Trabajo Práctico N°1

1. Características de GNU/Linux

a.

- Multiusuario: permite que varios usuarios se conecten y operen en él simultáneamente, compartiendo recursos como el procesador, la memoria y el almacenamiento.
- Multitarea: permite ejecutar varios programas o tareas de forma que parecen ejecutarse simultáneamente, aunque en realidad es una rápida alternancia en la cual el procesador comparte su tiempo disponible entre diferentes tareas.
- Multiprocesador: permite administrar más de un procesador en el caso de que el hardware cuente con varios procesadores independientes que trabajan paralelamente. Esto mejora el rendimiento general y la capacidad de respuesta del sistema.
- Altamente portable: está diseñado para poder ejecutarse en diferentes arquitecturas de hardware o plataformas sin requerir modificaciones importantes en su código o configuración.
- Posee diversos intérpretes de comandos, de los cuales algunos son programables.
- Permite el manejo de usuarios y permisos.
- Todo es un archivo (incluso dispositivos y directorios).
- Cada directorio puede estar en una partición del disco diferente.
- Es case sensitive.
- Es de código abierto.
- Posee desarrollo en capas, lo cual permite la separación de funciones. Cada capa actúa como una caja negra respecto a las demás. Esto posibilita el desarrollo distribuido.
- Soporte para diversos file systems: significa que puede leer, escribir y organizar datos en dispositivos de almacenamiento que usan distintas metodologías de estructura y gestión de datos (NTFS en Windows, APFS en macOS, EXT4 en Linux). Esto permite un alto grado de interoperabilidad.
- Memoria virtual: RAM + SWAP. Puede aparentar tener más memoria RAM de la que realmente tiene físicamente, haciendo uso de un espacio del disco como extensión de la RAM, a través de un proceso de intercambio llamado swapping.
- Su desarrollo está hecho mayoritariamente en C y Assembler.

b. **Windows:**

- Es multiusuario, multitarea y multiprocesador.
- No es altamente portable, ya que no se adapta fácilmente a diferentes arquitecturas de hardware. Sin embargo, se pueden crear versiones portátiles haciendo uso de herramientas de terceros.
- También permite el manejo de usuarios y permisos.
- Los directorios están siempre ubicados en lugares predeterminados dentro del disco, no pueden ubicarse en diferentes particiones.
- Es case insensitive.
- Es software propietario.
- Soporta varios sistemas de archivos nativamente (NFTS, FAT32, exFAT, ReFS). Sin embargo, para leer o escribir en file systems pertenecientes a otros sistemas operativos, es necesario el uso de programas de terceros. Por lo tanto no es altamente interoperable.
- Su desarrollo está hecho en C, C++ y una pequeña parte en Assembler.

c. GNU es un sistema operativo basado en Unix que incluye una colección de herramientas, bibliotecas, aplicaciones y utilidades, como compiladores y editores de texto.

d. GNU es un proyecto creado por Richard Stallman en 1983, con el objetivo de desarrollar un sistema operativo completo y libre, compatible con Unix. Para asegurar que el mismo fuera libre, fue necesario crear un marco regulatorio conocido como GPL.

En 1990, GNU ya contaba con un editor de textos (Emacs), un compilador (GCC) y una gran cantidad de bibliotecas que componen un Unix típico. Lo único que faltaba era el componente más importante: el núcleo.

Si bien se venía trabajando en un núcleo conocido como TRIX, en 1988 se decide abandonarlo debido a que corría en hardware muy costoso. Entonces se adopta el núcleo MACH para crear GNU Hurd, pero este proyecto tampoco prosperó.

Desde el año 1991, Linus Torvalds estaba trabajando en el desarrollo de un núcleo conocido como Linux, el cual se distribuiría también bajo la licencia GPL.

Finalmente, en el año 1992, Richard Stallman y Linus Torvalds deciden fusionar ambos proyectos, dándole vida a GNU/Linux.

e. Ya está respondida en el inciso a.

f. POSIX significa “Portable Operating System Interface” y es un conjunto de estándares desarrollados por el IEEE para asegurar la compatibilidad entre diferentes sistemas operativos, especialmente aquellos basados en Unix. En esencia, define como un sistema operativo debe comportarse y cómo las aplicaciones deben interactuar con él, permitiendo que el software escrito para un sistema compatible con POSIX funcione en otros sistemas que también sigan el estándar.

2. Distribuciones de GNU/Linux

   a. Si bien el núcleo es el mismo, las distribuciones de Linux varían en cómo se configura y qué software adicional se incluye para adaptarse a diferentes necesidades, como servidores, escritorios o dispositivos específicos.

