y cargar el sistema operativo, pero ofrece más características avanzadas.

Funciones principales de UEFI:

1. Arranque más rápido que la BIOS tradicional.
2. Soporte para discos grandes (más de 2 TB) con el sistema de partición GPT.
3. Interfaz gráfica más amigable y con soporte para ratón.
4. Mayor seguridad mediante características como Secure Boot, que previene el arranque de software no autorizado.

UEFI es más eficiente y flexible que la antigua BIOS.

(c) ¿Qué es el MBR? ¿Que es el MBC?

MBR (Master Boot Record):

Es el registro de arranque maestro, ubicado en el primer sector de un disco duro (sector 0). Contiene:

- La tabla de particiones del disco.
- Un pequeño programa que se encarga de iniciar el proceso de arranque del sistema operativo.

Limitaciones:

- Solo soporta discos de hasta 2 TB.
- Permite un máximo de 4 particiones primarias.

MBC (Master Boot Code):

Es el código de arranque maestro, una parte del MBR. Su tarea es localizar el sector de arranque de la partición activa (que contiene el sistema operativo) y transferirle el control para que se cargue.

En resumen, el MBR es toda la estructura que incluye la tabla de particiones y el MBC, mientras que el MBC es solo el código de arranque dentro del MBR.

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

Las siglas **GPT** se refieren a **GUID Partition Table** (Tabla de Partición con Identificador Único Global). Es un estándar de particionado de discos que sustituye al antiguo **MBR** (Master Boot Record).

GPT reemplaza al MBR, superando sus limitaciones, como el soporte para discos mayores de 2 TB y el límite de 4 particiones primarias.

Formato:

- Usa un Identificador Único Global (GUID) para cada partición.
- Permite hasta 128 particiones en discos con formato GPT.
- Es compatible con discos de gran capacidad (mayores a 2 TB).

GPT es parte del estándar UEFI y es más moderno y robusto que el MBR.

(e) ¿Cuál es la funcionalidad de un “Gestor de Arranque”? ¿Qué tipos existen? ¿Dónde se instalan? Cite gestores de arranque conocidos.

Un **gestor de arranque** es un software que se ejecuta al inicio de la computadora para cargar el sistema operativo. Permite seleccionar entre varios sistemas operativos instalados en el mismo equipo y gestionar el proceso de arranque.

Tipos de Gestores de Arranque:

1. Monoboot: Diseñado para cargar un solo sistema operativo.
2. Multiboot: Permite elegir entre varios sistemas operativos instalados en la misma máquina.

El gestor de arranque se instala generalmente en:

- El MBR (Master Boot Record) en discos con este esquema de particiones.
- La partición EFI en sistemas que usan UEFI.

Gestores de Arranque Conocidos:

- GRUB (GRand Unified Bootloader): Muy utilizado en sistemas Linux.
- LILO (Linux Loader): Otro gestor para Linux, aunque menos usado hoy en día.
- Windows Boot Manager: Utilizado en sistemas Windows.
- Syslinux: Ligero, usado en sistemas Linux.
- rEFInd: Un gestor de arranque gráfico, compatible con UEFI, usado para sistemas duales o múltiples.

Estos gestores permiten iniciar diferentes sistemas operativos o configuraciones de arranque.

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

El proceso de arranque (bootstrap) de una computadora, desde que se enciende hasta que el sistema operativo es cargado, sigue estos pasos:

Encendido del hardware: La energía se distribuye y el procesador empieza a ejecutar instrucciones básicas.

1. **Ejecución de BIOS/UEFI**: Realiza una autoprueba de encendido (POST) y detecta dispositivos.
2. **Selección del dispositivo de arranque**: Busca el dispositivo de almacenamiento para cargar el sistema operativo.
3. **Carga del Gestor de Arranque**: El gestor de arranque (como GRUB o Windows Boot Manager) se carga y prepara el sistema.
4. **Carga del Sistema Operativo**: El gestor de arranque carga el kernel del sistema operativo.
5. **Inicialización del Kernel**: El kernel configura el hardware y monta el sistema de archivos.
6. **Carga de servicios y entorno de usuario**: Inicia los servicios y carga la interfaz de usuario.

El proceso finaliza cuando el sistema operativo está listo para usarse.

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

El proceso de arranque en GNU/Linux se puede resumir en estos pasos:

1. Encendido y POST: La BIOS/UEFI realiza un chequeo del hardware.
2. Cargador de Arranque: GRUB u otro cargador carga el kernel de Linux en la memoria.
3. Cargar el Kernel: El kernel se carga y se inicializa, detectando y configurando el hardware.
4. Inicialización del Kernel: El kernel monta el sistema de archivos raíz.
5. Sistema de Inicialización: `systemd` (o `init`) gestiona el arranque de servicios y procesos.
6. Ejecución de Scripts de Arranque: Se configuran servicios como red y entorno gráfico.
7. Acceso de Usuario: Se presenta el login para que el usuario acceda al sistema.

Este flujo lleva a que el sistema esté listo para su uso.

