

- 1) a) Linux/GNU es un Sistema Operativo tipo Unix (es una familia de sistemas operativos), pero libre, la cual está diseñada por miles de programadores. Corresponde a un S.O gratuito, de libre distribución (existen diversas distribuciones (customizaciones) y representa ser de código abierto, posee la licencia GPL en la cual establece los términos y condiciones bajo los cuales se puede distribuir y modificar el software. Asegura que se pueda acceder al código fuente, estudiar cómo funciona, realizar modificaciones y compartirlas con otros y también protege la apropiación indebida del software de código abierto y asegura que el software libre permanezca libre para siempre.

En cuanto a sus características específicas es multiusuario, la cual quiere decir que este sistema tiene la capacidad de admitir y gestionar múltiples usuarios que pueden interactuar con el sistema de manera concurrente cada uno con su propia cuenta y su propio entorno de trabajo personalizado. Aunque los usuarios comparten el mismo hardware, el sistema operativo se encarga de asegurar que los datos y las actividades de un usuario estén aislados de los otros usuarios, brindando privacidad y seguridad. Linux logra esto a partir de un sistema de permisos y protecciones. Cada archivo y recurso del sistema tiene permisos asociados que indican quién puede acceder, leer, escribir o ejecutar esos elementos. Por otro lado este S.O es multitarea y multiprocesador, es altamente portable, es decir que el mismo código fuente lo puedes compilar en cualquier plataforma, posee diversos intérpretes de comandos (los intérpretes de comandos brindan a los usuarios la capacidad de interactuar con el sistema operativo de manera poderosa y eficiente mediante la entrada de comandos en una línea de comandos, lo que es especialmente útil para tareas de administración y automatización, (es el shell básicamente), un ejemplo de un intérprete en linux es el bash), de los cuales algunos son programables. Además, permite el manejo de usuarios y permisos (Cada usuario puede acceder solo a los recursos para los que tiene permisos específicos, y estos permisos se gestionan cuidadosamente para evitar problemas de seguridad y mal uso). Todo es un archivo (hasta los dispositivos y directorios), cada directorio puede estar en una partición diferente, es case sensitive (sensible a las mayúsculas y minúsculas) y es de código abierto (mencionado anteriormente).

b) Otro sistema operativo puede ser windows, la cual en relación con Linux es un S.O de multiusuario, multitarea y multiprocesador. Sin embargo, este no posee gran variedad de intérpretes de usuario (solamente cmd y powershell), no es de código abierto, no todo se considera un archivo como en linux, el sistema de archivo no abarca todos los componentes del sistema como lo hace en linux, por ejemplo los dispositivos de hardware en windows, por lo general, no se tratan como archivos directos. Por último, windows no es case sensitive y en cuanto a que cada directorio puede estar en una partición diferente esto se cumple para ambos S.O, pero cada uno lo hace de manera diferente.

c) d)

GNU es un acrónimo recursivo que significa "GNU's Not Unix". Fue creado como parte del proyecto GNU, un esfuerzo de software libre iniciado por Richard Stallman en 1983 con el objetivo de desarrollar un sistema operativo completo y libre similar a Unix.

La idea detrás del nombre "GNU's Not Unix" es que el proyecto GNU buscaba crear un sistema operativo similar a Unix pero que no tuviera las restricciones de propiedad y licencia que Unix solía tener en ese momento. En otras palabras, GNU sería compatible con Unix pero no estaría limitado por las mismas restricciones legales.

El proyecto GNU desarrolló muchas herramientas y componentes del sistema operativo, incluyendo el famoso editor de texto Emacs, el compilador GCC (GNU Compiler Collection) y el núcleo GNU Hurd (aunque este último no tuvo tanto éxito como el núcleo de Linux, que finalmente se combinó con las herramientas GNU para crear sistemas operativos funcionales).

En resumen, GNU es un acrónimo que representa los esfuerzos y los principios del proyecto GNU, que tiene como objetivo crear un sistema operativo completo y libre similar a Unix pero sin las restricciones de propiedad intelectual asociadas a Unix.

GNU se refiere a 4 libertades principales de los usuarios del software:

- ^ Libertad de usar el programa con cualquier propósito
- ^ Libertad de estudiar su funcionamiento
- ^ Libertad para distribuir sus copias
- ^ Libertad para mejorar los programas

e) La multitarea es una característica que cumple Linux/GNU, esta se basa en la capacidad de un sistema operativo para ejecutar múltiples tareas (programas, procesos, hilos) de manera aparentemente simultánea, compartiendo el tiempo de la CPU de manera eficiente.

f)

POSIX es el acrónimo de "Portable Operating System Interface" (Interfaz Portátil del Sistema Operativo). Es un conjunto de estándares desarrollados para asegurar la compatibilidad y portabilidad entre sistemas operativos Unix-like, incluyendo Linux y el sistema operativo GNU.

Los estándares POSIX son definidos por el IEEE (Institute of Electrical and Electronics Engineers) y están destinados a estandarizar las interfaces y funcionalidades del sistema operativo para que las aplicaciones puedan ser desarrolladas de manera portátil, lo que significa que pueden ser compiladas y ejecutadas en diferentes sistemas Unix-like sin mayores modificaciones.

- 2) Una distribución de GNU/Linux es un sistema operativo completo que combina el núcleo (kernel) con herramientas y aplicaciones del proyecto GNU, además de otros componentes, para proporcionar un entorno de trabajo funcional y listo para usar.

4 distribuciones:

Ubuntu:

- Enfoque: Ubuntu es conocida por su enfoque en la facilidad de uso y la accesibilidad para los usuarios nuevos en GNU/Linux.
- Entornos de Escritorio: Utiliza principalmente el entorno de escritorio GNOME, pero también ofrece variantes con otros entornos como KDE y Xfce.
- Lanzamientos: Tiene versiones de soporte a largo plazo (LTS) que se centran en la estabilidad y actualizaciones regulares de versiones normales.
- Audiencia: Adecuada tanto para usuarios principiantes como para usuarios avanzados.

Fedora:

- Enfoque: Fedora se enfoca en la innovación y la adopción temprana de nuevas tecnologías en el mundo de GNU/Linux.

- Entornos de Escritorio: Utiliza principalmente el entorno de escritorio GNOME, pero también ofrece opciones para otros entornos.
- Lanzamientos: Tiene una versión principal que se actualiza con nuevas características aproximadamente cada 6 meses.
- Audiencia: Atrae a usuarios que desean estar a la vanguardia de las tecnologías de código abierto.

Debian:

- Enfoque: Debian se enfoca en la estabilidad, la seguridad y la filosofía del software libre.
- Entornos de Escritorio: Ofrece múltiples entornos de escritorio, como GNOME, KDE, Xfce, entre otros.
- Lanzamientos: Sus versiones "estable" se caracterizan por su alta estabilidad y ciclos de lanzamiento más largos.
- Audiencia: Apropia tanto para usuarios que valoran la estabilidad como para aquellos que desean opciones de software completamente libres.

Arch Linux:

- Enfoque: Arch Linux se centra en la simplicidad, la personalización y el principio KISS ("Keep It Simple, Stupid").
- Entornos de Escritorio: No impone un entorno de escritorio específico, permitiendo a los usuarios elegir y personalizar su propio entorno.
- Modelo Rolling Release: En lugar de versiones separadas, sigue un modelo de lanzamiento continuo donde las actualizaciones son constantes.
- Audiencia: Adecuada para usuarios que desean un control total sobre su sistema y están dispuestos a aprender y configurar manualmente.