      - **Ubuntu**: Amigable para el usuario, con un gran enfoque en la facilidad de uso y una gran comunidad.
      - **Debian**: Conocida por su gran enfoque en el software libre, la estabilidad y es una excelente opción para servidores.
      - **Fedora**: Patrocinada por Red Hat, ofrece software de vanguardia y es un banco de pruebas para tecnologías futuras.
      - **Arch**: Para usuarios avanzados que desean un sistema mínimo y un control total sobre la configuración.

   b. -

   c. Debian es una de las primeras distribuciones de Linux. A diferencia de otras distribuciones, en vez de estar desarrollada por un individuo aislado o un grupo, Debian se desarrolla abiertamente en el espíritu de GNU/Linux. La creación de Debian fue un intento por crear una distribución no comercial que fuera capaz de competir efectivamente en el mercado comercial.

La principal prioridad de Debian está en proporcionar un producto de primera fila y no en los beneficios o los ingresos, y el margen de los productos o los servicios puede usarse para mejorar el software en sí para todos los usuarios, hayan pagado por su Debian o no.

3. Estructura de GNU/Linux

   a. Los tres componentes fundamentales de GNU/Linux son el kernel (núcleo), el Shell (intérprete de comandos) y el File System (sistema de archivos).

   b. El **kernel** podría definirse como el corazón del sistema operativo. A grandes rasgos, es el encargado de que el software y el hardware de un sistema puedan trabajar juntos.

      El **Shell** es el programa que lee las entradas del usuario en la terminal y las traduce a instrucciones que el sistema es capaz de entender y utilizar.

      El **Filesystem** es la forma en la que los archivos se organizan y administran dentro de un sistema operativo.

4. Kernel

   a. Las funciones principales del kernel son la administración de memoria, E/S y CPU.

   b. La versión actual del kernel es la 6.16.3. El esquema de versionado está compuesto de la siguiente forma: A.B.C[.D].

      A: denota la versión del kernel. Cambia con muy poca frecuencia.
      B: denota el mayor nivel de revisión del kernel.
      C: indica revisión menor del kernel. Es cambiado cuando se introducen nuevos drivers o características.
      D: maneja bug-fixes y parches de seguridad que no implican cambio suficiente para lanzar una nueva revisión.

      Antes de la versión 2.6, los números pares en el dígito B (1.2, 2.4, 2.6) indicaban la versión “estable” lanzada. Los números impares (2.5), en cambio, indicaban

versiones de desarrollo que no son consideradas producción. Sin embargo, desde la versión 2.6, no hay gran diferencia entre los números pares e impares.

c. Sí, es posible tener más de un kernel de Linux instalado en la misma computadora. Al arrancar el sistema te permite seleccionar qué kernel usar. No es necesario que estén instalados en distintas particiones del disco.

d. El kernel de Linux se encuentra principalmente en el directorio /boot, donde se encuentra la imagen del kernel (vmlinuz) y otros archivos de arranque. Además, el directorio /proc es un sistema de archivos virtual que proporciona información sobre el kernel y la configuración del sistema en tiempo de ejecución.

5. Shell

a. El Shell es el intérprete de comandos (consola o terminal) que actúa como interfaz entre el usuario y el sistema operativo.

b. Consta de una ventana que espera comandos textuales ingresados por el usuario, los cuales traduce e interpreta y los entrega al kernel para su ejecución.

c. Intérpretes de GNU/Linux:

- Bourne Shell (sh): disponible en todas las versiones de Unix y lo suficientemente básico para funcionar en todas las plataformas.

- Korn Shell (ksh): estándar de SYSV (una de las versiones de Unix). Maneja un historial de comandos. Está basada en sh, con agregados para ser más amigable.

- Bourne Again Shell (bash): Uno de los shell más avanzados y populares. Ofrece las mismas capacidades que csh (derivado del lenguaje C, permite utilizar una sintaxis parecida en scripts), pero incluye funciones avanzadas: historial de comandos que se conserva al pasar de una sesión a otra, accesible con los cursores (arriba/abajo), auto completado de nombres de comandos o archivos presionando TAB, manejo de varios tipos de redirecciones de E/S.

d. Los comandos internos vienen integrados con la Shell y se encuentran ubicados en el directorio /bin, mientras que los comandos externos pueden encontrarse ubicados en los directorios /bin, /usr/bin, /usr/local/bin o cualquier otra ubicación si se la agrega a la variable PATH.
e. El Shell no forma parte del kernel, sino más bien funciona como intermediario entre el usuario y el kernel, comunicándole a este último los comandos que el usuario desea ejecutar.

f. Sí, es posible definir un intérprete de comandos diferente para cada usuario. El mismo se define principalmente desde el archivo /etc/passwd, que contiene una línea para cada usuario, siendo el último campo de la línea el Shell predeterminado.