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

El proceso de parada (shutdown) en GNU/Linux implica estos pasos:

1. Notificación de Apagado: Se avisa a los usuarios y procesos sobre el apagado inminente.
2. Detención de Servicios: `systemd` (o `init`) comienza a detener los servicios y demonios en el orden adecuado, respetando las dependencias.

3. Desmontaje de Sistemas de Archivos: Se cierran los archivos abiertos y se desmontan los sistemas de archivos, garantizando la integridad de los datos.
4. Sincronización de Discos: Se sincronizan los buffers del disco para asegurar que todos los datos en memoria se escriban en el almacenamiento.
5. Parada del Kernel: El kernel detiene todos los procesos restantes y finaliza el acceso al hardware.
6. Apagado del Sistema: El hardware se apaga o reinicia, dependiendo del comando utilizado (`shutdown`, `halt`, `reboot`).

(i) ¿Es posible tener en una PC GNU/Linux y otro Sistema Operativo instalado? Justifique
Sí, es posible tener GNU/Linux y otro sistema operativo instalados en la misma PC mediante un arranque dual (dual boot). Esto se logra particionando el disco duro para que cada sistema operativo tenga su propio espacio, y utilizando un cargador de arranque como GRUB, que permite seleccionar qué sistema operativo iniciar al encender la computadora. Este método es común cuando se necesita usar tanto Linux como otro sistema operativo, como Windows, en el mismo equipo.

9. Archivos:
(a) ¿Cómo se identifican los archivos en GNU/Linux?
En GNU/Linux, los archivos se identifican por:

1. Nombre: Un nombre único dentro de su directorio.
2. Ruta: Puede ser absoluta (desde `/`) o relativa.
3. Inodo: Un identificador que almacena metadatos del archivo.
4. Tipo de archivo: Determinado por el sistema de archivos o inspección del contenido.

Estos elementos permiten la correcta identificación y manejo de archivos en el sistema.
(b) Investigue el funcionamiento de los editores vi y mcedit, y los comandos cat y more.
vi:

- Editor de texto modal (modo comando e inserción).
- Comandos básicos:
    - `i` para insertar, `:wq` para guardar y salir, `:q!` para salir sin guardar.

mcedit:

- Editor más visual y fácil de usar.
- Comandos básicos:
    - `F2` para guardar, `F10` para salir.

cat:

- Muestra, concatena o crea archivos.
    - Ejemplo: `cat archivo.txt` (muestra contenido), `cat archivo1 archivo2 > combinado.txt` (concatena).

more:

- Muestra archivos de manera paginada.
    - Ejemplo: `more archivo.txt`, usa `Espacio` para avanzar y `q` para salir.

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

Resuelto en la máquina virtual. Pasos por si quiero volver a hacerlo:

- cd ~ (para ubicarme en el directorio principal)

- vi prueba.exe (para crear el archivo)

- i, enter (para poder editar)

- ingreso el num de alumno, enter

- ingreso mi nombre y apellido

- esc (para salir de modo edicion)

- :wq (para guardar los cambios)

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

El comando file en GNU/Linux identifica el tipo de archivo basándose en su contenido, no en su extensión. Aquí tienes un resumen de cómo funciona y algunos ejemplos:

- Uso básico:

  bash
  Copiar código
  file nombre_del_archivo

- Ejemplos:
  - Archivo de texto:

    bash
    Copiar código
    file texto.txt

    Salida: texto.txt: ASCII text

  - Archivo binario:

    bash
    Copiar código
    file binario.bin

    Salida: binario.bin: data

  - Archivo de imagen:

    bash
    Copiar código
    file ejemplo.jpg

    Salida: ejemplo.jpg: JPEG image data

  - Archivo ejecutable:

bash
Copiar código
file /bin/ls

Salida: /bin/ls: ELF 64-bit LSB executable, x86-64, ...

- Diferencias:
    - No depende de extensiones: Identifica archivos independientemente de su nombre o extensión.
    - Basado en contenido: Examina el contenido del archivo para determinar su tipo.

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

(a) Cree la carpeta ISO2017

Comando: mkdir ISO2017

Descripción: mkdir (make directory) crea un nuevo directorio

(b) Acceda a la carpeta (cd)

Comando: cd ISO2017

Descripción: cd (change directory) cambia el directorio actual a ISO2017

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

Comando: touch iso2017-1 iso2017-2

Descripción: touch crea archivos vacíos o actualiza la fecha de modificación de los archivos si ya existen

(d) Liste el contenido del directorio actual (ls)

Comando: ls

Descripción: ls lista los archivos y directorios en el directorio actual.

Parámetros importantes:

→ -l para una lista detallada

→ -a para incluir archivos ocultos

(e) Visualizar la ruta donde estoy situado (pwd)

Comando: pwd

Descripción: pwd (print working directory) muestra la ruta completa del directorio actual

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

Comando: find . -name “iso*”

Descripción: find busca archivos y directorios que coincidan con los criterios especificados.