3)a) Los tres componentes fundamentales son el Kernel (núcleo del sistema operativo) el shell (intérpretes de comandos) y los sistemas de archivo.

b)

Kernel: Es la parte fundamental del sistema operativo (ES SOFTWARE), y se encarga de administrar los recursos de hardware, como la memoria, la CPU, la ejecución de procesos, la E/S y la comunicación y concurrencia. Es el encargado de que el software y el hardware puedan trabajar en conjunto. Representa el código que administra el hardware.

Se dice que es un núcleo monolítico híbrido, ya que todo el sistema operativo corre en modo kernel, pero se tiene la posibilidad de cargar y descargar drivers y otras porciones de kernel mientras el sistema sigue funcionando.

Shell: el shell o intérprete de comandos es un programa que actúa como interfaz, con el objetivo de comunicar al usuario con el sistema operativo mediante una ventana que acepta comandos de texto, los interpreta y se los entrega al SO para su ejecución. La respuesta del SO es mostrada en la misma ventana.

FileSystem: el FileSystem (sistema de archivos) es la estructura en la cual se organizan y se administran los archivos (organiza la forma en que se almacenan los archivos en dispositivos de almacenamiento). El sistema de archivos está conformado por directorios, que es una forma de agrupar archivos, y estos directorios son a su vez archivos y se pueden tratar como tales. Los directorios pueden tener subdirectorios, de manera que el sistema de archivos que se forma es

jerárquico. En linux existen varios tipos de sistemas de archivos, cada uno con sus ventajas y desventajas, por ejemplo: fat, ntfs, ext2, ext3, reiser, etc).

4)a) El kernel en líneas generales representa el código que administra el hardware, es el encargado de que el software y el hardware puedan trabajar juntos.

b) Se encarga de administrar los recursos de hardware, como la memoria, la CPU, la ejecución de procesos, la E/S y la comunicación y concurrencia.

c) La versión actual es la Linux Kernel 6.2.16. El esquema de versionado se realizaba de la siguiente manera:

^ A: Denota versión. Cambia con menor frecuencia. En 1994 (versión 1.0) y en 1996 (versión 2.0)

^ B: Denota mayor revisión. Antes de la versión 2.6, los números impares indican desarrollo, los pares producción

^ C: Denota menor revisión. Solo cambia cuando hay nuevos drivers o características

^ D: Cambia cuando se corrige un grave error sin agregar nueva funcionalidad ← Casi no se usa en las ramas 3.x y 4.x, viéndose reflejado en C.

Falta ultima pregunta.

D) Sí, es posible tener más de un kernel de GNU/Linux instalado en la misma máquina. De hecho, es una práctica común y recomendada en muchos casos. Tener múltiples kernels instalados puede ser beneficioso por varias razones:

Respaldo y Recuperación: Si un nuevo kernel causa problemas de incompatibilidad o funcionamiento incorrecto en tu sistema, tener un kernel anterior instalado te permite volver a un estado conocido y funcional.

Pruebas y Desarrollo: Si eres desarrollador de software o estás probando nuevas características, tener varios kernels te permite probar tu software en diferentes entornos y asegurarte de que sea compatible.

Compatibilidad de Hardware: Algunos kernels pueden tener mejores controladores o soporte para cierto hardware. Puedes tener diferentes kernels instalados para garantizar que tu hardware funcione correctamente.

Experimentación: Puedes experimentar con diferentes versiones de kernel para ver si obtienes mejoras de rendimiento o características que te interesen.

Aprendizaje: Tener múltiples kernels instalados te permite explorar las diferencias entre versiones y comprender mejor cómo funcionan.

También puedes tener la posibilidad de elegir cuál vas a bootear si tienes un gestor de arranque GRUB por ejemplo.

Sin embargo, aunque tener varios kernels puede ser útil, también ocupan espacio en el disco. Por lo tanto, es una buena práctica eliminar kernels antiguos y no utilizados una vez que estés seguro de que no los necesitas más.

e) El núcleo (kernel) de Linux generalmente se encuentra en el directorio /boot en el sistema de archivos. En este directorio, se almacenan los archivos relacionados con el proceso de arranque del sistema, incluyendo el propio núcleo.

El archivo del núcleo en sí se llama típicamente `vmlinuz` (o `vmlinuz-{versión}` para especificar la versión) en muchas distribuciones de Linux. El nombre `vmlinuz` es una abreviatura de "vmlinux compressed", ya que el núcleo se comprime para ocupar menos espacio en el disco.

La ruta completa al núcleo suele ser algo como `/boot/vmlinuz-{versión}`. Junto al archivo del núcleo, en el directorio `/boot`

f) Si, es monolítico híbrido, que quiere decir que todo el sistema operativo corre en modo kernel, pero se tiene la posibilidad de cargar y descargar drivers y otras porciones de kernel mientras el sistema sigue funcionando.

5) a) En líneas generales, el shell es una herramienta que permite la interacción entre el S.O y el usuario mediante una ventana de comandos, donde el usuario da instrucciones que son interpretadas y dadas al S.O y este da respuesta a esos mensajes a través de la misma ventana de comandos. (lo puse con mis palabras).

b) Las funciones que permite este intérprete son:

- . Ejecutar programas a partir del ingreso de comandos.
- . Traduce las entradas del usuario a señales que el sistema puede entender y manejar.

c)

Bash (Bourne Again Shell):

- Popularidad: Bash es el intérprete de comandos más común y ampliamente utilizado en sistemas Linux.
- Personalización: Bash es altamente personalizable a través de la configuración de archivos como `.bashrc` y `.bash_profile`.
- Sintaxis: Tiene una sintaxis similar a la del shell Bourne original, pero con muchas mejoras y características adicionales.
- Compatibilidad: Su sintaxis y comandos son muy compatibles con el Bourne Shell original, lo que facilita la migración desde otros sistemas Unix.
- Historial de Comandos: Ofrece funciones avanzadas para buscar y repetir comandos anteriores utilizando teclas de flecha y atajos.
- Extensiones y Funciones: Admite funciones, alias y variables, lo que permite una amplia personalización y automatización de tareas.

Zsh (Z Shell):

- Características Avanzadas: Zsh ofrece una amplia gama de características avanzadas, como autocompletado potente, corrección de errores interactiva y plugins.
- Autocompletado: El autocompletado de Zsh es extremadamente potente y puede completar nombres de archivos, comandos y más con gran precisión.
- Tema y Estilos: Zsh permite personalizar la apariencia del prompt y la visualización de información útil en el shell.
- Plugins: Ofrece un sistema de plugins que permite agregar funcionalidades adicionales y ampliar sus capacidades.
- Configuración: La configuración de Zsh es más extensa y puede requerir una curva de aprendizaje para aprovechar al máximo todas sus funciones.

Fish (Friendly Interactive Shell):

- Sintaxis Simple: Fish se destaca por su sintaxis más sencilla y amigable para usuarios nuevos. Ofrece autocompletado intuitivo y resaltado de sintaxis.
- Autocompletado Avanzado: El autocompletado en Fish es poderoso y proporciona sugerencias contextuales y detalles útiles mientras escribes.
- Historial de Comandos: El historial de comandos se muestra de forma interactiva y amigable, lo que facilita la búsqueda y la reutilización de comandos anteriores.
- No Requiere Configuración Compleja: Fish busca simplificar la experiencia del usuario al máximo, evitando la necesidad de configuraciones complejas.
- Menos Personalización: Aunque Fish es menos personalizable en comparación con Bash o Zsh, su objetivo es proporcionar una experiencia amigable de inmediato

d) En Linux/GNU, los comandos propios del sistema y los comandos externos están ubicados en varias rutas o directorios que forman parte de la variable de entorno PATH. La **variable PATH** contiene una lista de directorios en los cuales el sistema buscará cuando intentes ejecutar un comando desde el intérprete de comandos. **O en • /usr/bin/ls ?**

e)

El shell y el kernel son dos componentes distintos en un sistema operativo, y esta separación se basa en la arquitectura y el diseño de cómo funciona un sistema operativo moderno.