   También se puede asignar un shell al crear un usuario con el comando useradd y la opción --shell, o cambiarlo para un usuario existente usando el comando chsh

   El usuario administrador (root) o un usuario con permisos administrativos son los que pueden llevar a cabo esta tarea. Sin embargo, cada usuario puede modificar particularmente su propio Shell (con chsh) si tiene los permisos adecuados.

6. Filesystem

a. El Filesystem es la forma es la que se organizan y administran los archivos dentro del sistema.

b. La FHS (Filesystem Hierarchy Standard) define los directorios principales y sus contenidos en el sistema GNU/Linux y otros sistemas de la familia Unix. Se diseñó en 1994 para estandarizar el sistema de archivos de las distribuciones GNU/Linux.

   Los siguientes directorios son esenciales para que el sistema esté operativo, a excepción de /usr que la mayoría de su contenido es opcional para el sistema.

   - /bin: bin es la abreviación de binaries, o ejecutables. Es donde residen la mayoría de los programas esenciales del sistema, como cp, ls y mv.
   - /dev: Los archivos en /dev son conocidos como controladores de dispositivo (device drivers), son usados para acceder a los dispositivos del sistema y recursos, como discos duros, memoria, etc.
   - /etc: contiene una serie de archivos de configuración del sistema. Estos incluyen /etc/passwd (la base de datos de usuarios), /etc/rc (scripts de inicialización del sistema), etc.

- /sbin: se usa para almacenar programas esenciales del sistema, que usará el administrador del sistema.
- /home: contiene los directorios "home" de los usuarios. Por ejemplo, /home/ISO_CSO es el directorio del usuario ISO_CSO.
- /lib: contiene las imágenes de las librerías compartidas. Estos archivos contienen códigoque compartirán muchos programas.
- /proc: es un "sistema de ficheros virtual". Los ficheros que contiene realmente residen en la memoria, no en un disco. Hacen referencia a varios procesos que corren en el sistema, y le permiten obtener información acerca de que programas y procesos están corriendo en un momento dado.
- /root: Directorio home de root.
- /tmp: Muchos programas tienen la necesidad de generar cierta información temporal y guardarla en un fichero temporal. El lugar habitual para esos ficheros es en /tmp.
- /usr: es un directorio muy importante. Contienen una serie de subdirectorios que contienen a su vez algunos de los más importantes y útiles programas y archivos de configuración usados en el sistema.

c. Linux es muy versátil y soporta una gran cantidad de sistemas de archivos, tanto los nativos como el Ext4, como los de otros sistemas como NTFS (Windows) y HFS/APFS (macOS), además de sistemas para redes como NFS y sistemas virtuales como proc y tmpfs.

d. Sí, Linux es altamente interoperable en cuanto a filesystems, permitiendo leer y escribir en múltiples sistemas de archivos pertenecientes a otros sistemas operativos, incluyendo FAT y NTFS de Windows.

7. –

8. Bootstrap

a. El BIOS (Basic Input/Output System) es un firmware que se encuentra en la placa madre y actúa de intermediario entre el hardware y el sistema operativo, siendo crucial para el proceso de arranque de una computadora.

   El BIOS es el responsable de iniciar la carga del MBC desde el MBR.

b. UEFI es un estándar unificado de comunicación entre el hardware y el firmware, propiedad del UEFI Forum, que reemplazó a la BIOS debido a que cuenta con más ventajas.

   Su función es iniciar diferentes componentes de hardware y posteriormente el arranque del sistema. El UEFI BIOS realiza una serie de comprobaciones para ver que todo funciona correctamente en la computadora antes de "entregarle el control" al sistema operativo.

c. El MBR (Master Boot Record) es la información contenida en el primer sector de un disco duro o unidad extraíble. Está constituido por el bootloader (que se encuentra en los primeros 446 bytes de los 512 bytes disponibles), la tabla de particiones (que indica la ubicación y el tamaño de cada partición y ocupa 64 bytes) y la firma de arranque (número mágico de 2 bytes que identifica el MBR).