Parámetros importantes:

→ . indica que la búsqueda comienza en el directorio actual

→ -name especifica el patrón de búsqueda.

(g) Informar la cantidad de espacio libre en disco (df)

Comando: df

Descripción: df (disk free) muestra el uso del espacio en disco para los sistemas de archivos montados.

Parámetros importantes:

→ -h muestra el tamaño en formato legible para humanos (GB, MB).

(h) Verifique los usuarios conectado al sistema (who)

Comando: who

Descripción: who muestra información sobre los usuarios actualmente conectados al sistema.

(i) Acceder a el archivo iso2017-1 e ingresar Nombre y Apellido
Comando: nano iso2017-1 ///// vi iso2017-1
Descripción: edita el archivo iso2017-1 usando nano o vi, donde puedes ingresar el texto nombre y apellido
(j) Mostrar en pantalla las últimas líneas de un archivo (tail).
Comando: tail archivo.txt
Desccripción: tail muestra las últimas líneas de un archivo
Parámetros importantes:
→ -n especifica el número de líneas a mostrar, por ejemplo, -n 10 para las últimas 10 líneas

11. Investigue su funcionamiento y parámetros más importantes:
(a) shutdown
Función: Apaga o reinicia el sistema
Comando: shutdown [opciones] [tiempo] [mensaje]
Parámetros importantes:
→ -h apaga el sistema
→ -r reinicia el sistema
→ now para ejecutar inmediatamente.
(b) reboot
Función: Reinicia el sistema
Comando: reboot
(c) halt
Función: Detiene el sistema
Comando: halt
Parámetros importantes: Similar a shutdown h now, detiene el sistema sin apagado completo
(d) locate
Función: encuentra archivos rápidamente usando una base de datos indexada
Comando: locate [opciones] nombre_del_archivo
Parámetros importantes:
→ -i busca sin distinguir entre mayúsculas y minúsculas.
→ -r usa expresiones regulares.
(e) uname
Función: Muestra información sobre el sistema.
Comando: uname [opciones]
Parámetros importantes:
→ -a muestra toda la infomación del sistema.
→ -r muestra la versión del kernel
(f) gmesg
Función: Muestra los mensajes del buffer del kernel
Comando: dmesg
Parámetros importantes:
Generalmente se usa para diagnosticar problemas del sistema.
(g) lspci
Función: Muestra infomación sobre el hardware PCI
Comando: lspci
Parámetros importantes:

→ -v para detalles más detallados

(h) at

Función: Programa tareas Para ejecutar en un momento específico.

Comando: at [opciones] hora

Parámetros importantes:

→ -l lista los trabajos programados.

(i) netstat

Función: Muestra estadísticas de red y conexiones.

Comando: netstat [opciones]

Parámetros importantes:

→ -a muestra todas las conexiones y puertos.

→ -t muestra conexiones TCP.

→ -u muestra conexiones UDP

(j) mount

Función: Monta un sistema de archivos.

Comando: mount [opciones] dispositivo punto_de_montaje

Parámetros importantes:

→ -t especifica el tipo de sistema de archivos.

(k) umount

Función: Desmonta un sistema de archivos.

Comando: umount [opciones] punto_de_montaje

(l) head

Función: Muestra las primeras líneas de un archivo.

Comando: head [opciones] archivo

Parámetros importantes:

→ -n especifica el número de líneas a mostrar.

(m) losetup

Función: Configura dispositivos de bucle (loopback)

Comando: losetup [opciones] dispositivo archivo.

Parámetros importantes:

→ -f encuentra el primer dispositivo disponible.

(n) write

Función: Envía un mensaje a otro usuario en el sistema.

Comando: write usuario

Parámetros importantes: Después de ejecutar, escribe el mensaje y finaliza con ctrl+D

(ñ) mkfs

Función: Crea un sistema de archivos en una partición.

Comando: mkfs [opciones] dispositivo

Patámetros importantes:

→ -t especifica el tipo de sistema de archivos (por ejemplo, ext4, vfat)

(o) fdisk (con cuidado)

Función: Manipula las particiones del disco.

Comando: fdisk [opciones] dispositivo

Parámetros importantes:

→ -l lista las particiones del dispositivo.

→ Precaución: cambiar las particiones puede llevar a la pérdida de datos.

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

- **shutdown**: `/sbin/shutdown`
- **reboot**: `/sbin/reboot`
- **halt**: `/sbin/halt`
- **locate**: `/usr/bin/locate`
- **uname**: `/bin/uname`
- **dmesg**: `/bin/dmesg`
- **lspci**: `/usr/bin/lspci`
- **at**: `/usr/bin/at`
- **netstat**: `/bin/netstat` (en sistemas más recientes, reemplazado por `ss` en `/bin/ss`)
- **mount**: `/bin/mount`
- **umount**: `/bin/umount`
- **head**: `/usr/bin/head`
- **losetup**: `/sbin/losetup`
- **write**: `/usr/bin/write`
- **mkfs**: `/sbin/mkfs`
- **fdisk**: `/sbin/fdisk`