Funciones y Propósitos Diferentes: El kernel es el núcleo del sistema operativo y se encarga de gestionar los recursos de hardware, como la administración de procesos, la gestión de memoria, la comunicación con dispositivos y otros aspectos fundamentales. Por otro lado, el shell es una interfaz que permite a los usuarios interactuar con el sistema operativo al proporcionar una forma de ingresar comandos y recibir resultados.

Capas de Abstracción: El kernel opera en un nivel más bajo y está más cerca del hardware físico, mientras que el shell es una capa de abstracción que se encuentra por encima del kernel y facilita la comunicación entre los usuarios y el sistema operativo.

Flexibilidad y Elección: Linux es conocido por su capacidad de elección y flexibilidad. Puedes tener diferentes shells en un sistema, como Bash, Zsh, Fish, entre otros. Cambiar el shell no afecta la funcionalidad del kernel ni su capacidad para interactuar con el hardware.

Independencia del Usuario: El shell es una herramienta que el usuario puede seleccionar y personalizar según sus preferencias. Puedes usar diferentes shells según tus necesidades y nivel de experiencia sin afectar directamente el funcionamiento interno del kernel.

Actualizaciones y Mantenimiento: Tanto el kernel como el shell se actualizan de manera independiente. Actualizar el shell no requerirá una actualización del kernel y viceversa.

En resumen, la separación entre el kernel y el shell es una característica fundamental del diseño de los sistemas operativos modernos. El shell actúa como una interfaz

para que los usuarios interactúen con el sistema operativo y ejecute comandos, mientras que el kernel se encarga de las tareas fundamentales de administración de recursos y comunicación con el hardware.

f) **consultar**

Sí, en Linux es posible definir un intérprete de comandos distinto para cada usuario. Esto se hace mediante la configuración del archivo de inicio de sesión del usuario, que puede ser modificado por el propio usuario o por un administrador del sistema. El archivo de inicio de sesión suele ser `.bashrc` para Bash (el intérprete de comandos por defecto), aunque otros shells también pueden utilizar diferentes archivos.

Cada vez que un usuario inicia sesión, el shell cargará la configuración contenida en su archivo de inicio de sesión. Aquí se encuentra el punto donde se puede personalizar el intérprete de comandos. Por ejemplo, si deseas que un usuario específico use Zsh en lugar de Bash, puedes configurar su archivo `.bashrc` (o el archivo correspondiente si estás usando otro shell) para que inicie Zsh en su lugar.

La ruta y el nombre del archivo de inicio de sesión pueden variar según el shell y la configuración del sistema. Algunos ejemplos de los archivos de inicio de sesión son:

- **.bashrc para Bash**
- **.zshrc para Zsh**
- **.fishrc para Fish**

Por ejemplo, para configurar un intérprete de comandos distinto para un usuario en Bash, puedes editar su archivo `.bashrc` y agregar un comando como:

exec zsh

Esto cambiaría el intérprete de comandos a Zsh cada vez que el usuario inicie sesión.

Es importante mencionar que no todos los usuarios (CONSULTARLO) pueden realizar esta tarea por defecto. Cambiar el intérprete de comandos puede afectar la experiencia del usuario y, en algunos casos, incluso la seguridad del sistema. Por lo tanto, generalmente solo los usuarios con permisos administrativos (como el superusuario o aquellos con acceso a la configuración del sistema) pueden realizar estos cambios para otros usuarios

6) a)b) El FileSystem (sistema de archivos) es la estructura en la cual se organizan y se administran los archivos (organiza la forma en que se almacenan los archivos en dispositivos de almacenamiento). El sistema de archivos está conformado por directorios, que es una forma de agrupar archivos, y estos directorios son a su vez archivos y se pueden tratar como tales. Los directorios pueden tener subdirectorios, de manera que el sistema de archivos que se forma es jerárquico. En linux existen varios tipos de sistemas de archivos, cada uno con sus ventajas y desventajas, por ejemplo: fat, ntfs, ext2, ext3, reiser, etc).

c) Sí, es posible visualizar y acceder a particiones formateadas en los sistemas de archivos FAT y NTFS en GNU/Linux. El sistema operativo GNU/Linux ofrece soporte nativo para estos sistemas de archivos, lo que te permite leer y escribir en particiones formateadas en FAT y NTFS.

d) Linux/GNU sigue una jerarquía estándar definida por el sistema de normas de la FHS (Filesystem Hierarchy Standard). Esta jerarquía organiza los archivos y directorios de manera lógica y coherente en todo el sistema, lo que facilita la administración y el acceso a los recursos del sistema.

Directorios más importantes según FHS (Filesystem Hierarchy Standard)

- ^ / Tope de la estructura de directorios. Es como el C:\
- ^ /home Se almacenan archivos de usuarios (Mis documentos)
- ^ /var Información que varía de tamaño (logs, BD, spools)
- ^ /etc Archivos de configuración
- ^ /bin Archivos binarios y ejecutables
- ^ /dev Enlace a dispositivos
- ^ /usr Aplicaciones de usuarios

7)

- a) Una partición es una división lógica de un disco duro o dispositivo de almacenamiento en secciones separadas. Cada partición actúa como una entidad independiente y puede tener su propio sistema de archivos, sistema operativo y datos. Las particiones se utilizan para organizar y administrar el espacio de almacenamiento de manera eficiente. (del texto= Es una forma de dividir lógicamente el disco físico, El disco se divide en particiones).

Tipos de particiones:

Cada disco basado en MBR podía tener (antes) 4 particiones primarias como max. Dsp se permitió tener 3 primarias y una extendida (que contiene más particiones primarias) (tiene su propia tabla de particiones) y dentro de las particiones extendidas estas sirven para poder crear particiones lógicas. La lógica de esto era poder tener más particiones en vez de solo 4. Solo puede haber 1 partición extendida por disco. Estas eran restricciones para el arranque basado en MBR (era un estándar viejo de manejo de discos).

Ventajas de las Particiones:

Organización y Administración: Las particiones permiten organizar y administrar los datos de manera más efectiva al separar diferentes tipos de contenido. Esto facilita la búsqueda y el acceso a archivos específicos.

Mejora del Rendimiento: Particionar un disco puede mejorar el rendimiento al distribuir la carga de trabajo en varias particiones. Por ejemplo, tener una partición separada para el sistema operativo y otra para datos puede mejorar la velocidad de acceso.

Seguridad y Aislamiento: Al tener particiones separadas, es posible aislar y proteger datos importantes. Si una partición se corrompe o infecta con malware, las otras particiones pueden permanecer intactas.

Recuperación de Datos: Si una partición se daña o se corrompe, los datos en otras particiones pueden seguir siendo accesibles y recuperables.

Uso de Sistemas Múltiples: Al tener particiones separadas, es posible tener sistemas operativos diferentes (por ejemplo, Windows y Linux) en el mismo disco.

Desventajas de las Particiones:

Uso Ineficiente del Espacio: Si se asignan tamaños incorrectos a las particiones, es posible desperdiciar espacio en disco o quedarse sin espacio en una partición mientras otras están subutilizadas.

Complejidad: Demasiadas particiones pueden aumentar la complejidad de la administración y la gestión del almacenamiento.

Falta de Flexibilidad: Una vez que se asigna espacio a una partición, cambiar su tamaño puede ser complicado y puede requerir la reorganización de datos.

Mayor Riesgo de Pérdida de Datos: Si una partición se daña, todos los datos en esa partición podrían perderse. Si no se tiene una copia de seguridad adecuada, esto puede ser problemático.