   El MBC (Master Boot Code) es una parte del MBR cuya función principal es localizar y cargar el gestor de arranque del sistema operativo.

d. El sistema GPT (GUID Partition Table) forma parte del estándar EFI y especifica la ubicación y formato de la tabla de particiones de un disco duro. Puede interpretarse como el sustituto del MBR.
   GPT usa un modo de direccionamiento lógico (logical block addressing). En el LBA 0 está el "MBR heredado" que se mantiene por cuestiones de compatibilidad con el esquema BIOS, en el LBA 1 se encuentra la cabecera GPT y la tabla de particiones se ubica en los bloques sucesivos.
   Tanto la cabecera como la tabla de particiones se encuentran escritas al principio y al final del disco, con el objetivo de reducir la probabilidad de fallos.

e. La función del gestor de arranque es cargar una imagen del kernel de alguna partición para su ejecución. Existen dos modos de instalación del gestor de arranque: en el MBR o en el sector de arranque de la partición activa (Volume Boot Record).
   Algunos ejemplos de gestor de arranque son: GRUB, LILO, NTLDR, etc.

f. El BIOS (Basic I/O System) es el responsable de iniciar la carga del SO a través del MBC. Este 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 y luego pasa el control al proceso init.
El proceso de arranque se ve como una serie de pequeños programas de ejecución encadenada

g. -

h. El comando para finalizar correctamente un sistema Linux es shutdown. Se utiliza generalmente de una de dos maneras diferentes:

- Si existe un único usuario en el sistema, debe finalizar todos los programas que estén en ejecución, finalizar todas las sesiones de todas las consolas virtuales, e iniciar una sesión como usuario root (o mantener la sesión si ya existe una, pero debe cambiar de directorio de trabajo al directorio HOME de root, para evitar problemas al desmontarse los sistemas de archivos). Finalmente se ejecuta el comando shutdown -h now. Si se desea postergar la ejecución del comando shutdown, se debe reemplazar now con un signo + y un número que indica minutos de espera.

- Si el sistema está siendo utilizado por varios usuarios, se debe utilizar el comando shutdown -h +time mensaje, donde time es el número de minutos en que se posterga la detención del sistema, y el mensaje es una explicación breve del porqué se está apagando el sistema. # shutdown -h +10 'We will install a new disk. System should > be back on-line in three hours.' # El ejemplo advierte a todos los usuarios que el sistema se apagará en diez minutos, y que sería mejor que se desconecten o se arriesgan a perder la información con la que están trabajando. La advertencia se muestra en cada terminal donde existe un usuario conectado.

i. Sí, es posible tener GNU/Linux y otro sistema operativo instalados en la misma PC, aunque cada uno debe ser instalado en diferentes discos o diferentes particiones de un disco.

9.

a. En Linux no es necesario que los archivos lleven una extensión, ya que el sistema identifica el tipo de archivo leyendo la firma que se encuentra dentro del archivo, la cual indica su tipo.

10.

a. mkdir
b. cd
c. touch
d. ls
e. pwd
f. find
g. df
h. who / users / w
i. vim nombre_archivo
j. tail

11.

a. El comando man es un manual integrado de Linux que proporciona documentación detallada sobre comandos, archivos y programas. Los parámetros principales son: nombre del comando/archivo y, opcionalmente, un número de sección para buscar categorías específicas. La ubicación de las páginas del manual puede variar, pero por lo general se encuentran en directorios como usr/share/man.

b. El comando shutdown reinicia o detiene el sistema con opciones de tiempo específico, reinicio o apagado completo. Su sintaxis es shutdown [OPCIONES] [HORA] [MENSAJE]. Entre los parámetros más comunes se encuentran: -r para reiniciar, -h para detener/apagar, -P para apagar, -c para cancelar un apagado programado previamente y -k envía mensajes de advertencia y deshabilita los inicias de sesión, sin apagar realmente el sistema. El comando se encuentra generalmente ubicado en /sbin o /usr/sbin

   Ejemplo: sudo shutdown -r +10 "El sistema se reiniciará pronto." El sistema se reiniciará en 10 minutos y se mostrará el mensaje a todos los usuarios conectados.

c. El comando reboot se usa para reiniciar el sistema de forma segura, cerrando procesos y desmotando sistemas de archivos. Puede ir acompañado del parámetro -f o --force para un reinicio forzado, el cual omite el apagado normal (debe usarse con precaución). La ubicación principal del comando se encuentra en /sbin/reboot o /usr/sbin/reboot.

d. El comando halt detiene todas las funciones de la CPU, dejando el sistema en un estado de apagado que puede ser completo o un estado de suspensión con apagado total o parcial, según se configure. Si bien es un comando simple, tiene la opción de ser utilizado con -p o --poweroff que apaga el sistema completamente o --rebot el cual reinicia la máquina. La ubicación del comando es /sbin/halt o /usr/sbin/halt.

e. El comando uname imprime información del sistema, como el nombre del kernel y su versión, arquitectura y nombre del host. Cuenta con varios parámetros:

    a. -a o --al: extiende la información a mostrar.
    b. -s: imprime solo el nombre del kernel.
    c. -n: muestra el nombre del host de la red.
    d. -r: muestra la versión del kernel.
    e. -v: muestra la fecha de compilación del kernel.
    f. -m: muestra el tipo de arquitectura del hardware.
    g. -o: muestra el nombre del sistema operativo (GNU/Linux por ejemplo).