Limitaciones de Número de Particiones: En sistemas con tablas de particiones MBR, existe un límite en el número de particiones primarias y extendidas que se pueden crear. Las tablas GPT (más modernas) resuelven esto en parte, pero todavía hay límites.

Requisitos de Mantenimiento: Mantener varias particiones puede requerir más atención y planificación para asegurarse de que cada partición tenga suficiente espacio y esté respaldada adecuadamente.

b)

Discos IDE (PATA):

- El primer disco IDE (generalmente el disco maestro en el primer controlador IDE) se etiqueta como `/dev/hda`.
- El segundo disco IDE se etiqueta como `/dev/hdb`, y así sucesivamente.

Luego, las particiones en cada disco IDE se numeran:

- Partición primaria 1: `/dev/hda1`
- Partición primaria 2: `/dev/hda2`
- Partición extendida: `/dev/hda4` (si existe)
- Lógica dentro de la partición extendida: `/dev/hda5`, `/dev/hda6`, etc.

Discos SCSI y SATA:

- Los discos SCSI y SATA se etiquetan como `/dev/sd[a-z]`, donde la letra a-z varía según el número de dispositivos conectados.

Las particiones en discos SCSI y SATA se numeran de manera similar a los discos IDE:

- Partición primaria 1: `/dev/sda1`
- Partición primaria 2: `/dev/sda2`
- Partición extendida: `/dev/sda4` (si existe)
- Lógica dentro de la partición extendida: `/dev/sda5`, `/dev/sda6`, etc.

Es importante mencionar que con la introducción de UEFI y la tabla de particiones GPT, la nomenclatura puede ser diferente. En sistemas con GPT, los discos se identifican como `/dev/sd[a-z]`, y las particiones se numeran de manera consecutiva sin tener en cuenta si son primarias, extendidas o lógicas.

Para una nomenclatura más consistente y moderna, especialmente en sistemas con GPT, se recomienda usar las rutas completas a las particiones, como `/dev/disk/by-id/` o `/dev/disk/by-uuid/`, que utilizan identificadores únicos en lugar de nombres genéricos basados en la ubicación física del dispositivo. Estos identificadores son más estables y menos propensos a cambiar si se agregan o retiran dispositivos.

c)

Como mínimo es necesario una partición (para el /) (EN CHAT GPT ME DICE Q ES EL / Y EL SWAP, PUEDE SER?, OSEA MINIMO 2).

^ Es recomendable crear al menos 2 (/ y SWAP)

^ Para crearlas, se utiliza un software denominado **particionador**

. **Existen 2 tipos:** ^ **Destructivos:** permiten crear y eliminar particiones (fdisk)

^ **No destructivo:** permiten crear, eliminar y modificar particiones.

Es importante destacar que aunque estas son las particiones mínimas necesarias para instalar GNU/Linux, es posible que desees crear particiones adicionales para mejorar la organización y el rendimiento. Por ejemplo, puedes optar por una partición /home separada para almacenar tus archivos personales, o una partición /boot separada para el arranque.

CHAT GPT:

Para instalar GNU/Linux, generalmente se pueden utilizar un mínimo de dos particiones: una partición de sistema raíz (root) y una partición de intercambio (swap). A continuación, te proporciono información sobre estas dos particiones:

Partición de Sistema Raíz (root):

- Tipo de Partición: Primaria o Lógica
- Identificación: /dev/sdXN (donde X es la letra del disco y N es el número de partición)
- Tipo de Sistema de Archivos: Ext4 (puede variar, pero Ext4 es común)
- Punto de Montaje: /
- Descripción: Esta partición contiene el sistema operativo y la mayoría de los programas. Es el directorio raíz del sistema de archivos.

Partición de Intercambio (swap):

- Tipo de Partición: Primaria o Lógica
- Identificación: /dev/sdXN (igual que en la partición de sistema raíz)
- Tipo de Sistema de Archivos: swap
- Punto de Montaje: [Sin punto de montaje, es una partición de intercambio]
- Descripción: Esta partición se utiliza como memoria virtual adicional cuando la memoria RAM se agota. Ayuda a mejorar el rendimiento del sistema.

Es importante destacar que aunque estas son las particiones mínimas necesarias para instalar GNU/Linux, es posible que desees crear particiones adicionales para mejorar la organización y el rendimiento. Por ejemplo, puedes optar por una partición /home separada para almacenar tus archivos personales, o una partición /boot separada para el arranque. También puedes considerar otras particiones según tus necesidades y preferencias.

d) Por ejemplo, puedes optar por una partición /home separada para almacenar tus archivos personales, o una partición /boot separada para el arranque.

e) **aca no va lo del software particionador?**

Existen varios programas de software que permiten particionar discos y administrar el almacenamiento en sistemas operativos, incluyendo GNU/Linux. Aquí hay algunos ejemplos populares:

GParted: GParted es una herramienta gráfica ampliamente utilizada para particionar y administrar discos en GNU/Linux. Ofrece una interfaz fácil de usar que permite crear, eliminar, redimensionar y mover particiones.

Parted: Parted es la herramienta de línea de comandos en la que se basa GParted. Ofrece funciones avanzadas de particionamiento, como soporte para GPT y MBR, y es especialmente útil para usuarios más técnicos.

KDE Partition Manager: Esta herramienta es parte del entorno de escritorio KDE y ofrece funciones de particionamiento y administración de discos. Es ideal para usuarios que utilizan KDE Plasma como su entorno de escritorio.

fdisk y parted: Son utilidades de línea de comandos que permiten particionar y administrar discos desde la terminal. Son poderosas, pero requieren un conocimiento básico del particionamiento.

cfdisk: Otra utilidad de línea de comandos que proporciona una interfaz más simple para particionar discos. Es adecuada para usuarios que buscan una herramienta más básica.

TestDisk: Aunque su enfoque principal es la recuperación de datos, TestDisk también ofrece funciones de particionamiento y puede ser útil en situaciones de emergencia.

Disks (gnome-disk-utility): Esta es una aplicación gráfica que se encuentra comúnmente en entornos GNOME. Permite crear y administrar particiones de manera intuitiva.

cfdisk: Una herramienta de línea de comandos más básica que proporciona una interfaz de usuario simplificada para particionar discos.

SystemRescueCD: Esta es una distribución en vivo basada en Linux que incluye herramientas de rescate y particionamiento. Puedes usarla para trabajar en particiones sin iniciar el sistema operativo.

EaseUS Partition Master: Aunque es más conocido en el ámbito de Windows, EaseUS Partition Master también tiene una versión para Linux que permite particionar discos desde una interfaz gráfica.

8)a) CONSULTAR, PQ ES MEDIO DUDOSO POR COMO LO PONE Y EXPLICA EN LA DIAPOSITIVA DE EXPLICACION 1 PAG 45.

La bios en el bootstrap del S.O es el encargado de iniciar la carga del SO a través del MBC. Este está grabado en un chip (ROM,NVRAM). Carga el programa de booteo (desde el MBR). El gestor de arranque lanzado desde el MBC carga el Kernel (prueba y hace disponibles los dispositivos, luego pasa el control al proceso init). La última acción del BIOS es leer el MBC del MBR.

chat gpt:

El BIOS (Basic Input/Output System) es un conjunto de instrucciones y software de bajo nivel que se encuentra en un chip de memoria flash en la placa base (motherboard) de una computadora. Su función principal es realizar una serie de tareas esenciales durante el proceso de arranque o inicio de la computadora. Estas tareas incluyen:

1. Inicialización del hardware: El BIOS realiza pruebas y configuraciones iniciales de los componentes de hardware de la computadora, como la CPU (unidad central de

procesamiento), la memoria RAM, los dispositivos de almacenamiento, las tarjetas gráficas y otros dispositivos periféricos. Esto garantiza que todos los componentes estén funcionando correctamente antes de que el sistema operativo se cargue.