    El comando se encuentra generalmente en una de las ubicaciones estándar para binarios como /bin o /usr/bin.

f. El comando dmesg muestra mensajes del kernel para diagnóstico de hardware y controladores. Los archivos de dmesg no tienen una ubicación fija, ya que se obtiene la información directamente del kernel. Cuenta con varios parámetros:

    a. --level o -l: muestra mensajes de un nivel de gravedad específico.
    b. --human o -H: muestra la salida en formato legible para humanos.
    c. --ctime o -T: agregar marcas de tiempo a cada mensaje para saber cuando ocurrió.
    d. --follow o -w: muestra los mensajes nuevos a medida que se generan, similar a un tail -f.

e. grep <patrón>: permite filtrar la salida para buscar cadenas de texto específicas.

g. El comando lspci lista los dispositivos y buses PCI del sistema, ofreciendo información detallada sobre el hardware conectado. No tiene una ubicación fija como un archivo. También cuenta con varios parámetros:

    a. -v: muestra información detallada sobre los dispositivos, como capacidades y configuraciones.
    b. -vv o -vvv: proporciona aún más detalles, incluyendo registros y datos del kernel.
    c. -n: muestra los IDs numéricos de los dispositivos y fabricantes, útil para buscar en bases de datos.
    d. -nn: combina nombres y IDs numéricos para una mejor identificación.
    e. -k: muestra los módulos del kernel que están siendo utilizados por cada dispositivo.
    f. -m o -mm: genera una salida en formato de “legado” que puede ser interpretada por otros programas.

h. El comando at permite ejecutar comandos o scripts una única vez en un momento futuro específico. La ubicación del comando es /usr/bin/at o /bin/at. Parámetros:

    a. -q: especifica la cola del trabajo (a-z).
    b. -m: envía un correo electrónico al usuario cuando el trabajo se completa.
    c. -f nombre_archivo: lee los comandos desde un archivo.
    d. -V: muestra la versión del comando.

i. El comando netstat muestra el estado de la red, incluyendo conexiones activas, tablas de enrutamiento y estadísticas de interfaces de red y puertos. No tiene una ubicación de archivo específica en el sistema. Parámetros:

    a. -a: muestra todas las conexiones, tanto las activas como la que están en estado de escucha.
    b. -t: muestra solo las conexiones y sockets TCP.
    c. -u: muestra solo las conexiones y sockets UDP.
    d. -n: muestra las direcciones IP y los números de puerto en formato numérico, en lugar de resolver sus nombres.

e. -p: muestra el PID y el nombre del programa asociado a cada conexión.
f. -i: muestra la tabla de enrutamiento de las interfaces de red.
g. -r: muestra la tabla de enrutamiento del sistema.
h. -s: muestra estadísticas de red por protocolo.

j. El comando head muestra la parte superior de un archivo, por defecto las primeras 10 líneas. Se ubica en la ruta /bin/head. Parámetros:

    a. -n N o --lines=N: muestrea las primeras N líneas del archivo.
    b. -c N o --bytes=N: muestra los primeros N bytes del archivo.
    c. -q o --quiet o --silent: desactiva la visualización de los nombres de los archivos cuando se procesan múltiples archivos.
    d. -v o --verbose: fuerza la visualización de los nombres de los archivos cuando se procesas múltiples archivos.

k. El comando tail se usa para mostrar las últimas N líneas de un archivo o la entrada estándar. No tiene una ubicación fija de comando. Parámetros:

    a. -n NÚMERO: muestra las últimas NÚMERO líneas del archivo.
    b. -f: muestra las últimas líneas y luego espera, mostrando cualquier nueva línea que se agregue al final del archivo. Se detiene presionando Ctrl+C.
    c. -f -n NÚMERO: combina las dos opciones anteriores, muestra las últimas NÚMERO líneas y luego sigue monitoreando.
    d. +N: imprime el contenido del archivo a partir de la línea N.

12.

a. Un proceso es una abstracción de un programa en ejecución generada por el sistema operativo para poder administrar su consumo de los recursos de un sistema.

    Todos los procesos cuentan con un atributo conocido como PID (process identification) único asignado por el sistema operativo, el cual los identifica. Todos los procesos, excepto el proceso init (o su equivalente), tienen un PPID (parent process identification) el cual representa al proceso padre que lo inició, estableciendo una relación jerárquica en forma de árbol entre ellos.

Los procesos también cuentan con otros atributos como: su estado, los IDs del usuario y grupo que lo poseen, su prioridad, la línea de comandos que lo inició, los recursos de memoria y CPU que consume y el terminal asociado.