2. Búsqueda y carga del sistema operativo: Una de las funciones más importantes del BIOS es buscar y cargar el sistema operativo desde el dispositivo de almacenamiento designado, como un disco duro, una unidad de estado sólido (SSD), un CD/DVD o una unidad USB. El BIOS tiene una lista de dispositivos de arranque configurados en un orden específico, conocido como "orden de arranque", y busca en estos dispositivos hasta que encuentra uno con un sistema operativo válido para cargar.

3. Interacción con el usuario: El BIOS proporciona una interfaz de configuración básica a la que se puede acceder durante el proceso de arranque presionando una tecla específica (como la tecla Suprimir o F2) antes de que se inicie el sistema operativo. A través de esta interfaz, los usuarios pueden realizar ajustes en la configuración del hardware, como la fecha y la hora, la secuencia de arranque, la contraseña del BIOS y otros parámetros.

4. Mantenimiento y actualización: En algunos casos, el BIOS puede necesitar actualizaciones para admitir nuevos hardware o solucionar problemas de seguridad. Los fabricantes de placas base suelen proporcionar actualizaciones de BIOS que los usuarios pueden instalar para mejorar la compatibilidad y la seguridad de sus sistemas.

En resumen, el BIOS es un componente esencial en cualquier computadora, ya que se encarga de iniciar y preparar el hardware de la máquina para cargar el sistema operativo y permitir que los usuarios realicen ajustes de configuración básicos. Sin el BIOS, una computadora no podría arrancar y funcionar correctamente.

Está bien por como lo explica el chat gpt? :

La BIOS (Sistema Básico de Entrada/Salida, por sus siglas en inglés) juega un papel fundamental en el proceso de arranque de un sistema operativo. La BIOS es un firmware integrado en la placa base de una computadora y se encarga de varias tareas esenciales durante el proceso de inicio:

Autoevaluación del Hardware (POST): La BIOS realiza una prueba de autodiagnóstico del hardware conocida como POST (Power-On Self-Test). Durante esta fase, la BIOS verifica si los componentes principales, como la memoria RAM, el procesador, los controladores de disco y otros dispositivos, funcionan correctamente. Si se detecta algún problema, se pueden emitir pitidos o códigos de error.

Inicialización de Periféricos: La BIOS inicializa y configura los periféricos básicos, como el teclado, el ratón, el monitor y los dispositivos de almacenamiento. Esto asegura que estos dispositivos estén listos para ser utilizados por el sistema operativo.

Detección de Dispositivos de Arranque: La BIOS busca en los dispositivos de almacenamiento (discos duros, unidades USB, CD/DVD) para identificar aquellos que contienen información de arranque válida. Estos dispositivos se enumeran en el "orden de arranque" configurado en la BIOS.

Carga del Gestor de Arranque: Una vez que se detecta un dispositivo de arranque válido, la BIOS carga el gestor de arranque correspondiente, como GRUB en sistemas Linux o el gestor de arranque de Windows. El gestor de arranque es responsable de cargar el sistema operativo real desde el dispositivo de almacenamiento.

Transición al Sistema Operativo: Después de cargar el gestor de arranque, la BIOS pasa el control al gestor de arranque, que luego carga el núcleo del sistema operativo (kernel) en la memoria y permite que el proceso de arranque continúe en el entorno del sistema operativo.

Las bios han sido reemplazadas en muchos sistemas por sistemas más modernos como el UEFI.

- b) **Como lo** mencione justo arriba, Las bios han sido reemplazadas en muchos sistemas por sistemas más modernos como el UEFI. **UEFI USA GPT como mecanismo de particionado.** Este surge de una alianza entre varias compañías con el objetivo de modernizar el proceso de arranque. Esta misma define la ubicación del gestor de arranque, la interfaz entre el gestor de arranque y el firmware. También expone información para los gestores de arranque con:
- ^ Información de hardware y configuración del firmware
 - ^ Punteros a rutinas que implementan los servicios que el firmware ofrece a los bootloaders u otras aplicaciones UEFI
- Provee un BootManager para cargar aplicaciones UEFI (e.j.: Grub) y drivers desde un UEFI filesystem

Chat gpt respondiendo lo de arriba:

Las afirmaciones que mencionas se refieren a dos funciones clave de UEFI (Interfaz de Firmware Extensible Unificada) en relación con el proceso de arranque de una computadora:

1. ****Define la ubicación del gestor de arranque**:** UEFI permite definir de manera clara y precisa la ubicación del gestor de arranque en la computadora. Esto se logra a través de información almacenada en el firmware UEFI que especifica dónde se encuentra el gestor de arranque en el sistema. Esta ubicación puede ser una partición específica del disco duro o una unidad de almacenamiento designada, y esta información se registra en la NVRAM (Memoria de Acceso No Volátil) del firmware UEFI.

Esta característica es útil porque permite a UEFI saber dónde buscar el gestor de arranque cuando se inicia la computadora, lo que simplifica el proceso de arranque y elimina la necesidad de buscar en múltiples ubicaciones o realizar configuraciones complejas.

2. ****Define la interfaz entre el gestor de arranque y el firmware**:** UEFI establece una interfaz estandarizada y bien definida entre el gestor de arranque y el firmware del sistema (UEFI firmware). Esta interfaz especifica cómo el gestor de arranque debe comunicarse con el firmware UEFI para llevar a cabo el proceso de arranque.

Define los protocolos, las rutinas y las funciones que el gestor de arranque puede utilizar para interactuar con el firmware.

Esta interfaz estandarizada es esencial porque garantiza la compatibilidad entre diferentes gestores de arranque y firmware UEFI. Los desarrolladores de gestores de arranque pueden utilizar esta interfaz para asegurarse de que sus aplicaciones funcionen en sistemas que utilizan UEFI, sin importar el fabricante o el modelo de la placa base.

En resumen, UEFI desempeña un papel fundamental al definir la ubicación del gestor de arranque en el sistema y al establecer una interfaz estándar que permite una comunicación eficiente y coherente entre el gestor de arranque y el firmware UEFI. Estas características son parte de lo que hace que UEFI sea una tecnología más avanzada y versátil en comparación con el BIOS tradicional en el proceso de inicio de una computadora.

Lo que estás describiendo se refiere a algunas de las características y funciones de UEFI (Interfaz de Firmware Extensible Unificada) en el contexto del inicio del sistema operativo. Aquí tienes una explicación más detallada:

1. ****Exponer información para los gestores de arranque****: UEFI proporciona información detallada sobre la configuración del hardware y el firmware de la computadora a los gestores de arranque, como Grub. Esto permite que los gestores de arranque tomen decisiones informadas sobre cómo arrancar el sistema operativo de manera eficiente.
2. ****Punteros a rutinas que implementan los servicios del firmware****: UEFI ofrece una serie de servicios que son utilizados por los gestores de arranque y otras aplicaciones UEFI. Estos servicios incluyen acceso a hardware, configuración del firmware y otras operaciones de bajo nivel. Los gestores de arranque pueden acceder a estas rutinas a través de punteros proporcionados por UEFI.
3. ****BootManager para cargar aplicaciones UEFI****: UEFI incluye un BootManager que permite cargar aplicaciones UEFI y controladores desde un sistema de archivos UEFI. Por ejemplo, Grub, que es un gestor de arranque ampliamente utilizado en sistemas Linux, se puede considerar una aplicación UEFI y puede residir en un sistema de archivos UEFI (generalmente formateado en FAT32). Esto simplifica el proceso de arranque, ya que el gestor de arranque y los controladores necesarios pueden residir en un solo lugar.
4. ****El bootloader como una aplicación UEFI****: Con UEFI, el bootloader (como Grub) se convierte en una aplicación UEFI en lugar de un programa independiente que realiza el arranque en varias etapas, como en el caso del BIOS tradicional. Esta transición a una aplicación UEFI simplifica el proceso de arranque y permite una mayor flexibilidad en la forma en que se gestionan los arranques y los controladores.

En resumen, UEFI ofrece una infraestructura más avanzada y versátil para la gestión del inicio del sistema operativo. **Permite una mayor interacción entre el firmware,**

los gestores de arranque y las aplicaciones UEFI, lo que simplifica y mejora el proceso de arranque de la computadora y proporciona una mayor flexibilidad para cargar aplicaciones y controladores necesarios.

El Secure Boot es una característica del UEFI (Unified Extensible Firmware Interface) que tiene como objetivo mejorar la seguridad del proceso de arranque de una computadora.

- c) **El MBR es el primer sector de un dispositivo de arranque (sector de arranque).**
Además es un sector privilegiado del disco que contiene al MBC (ocupa 446 bytes) y a la tabla de particiones (de 64 bytes).
El MBC es un pequeño código que permite arrancar el S.O. **El MBC lo que hace es examinar la tabla de particiones y se fija cual es la partición que está marcada como booteable y va a ejecutar lo que se encuentra en el primer sector de la partición que está marcada como booteable.**
- d) **CONSULTAR.** Las siglas GPT significan (GUID partition table). GPT es un estilo moderno de particionamiento de discos que ha reemplazado al tradicional MBR en muchas computadoras. La principal diferencia entre ambos está en la forma en que se estructuran y gestionan las particiones en el disco. GPT reemplaza a MBR como una forma más avanzada de organizar particiones en un disco. GPT especifica la ubicación y formato de la tabla de particiones en un disco duro.
- e) **La finalidad del bootloader** (gestor de arranque) es la de cargar una imagen de Kernel (sistema operativo) de alguna partición para su ejecución. Esta se ejecuta luego del código del BIOS. Es un programa que permite cargar el S.O.
Generalmente se utilizan los cargadores multietapas, en los que varios programas pequeños se van invocando hasta lograr la carga del SO.
Existen 2 modos de instalación, uno es en el MBR y el otro en el sector de arranque de la partición raíz o activa. Ejemplos: GRUB, LILO, NTLDR, GAG, YaST, etc.
En cuanto a donde se instalan, eso depende, pero por lo general se instala en la partición donde está el sistema operativo.

ChatGPT: El gestor de arranque es un programa especializado que se encarga de iniciar el sistema operativo en una computadora al arrancar. Su función principal es cargar el sistema operativo en la memoria RAM y transferir el control al Kernel del sistema operativo, lo que inicia el funcionamiento completo de la computadora. Además permite opciones de arranque y te permite seleccionar diferentes sistemas operativos o modos de arranque.

- f) **NO ENTENDI, NO SERIA LO MISMO Q EL INCISO G?**

g) Proceso de arranque:

- 1 Se empieza a ejecutar el código del BIOS
- 2 El BIOS ejecuta el POST
- 3 El BIOS lee el sector de arranque (MBR) (el sector de arranque es el primer sector del disco donde se encuentra el mbr)
- 4 Se carga el gestor de arranque (MBC)
- 5 El bootloader carga el kernel y el initrd
- 6 Se monta el initrd como sistema de archivos raíz y se inicializan componentes esenciales (ej.: scheduler)
- 7 El Kernel ejecuta el proceso init y se desmonta el initrd
- 8 Se lee el /etc/inittab
- 9 Se ejecutan los scripts apuntados por el runlevel 1
- 10 El final del runlevel 1 le indica que vaya al runlevel por defecto
- 11 Se ejecutan los scripts apuntados por el runlevel por defecto
- 12 El sistema está listo para usarse.

Mejor explicado (lo saque de una resolución):

Al encender el ordenador, en lo que se denomina un arranque en frío, éste se comprueba a sí mismo usando un código que se denomina POST (Power On Self Test). El objetivo de dicha prueba es verificar que los elementos imprescindibles para su funcionamiento están presentes y disponibles (la memoria RAM, por ejemplo, entre otros). Dados los escasos recursos con los que cuenta un ordenador al que pudieran fallarle partes muy básicas, los fallos suelen ser notificados al usuario por medio de indicaciones acústicas a través del zumbador o altavoz que suele llevar toda placa base, usando secuencias de pitidos predeterminadas para cada tipo de fallo. Si la prueba se pasa con éxito, se ejecuta a continuación un programa denominado "bootstrap loader" (cargador de inicio) que reside en la memoria de la BIOS. El objetivo del programa cargador es encontrar un sector de arranque en alguno de los dispositivos de almacenamiento disponibles (antes tan sólo discos duros o flexibles, pero hoy en día es posible arrancar de prácticamente cualquier dispositivo de almacenamiento - llaves USB, tarjetas CF (Compact Flash) - que pueda montar como disco, o incluso desde una red local). Un sector de arranque es el primer sector de un disco y contiene un pequeño programa que es capaz de cargar un Sistema Operativo. Los sectores de arranque están marcados con la firma 0xAA55 en los bytes 0x1FE y siguiente. Esto es lo que comprueba el programa cargador. El cargador de inicio tiene una lista de lugares en que buscar un sector de arranque. Esto es lo que se programa cuando en la BIOS seleccionamos los dispositivos de los que arrancar y su orden. El pequeño programa que se ejecuta desde el sector de arranque puede ser de varios tipos. En el caso de Linux, el más frecuentemente usado era LILO, pero ya hace tiempo que se usa en bastantes distribuciones un cargador alternativo, llamado GRUB. Otros Sistemas Operativos tienen su propio programa cargador. Usaremos LILO en la descripción, pues es más ilustrativo. En el caso concreto de LILO, lo que se carga en el sector de arranque es una parte de éste, denominada "first stage boot loader" (primer paso del cargador de

inicio). Su misión es cargar y ejecutar el segundo paso del cargador de inicio. Esta segunda parte suele mostrar una selección de Sistemas Operativos a cargar, procediendo a cargar a continuación el sistema escogido por el usuario (o bien el que se haya predeterminado como sistema por defecto, tras un tiempo de espera, si no escogemos nada). Esta información está incluida dentro del cargador de inicio y, para introducirla, se usa la orden 'lilo' que a su vez usa el contenido de '/etc/lilo.conf'. Todo ello sucede, por supuesto, con el ordenador ya en marcha. Una vez LILO ha cargado el "kernel" (núcleo) de Linux, le pasa el control a éste. Al cargarlo, le ha pasado algunos parámetros. De éstos, el más importante es el que le dice al núcleo qué dispositivo usar como sistema de ficheros raíz, es decir, lo que en UNIX se denomina '/'. En un ordenador de sobremesa, la raíz sería típicamente una partición de un disco duro, pero en sistemas incrustados es frecuente usar como raíz una partición virtual basada en memoria (Flash, RAM,...). Si el núcleo ha conseguido montar el sistema de ficheros raíz, lo siguiente a ejecutar es el programa 'init'. Sólo si dicho programa es estático (es decir, no usa librerías de funciones externas), no será necesario tener acceso a dichas librerías en la raíz. La librería básica en todo sistema GNU/Linux es la librería estándar C, "glibc". En un sistema mínimo, es decir, con una funcionalidad muy concreta, inmutable y sencilla, con tener solamente el programa 'init' enlazado estáticamente sería suficiente (y el núcleo, claro). En ese caso, init sería en realidad nuestro Programa de aplicación al completo. En general, 'init' es sólo el programa que se encarga de arrancar el resto de procesos que la máquina debe ejecutar. Entre sus tareas está el comprobar y montar sistemas de archivos, así como iniciar programas servidores (daemons) para cada función necesaria. Otra tarea importante es la de arrancar procesos 'getty' cuya misión es proporcionar consolas donde poder registrarse y entrar en el sistema. Las órdenes a seguir por 'init' están en el fichero '/etc/inittab'. A partir de ese punto, y en función del sistema de inicialización utilizado (el más frecuente es el denominado "System V") el proceso seguido por 'init' es distinto, pero en el fondo obedece más a un factor de forma, es decir, a una estrategia de ordenamiento de los "scripts" de inicialización de los distintos procesos que a un factor de fondo. Una vez iniciados todos los servidores y procesos de entrada de usuario, o bien estamos delante de una consola de texto en la que el ordenador nos pide que nos identifiquemos, o bien estamos ante una consola gráfica que nos pide lo mismo, o bien estamos ante una pantalla llena de opciones sobre qué ejecutar (escuchar música, ver películas, por ejemplo) si el sistema arranca bajo un usuario predeterminado y no nos pide registrarnos. Esto es, si es que hablamos de un ordenador de sobremesa que, típicamente, nos ofrece una interfaz basada en dispositivos de entrada (teclado, ratón, mando a distancia) y de salida (monitor, TV, audio) para interactuar con él. Pero si el ordenador que se ha iniciado es un dispositivo con una funcionalidad concreta y su misión es controlar una serie de procesos y accedemos a él a través de medios indirectos (como pueda ser un navegador Web), el ordenador se inicia cuando está en disposición de prestar sus servicios, aun cuando no haya una indicación visual de dicho estado.