13.

a. 1. Se inicia la BIOS.
   2. El hardware lee el sector de arranque (MBR).
   3. Se carga el gestor de arranque (MBC).
   4. Se monta la imagen del kernel.
   5. Se ejecuta el proceso init.

b. El proceso init es ejecutado por el propio kernel de Linux, siendo el primer proceso que se inicia en el sistema. Su objetivo es inicializar y gestionar todos los servicios, demonios y procesos necesarios para que el sistema operativo funcione correctamente.

c. Los runlevels son estados operativos (del 0 al 6) que determinan qué servicios y recursos del sistema están disponibles para el usuario. Cada level denota una configuración específica del sistema, desde el apagado y reinicio hasta el arranque de interfaces gráficas o modos de mantenimiento, lo que mejora el rendimiento, la disponibilidad de recursos y la seguridad del sistema.

d.

Runlevels

0: apaga el sistema completamente, detiene los procesos activos para apagar el equipo de forma segura.

1: permite una sesión de un único usuario para realizar tareas de mantenimiento y recuperación del sistema.

2: similar al nivel 1 pero permite múltiples usuarios, aunque sin funciones de red o compartición de datos vía NFS.

3: ofrece capacidades completas de red, pero sin entorno gráfico, recomendado para servidores.

4: solía estar en desuso, puede personalizarse para un inicio con servicios específicos.

5: como el nivel 3 pero incluye entorno gráfico, es el típico para escritorios.

6: reinicia el sistema, apagándolo y volviéndolo a encender.

La definición del runlevel de inicio se encuentra en el archivo /etc/inittab, el cual contiene la configuración que el comando init lee al arrancar para saber qué procesos y en qué niveles de ejecución debe iniciar.

Originalmente muchas de las distribuciones de Linux respetaron los runlevels del estilo SysVinit, sin embargo con la llegada del systemd como sistema de inicialización, el concepto de runlevels ha sido reemplazado por el de “targets” o estados de servicio.

e. El archivo /etc/inittab es un archivo de configuración para el programa init, que define el nivel de ejecución predeterminado, qué procesos se inician, y cuáles se supervisan y reinician si se terminan. La información es almacenada con un formato de texto con una estructura de cuatro campos: ID, nivel de ejecución, acción y comando.

f. Para cambiar del runlevel X al runlevel Y, se debe ejecutar el comando init <Y>. Sin embargo, este cambio no es permanente por sí solo, ya que la permanencia se determina editando el archivo /etc/inittab.

g. Los Scripts RC tienen la finalidad de almacenar información de configuración y recursos para un programa o sistema, como iconos, textos, o directivas de compilación, que luego se incorporan al ejecutable final. Se ubican en directorios de inicio de usuario y de sistema, como /etc/rc.d.

   Cuando un sistema GNU/Linux arranca o se detine, se ejecutan scripts determinados mediante un sistema de gestión de servicios (systemd o SysVinit) que utiliza un orden definido de scripts y dependencias para asegurar la correcta inicialización o apagado del sistema y sus servicios.

14.

a. SystemD es un gestor de sistemas y servicios para sistemas operativos Linux, que actúa como primer proceso (PID 1) en el arranque, iniciando y gestionando el resto del sistema y sus demonios. Está diseñado para reemplazar al antiguo sistema init, ofreciendo un arranque más rápido y flexible, permitiendo la ejecución paralela de procesos, gestionando la configuración de red, los puntos de montaje y el registro del sistema.

b. En SystemD, una Unit es un objeto estandarizado que representa y gestiona un recurso del sistema, como un servicio, un dispositivo, un montaje de filesystem, un socket o un temporizador.

c. El comando systemctl en Linux sirve para examinar y controlar el sistema de inicio y el administrador de servicios SystemD. Permite realizar tareas como iniciar, detener, reiniciar y comprobar el estado de los servicios del sistema, además de habilitarlos o deshabilitarlos para que arranquen automáticamente al iniciar el sistema.

d. En SystemD, un target es una unidad especial que funciona como punto de sincronización o un estado del sistema, agrupando otras unidades (como servicios) y definiendo un conjunto específico de configuraciones y servicios que deben iniciarse o deternerse. Los targers sustituyen al concepto de runlevels de sistemas anteriores como SysVinit.

e. Al ejecutar el comando pstree se puede observar la jerarquía de los procesos del sistema en un formato de árbol, lo que permite visualizar las relaciones padre-hijo entre los procesos y entender cómo se inician y cómo están organizados.

15.

a. En Linux, la relación relacionada con los usuarios se guarda principalmente en /etc/passwd, que contiene datos de la cuenta como nombre de usuario, ID, directorio personal y Shell. Las contraseñas cifradas se encuentran en el archivo /etc/shadow, mientras que la información de los grupos se almacena en /etc/group. Los directorios personales de cada usuario, donde se guardan sus documentos y configuraciones, están ubicados en /home/<username>.

b. UID significa User Identifier y es el número único que identifica a cada usuario en el sistema, mientras que GID significa Group Identifier y es un número único para identificar a los grupos de usuarios.

   No pueden coexistir dos UIDs iguales en un sistema GNU/Linux, ya que caad usuario debe tener un UID único que permita al sistema operativo identificarlo y asignar permisos de manera efectiva, lo cual es fundamental para el funcionamiento seguro y la gestión de los recursos del sistema.

c. El usuario root es el usuario administrador, con permisos ilimitados para realizar cualquier tarea y gestionar todo el sistema. No puede haber más de un usuario con el perfil de root ya que es una cuenta única con el UID 0.

16.

a. En sistemas Linux, los permisos de archivos se definen para tres categorías de usuarios (propietario, grupo y otros) y para tres tipos de operaciones (lectura, escritura y ejecución).

   Se utilizan las letras: **p**, **g** y **o** para las categorías de usuarios y **r, w** y **x** para los tipos de operaciones.

b. -

c. La notación octal en chmod representa los permisos de archivo y directorio en un formato numérico donde cada dígito corresponde a un conjunto de permisos para cada usuario, en el siguiente orden: propietario, grupo y otros.

   Cada permiso se asocia con un valor numérico: lectura (4), escritura (2) y ejecución (1). Sumando estos valores se obtiene el dígito que define los permisos para cada tipo de usuario.
   Por ejemplo: 755 significa lectura+escritura+ejecución para el propietario y lectura+ejecución para el grupo y otros.

d. Sí, bajo ciertas configuraciones o errores de permisos, un usuario podría acceder a un archivo sin tener permisos explícitos. Un usuario podría saltarse los permisos si el archivo está dentro de un directorio al que sí tiene acceso de ejecución, si se utiliza el superusuario “root”, si se explota un error de configuración o si se abusa de permisos en un directorio para modificar los permisos de archivos internos.

e. Un “full path name” especifica la ubicación de un archivo o directorio comenzando desde el directorio raíz del sistema, proporcionando la ruta completa (/home/usuario/documentos/informe.pdf). En cambio, un “relative path name” indica la ubicación de un archivo o directorio en relación con el directorio de trabajo actual, sin necesidad de indicar el inicio desde la raíz. Para acceder al archivo imagen.jpg que está dentro de /home/usuario/imagenes/, la ruta relativa sería: imagenes/imagen.jpg. Si queremos acceder a un archivo en el directorio superior, como /home/usuario/scripts/, y estamos en /home/usuario/documentos/, la ruta relativa sería ../scripts/mi_script.sh.

f. El comando pwd imprime el directorio en el cual se encuentra trabajando el usuario en Linux.

   Sí, para acceder al directorio personal sin escribir la ruta completa en Linux, se puede usar el comando cd sin argumentos o el comando cd ~. La tilde ~ es un atajo que representa tu directorio personal. Para acceder a otros directorios de manera similar, se puede usar la función alias, la cual permite asignar nombres cortos a rutas largas.

g. -

17. Procesos

a. Un proceso Foreground (primer plano) acapara la ventana o terminal desde la que se ejecuta, impidiendo que el usuario realice otras tareas hasta que termine. En cambio, un proceso Background (segundo plano) se ejecuta de manera independiente, liberando la terminal para que el usuario pueda seguir trabajando o ejecutar otros comandos sin interrupciones.

b. Para ejecutar un proceso en Background en Linux, se debe añadir un ampersand (&) al final del comando.

   Para mover un proceso a Background se utiliza el comando bg, mientar que para moverlo a Foreground se utiliza el comando fg.

c. La barra vertical | (pipe) tiene como finalidad conectar y redirigir la salida de un programa o función como entrada para otro, creando así una cadena de procesamiento de datos o comandos. Se utiliza para encadenar la ejecución de programas en la línea de comandos, transformar y formatear datos en interfaces de usuario de frameworks como Angular y pasar la salida de un análisis de datos a otro en lenguajes como R.

d. -

18. **Otros comandos**

a. Empaquetar archivos hace referencia a agrupar varios archivos y directorios en un único archivo contenedor, sin necesariamente reducir su tamaño. La herramienta más común para esta tarea es tar.

b. Se puede observar que la suma de los tamaños individuales de los archivos y el tamaño del archivo empaquetado son iguales.

c. Para comprimir tres archivos en uno solo en Linux, se puede utilizar el comando *zip nombre_archivo_comprimido.zip archivo1 archivo2 archivo3.*

   Alternativamente, también se puede crear un archivo tar.gz usando el comando *tar -czvf nombre_archivo_comprimido.tar.gz archivo1 archivo2 archivo3.*

d. Se respondió en la anterior.

e.