h) **(NO APARECE EN LOS MATERIALES DADOS). CHAT GPT:**

El proceso de apagado (shutdown) en un sistema operativo GNU/Linux implica una serie de pasos para detener de manera ordenada todos los servicios y componentes del sistema antes de apagar o reiniciar la computadora. Aquí están los pasos generales que se suceden durante el proceso de parada:

Finalización de Procesos y Servicios: El sistema operativo envía señales a todos los procesos en ejecución para que finalicen correctamente. Los procesos pueden guardar datos, cerrar conexiones de red y realizar otras tareas de limpieza antes de terminar.

Desmontaje de Sistemas de Archivos: El sistema operativo comienza a desmontar todos los sistemas de archivos montados para asegurarse de que todos los cambios se hayan guardado en el disco. Esto incluye el sistema de archivos raíz (/) y cualquier otro sistema de archivos montado en puntos de montaje específicos.

Detención de Servicios y Demonios: Los servicios en segundo plano (daemons) que se ejecutan en el sistema, como servidores web, bases de datos y otros servicios, se detienen de manera ordenada. Esto garantiza que los datos se cierren adecuadamente y que no se produzcan pérdidas ni corrupciones.

Apagado de Dispositivos: Los dispositivos hardware, como discos duros, tarjetas de red y otros periféricos, se apagan o se ponen en un estado de bajo consumo de energía.

Cierre de Red: Las conexiones de red se cierran, se liberan direcciones IP y se desactivan interfaces de red.

Desactivación de Controladores y Módulos: Se desactivan los controladores y módulos del kernel que ya no son necesarios.

Finalización del Kernel: El núcleo del sistema operativo (kernel) realiza tareas finales, como liberar recursos en memoria y realizar otros procesos de limpieza.

Envío de la Señal de Apagado o Reinicio: Una vez que todos los procesos y servicios se han detenido y los componentes del sistema se han apagado correctamente, se envía una señal al hardware para apagar o reiniciar la computadora.

Apagado o Reinicio del Hardware: El hardware recibe la señal y procede a apagar o reiniciar la computadora.

POST (Power-On Self-Test) en el Arranque: Al encender la computadora nuevamente, la BIOS realiza una autoevaluación del hardware.

i) Es posible, a esto se lo conoce como arranque dual. Al realizar una partición, esta misma actúa como una entidad independiente, y puede tener su propio sistema de archivos, sistema operativo y datos. Por ende, para lograr tener varios S.O tenes que particionar el disco duro para reservar espacio para cada sistema operativo y luego descargar los sistemas operativos en las particiones correspondientes. Después vas a tener que configurar el gestor de arranque para que se pueda seleccionar que S.O arrancar al encender el PC.

9)

a) (chat gpt). consultar.

En sistemas GNU/Linux, los archivos se identifican principalmente a través de sus nombres y rutas en el sistema de archivos. Aquí hay varias formas en las que los archivos se pueden identificar en GNU/Linux:

Nombres de Archivo: Cada archivo tiene un nombre único que lo identifica dentro de su directorio. Los nombres de archivo pueden contener letras, números,

caracteres especiales y puntos. Los nombres de archivo son sensibles a mayúsculas y minúsculas, lo que significa que "archivo.txt" y "Archivo.txt" se consideran diferentes.

Rutas de Archivo: Los archivos se encuentran dentro de directorios (carpetas). La ruta de archivo es la secuencia de directorios desde el directorio raíz hasta el archivo en cuestión. Por ejemplo, "/home/usuario/documentos/archivo.txt" es una ruta de archivo completa que indica la ubicación exacta del archivo.

Permisos y Propietarios: Cada archivo en GNU/Linux tiene asociados permisos que controlan quién puede acceder y modificar el archivo. También tiene un propietario y un grupo propietario. Los grupos propietarios son un mecanismo para administrar el acceso y los permisos de varios usuarios de manera eficiente. Los usuarios que son parte de un grupo propietario específico comparten ciertos permisos en los archivos y directorios asociados con ese grupo. Estos atributos son parte integral de la identificación de un archivo.

Números de Inodo: En el sistema de archivos ext utilizado en la mayoría de las distribuciones de GNU/Linux, cada archivo se asocia con un número de inodo. Los números de inodo son identificadores únicos asignados por el sistema de archivos y se utilizan internamente para rastrear y gestionar los archivos.

Enlaces Duros y Simbólicos: Enlaces duros y enlaces simbólicos son métodos para identificar un archivo por más de un nombre. Un enlace duro crea una copia del archivo que se puede acceder a través de múltiples nombres. Un enlace simbólico crea un "atajo" a un archivo y puede residir en una ubicación diferente.

Metadatos del Archivo: Además de los atributos mencionados anteriormente, los archivos también tienen otros metadatos, como la fecha de creación, la fecha de modificación y el tamaño del archivo. Estos metadatos también ayudan a identificar un archivo de manera única.

En resumen, los archivos en GNU/Linux se identifican por sus nombres, rutas, permisos, propietarios, números de inodo y otros atributos. Estos elementos en conjunto permiten al sistema operativo gestionar y localizar archivos de manera precisa y eficiente.

b)

Los editores vi y mcedit (editores de texto, **vi no es lo mismo que vim, son ambos editores de texto pero tienen diferencias**), así como los comandos cat y more, son herramientas utilizadas en sistemas GNU/Linux para trabajar con archivos de texto y visualizar su contenido. Cada uno tiene su propia función y características. Aquí hay una descripción de cómo funcionan:

Editor vi: Vi es un editor de texto en modo terminal que es ampliamente utilizado en sistemas UNIX y GNU/Linux. Tiene varias modalidades, como el modo de comandos y el modo de edición, que permiten realizar diversas acciones en los archivos. Para abrir un archivo con vi, escribes vi nombre-del-archivo en la terminal. Una vez abierto, puedes:

- Cambiar al modo de edición presionando la tecla i para insertar texto.
- Realizar ediciones, como agregar, modificar o eliminar contenido.
- Guardar los cambios presionando Esc y luego escribiendo :w y presionando Enter.
- Salir del editor escribiendo :q y presionando Enter.

- Guardar y salir simultáneamente escribiendo :wq y presionando Enter.

Editor mcedit (Midnight Commander): mcedit es el editor de texto incorporado en Midnight Commander, un administrador de archivos en modo terminal. Proporciona una interfaz más amigable en comparación con vi. Puedes abrir mcedit desde Midnight Commander seleccionando un archivo de texto y presionando la tecla F4. En mcedit, puedes:

- Editar el contenido del archivo de manera similar a otros editores de texto.
- Guardar los cambios presionando F2.
- Salir del editor presionando F10.

Comando cat: El comando cat se utiliza para mostrar el contenido completo de un archivo en la terminal. Simplemente escribes cat nombre-del-archivo y el contenido se mostrará en la pantalla. Sin embargo, cat muestra todo el contenido de una vez, lo que puede ser abrumador para archivos largos.

Comando more: El comando more permite visualizar el contenido de un archivo de manera paginada, lo que facilita la lectura de archivos largos. Simplemente escribes more nombre-del-archivo y podrás navegar a través del contenido usando las teclas de flecha. Presiona q para salir de more.

En resumen, vi y mcedit son editores de texto en modo terminal que te permiten editar archivos de texto directamente desde la línea de comandos. Por otro lado, los comandos cat y more se utilizan para visualizar el contenido de archivos de texto. Cada herramienta tiene sus propias características y propósitos, por lo que puedes elegir la que mejor se adapte a tus necesidades y preferencias.

c)

Para crear un archivo llamado "prueba.exe" en tu directorio personal utilizando el editor de texto "vi" en Debian y añadir tu número de alumno y nombre, sigue estos pasos:

Abre una terminal en Debian.

Asegúrate de estar en tu directorio personal. Puedes usar el comando cd para ir al directorio personal:

bash

Copy code

cd ~

Crea el archivo "prueba.exe" utilizando el editor "vi":

bash

Copy code

vi prueba.exe

Dentro del editor "vi", presiona i para entrar en modo de inserción. Ahora puedes comenzar a escribir en el archivo.

Escribe tu número de alumno y nombre en el archivo, por ejemplo:

yaml

Copy code

Número de Alumno: 12345

Nombre: Tu Nombre

Una vez hayas ingresado la información, presiona Esc para salir del modo de inserción.

Escribe :wq y presiona Enter para guardar los cambios y salir del editor "vi".

Ahora tendrás un archivo llamado "prueba.exe" en tu directorio personal, con tu número de alumno y nombre dentro. Cabe mencionar que en Linux, las extensiones de archivo como ".exe" no tienen el mismo significado que en Windows. Puedes usar cualquier nombre de archivo que desees, independientemente de su extensión.

d) **OJO NO LO PROBE, LO CONSULTE A CHAT GPT (IGUAL ESTE COMANDO DICE QUE FUNCIONA IGUAL PARA TODAS LAS DISTRIBUCIONES DE LINUX), PROBAR IGUAL.** El comando **file** en Linux se utiliza para determinar el tipo de archivo y mostrar información sobre su contenido. Puedes usar el comando file en la línea de comandos junto con el nombre de un archivo como argumento. Al ejecutarlo, el comando file intentará identificar el tipo de archivo basándose en su contenido y mostrará una descripción del archivo.

10)a) "mkdir nombre".

b) acceder a la carpeta: "cd nombre de la carpeta".

(c) Cree dos archivos con los nombres iso2017-1 e iso2017-2 (touch). El comando touch se utiliza para crear archivos vacíos o actualizar las marcas de tiempo de los archivos existentes. Aquí tienes cómo crear esos dos archivos:

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

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

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

El comando find en sistemas Unix y Linux se utiliza para buscar archivos y directorios en una jerarquía de directorios especificada. Es una herramienta poderosa que permite buscar archivos en función de diferentes criterios, como el nombre del archivo, el tipo de archivo, la fecha de modificación y más. El comando find es muy útil para tareas como la administración de archivos, la búsqueda de archivos específicos y la ejecución de operaciones en lotes en archivos que cumplen ciertos criterios.

El formato básico del comando `find` es:

```
find directorio -criterios
```

Algunos ejemplos de cómo se puede usar el comando `find` incluyen:

- Buscar archivos por nombre:

```
find /ruta/del/directorio -name "nombre_del_archivo"
```

-
- Buscar archivos por tipo:
- bash
- Copy code

```
find /ruta/del/directorio -type f
```

-
- Buscar directorios por nombre:

```
find /ruta/del/directorio -type d -name "nombre_del_directorio"
```

-
- Buscar archivos modificados en los últimos días:

```
find /ruta/del/directorio -type f -mtime -n
```

-
- Realizar operaciones en lotes en archivos encontrados:
- `find /ruta/del/directorio -name "*.txt" -exec rm {} \;`

- (g) Informar la cantidad de espacio libre en disco (df)
- (h) Verifique los usuarios conectado al sistema (who)
- (i) Acceder a el archivo iso2017-1 e ingresar Nombre y Apellido
- (j) Mostrar en pantalla las últimas líneas de un archivo (tail "tail archivo"):

```
luciano@Debian:~$ tail prueba.exe
Numero de alumno: 12345
Nombre: Tu nombre
luciano@Debian:~$
```

11) y 12) (chat gpt), **ojo (PROBAR)** que habla para ubuntu linux lo que puse abajo, pq no lo pude probar, por ende si quieres para el ejercicio 12 saber para cada uno de los comandos en qué directorio están tenes que usar el comando **"which + comando"** la cual te da la ruta del mismo.

(a) **shutdown**: Este comando se utiliza para apagar o reiniciar el sistema de manera controlada. Puedes usar opciones como -h para apagar y -r para reiniciar.

- (b) **reboot**: Este comando reinicia el sistema. Puedes usarlo simplemente escribiendo `reboot`.
- (c) **halt**: El comando "halt" se utiliza para detener el sistema y apagarlo de manera controlada.
- (d) **locate**: "locate" se utiliza para buscar archivos en la base de datos de nombres de archivos. Los resultados son rápidos pero pueden no ser siempre los más actualizados.
- (e) **uname**: "uname" muestra información sobre el sistema. Puedes usar opciones como `-a` para mostrar toda la información.
- (f) **dmesg**: Este comando muestra el registro del kernel, que contiene mensajes del sistema y del hardware.
- (g) **lspci**: Este comando muestra información detallada sobre los dispositivos PCI conectados al sistema.
- (h) **at**: El comando "at" se usa para programar comandos para que se ejecuten en el futuro.
- (i) **netstat**: "netstat" muestra información sobre las conexiones de red, tablas de enrutamiento, estadísticas de interfaz, etc.
- (j) **mount**: "mount" se usa para montar sistemas de archivos, como particiones o dispositivos externos, en un directorio del sistema.
- (k) **umount**: "umount" se utiliza para desmontar sistemas de archivos previamente montados.
- (l) **head**: "head" muestra las primeras líneas de un archivo de texto. Por defecto, muestra las primeras 10 líneas.
- (m) **losetup**: "losetup" se utiliza para configurar y administrar bucles de dispositivos. Los bucles de dispositivos permiten que los archivos se utilicen como dispositivos de bloques.
- (n) **write**: "write" se utiliza para enviar mensajes a otros usuarios que están conectados al sistema.
- (ñ) **mkfs**: "mkfs" se usa para crear sistemas de archivos en particiones.
- (o) **fdisk**: "fdisk" es una herramienta para crear y administrar particiones en discos duros.
- La mayoría de estos comandos están ubicados en los directorios del sistema que se encuentran en la variable de entorno **PATH**. Los binarios de estos comandos generalmente se encuentran en `/bin` o `/usr/bin`. Puedes usar el comando **which** para obtener la ruta completa de un comando específico. Por ejemplo: **which** shutdown.