- tar: agrupa varios archivos y directorios en un único archivo contenedor, sin reducir su tamaño (empaquetar).
- grep: busca líneas que contengan un patrón específico de texto dentro de archivos o en la salida de otros comandos.
- gzip: agrupa varios archivos y directorios en un único archivo contenedor y además los comprime.
- zgrep: busca expresiones regulares dentro de archivos comprimidos u otros tipos de archivos sin necesidad de descomprimirlos manualmente primero.
- wc: cuenta el número de líneas, palabras, caracteres o bytes en archivos.

19. **Indicar la acción de los siguientes comandos**

a. Imprime los directorios en formato de lista extendida pero no los muestra por terminal, sino que crea y carga la información en un archivo de texto “prueba”.

b. Imprime los procesos que se están ejecutando actualmente en el sistema en un archivo de texto “PRUEBA”.

c. Otorga al usuario propietario permisos de lectura, escritura y ejecución, solo permisos de ejecución al grupo y ningún permiso a los demás sobre el archivo prueba.

d. No puede ejecutarse el comando porque el usuario propietario no es root y por lo tanto no tiene privilegios para cambiar el ownership de un directorio, por más que sea su propietario.

e. Otorga permisos de lectura, escritura y ejecución para el usuario propietario, el grupo y todos los demás usuarios sobre el archivo PRUEBA.

f. No puede ejecutarse el comando porque los permisos del directorio /etc/passwd pueden ser modificados solo por el usuario root.

g. No puede ejecutarse el comando porque para visualizar o modificar el directorio /etc/passwd es necesario contar con privilegios de usuario root, para lo cual debe utilizarse sudo.

h. Elimina el archivo de texto PRUEBA que contenía la información de los procesos que se estaban ejecutando al momento de ser creado.

i. No puede ejecutarse el comando porque el archivo contiene información sensible sobre las contraseñas y por lo tanto para abrir su manual debe contarse con permisos de usuario root.

j. El comando se utiliza para listar todos los archivos del sistema que poseen la extensión .conf, los cuales son archivos de configuración.

k. El comando intenta cambiar el directorio donde se encuentra el home del usuario root pero no puede ejecutarse correctamente porque son necesarios privilegios de usuario root (sudo).

l. El comando no puede ejecutarse porque es necesario contar con privilegios de usuario root para acceder al directorio /root.

m. Elimina todos los archivos y directorios dentro del directorio donde se está trabajando de forma permanente.

n. Cambia el directorio de trabajo a /etc.

o. No se puede ejecutar el comando porque el * hace que se copien todos los archivos y directorios en /home pero la flag -R necesita que se determine un directorio en específico y por lo tanto tiene conflicto con *.

p. Programa el apagado del sistema para dentro de un minuto.

20. **Indicar el comando necesario para llevar a cabo las siguientes acciones**

a. kill 23
b. systemctl poweroff
c. find /usr -name *.config
d. ps > /home/ valentina /procesos
e. chmod 751 /home/ valentina /archivo
f. chmod 650 /home/ valentina /archivo2
g. rm /tmp/*
h. sudo chown isocso /opt/isodata
i. pwd >> /home/valentina/donde

21. **Indique los comandos necesarios**

a. sudo su -
b. sudo useradd vwiehl y sudo passwd vwiehl
c. se modifican los archivos /etc/group, /etc/shadow y /etc/passwd y se crea el directorio /home/vwiehl
d. mkdir /temp/miCursada
e. cp -r /var/log /tmp/miCursada
f. sudo chown vwiehl:users miCursada
g. sudo chmod -R 723 miCursada

h. su vwiehl
i. hostname
j. ps
k. who
l. wall [mensaje] para escribir a todos los usuarios.
   write nombre_usuario tty_usuario para escribir a un usuario específico, se presiona enter, se introduce el mensaje y se cierra con ctrl+D.
m. shutdown

22. **Indique los comandos necesarios**

a. mkdir 202728 && cd 202728
b. vi LEAME.txt
c. chmod 017 LEAME.txt
d. ls -l /etc > leame, se puede crear porque Linux es case sensitive y por lo tanto LEAME es distinto de leame
e. para buscar un solo archivo por nombre dentro del filesystem utilizaría el comando locate nombre_archivo, ya que consulta una base de datos indexada del filesystem, por lo que es muy rápido.

   para buscar varios archivos con características similares utilizaría el comando find [directorio base] -name [patrón] ya que si bien es más lento que el comando locate, es ideal para búsquedas complejas (por tamaño, permisos, propietario)

f. find / -name *.so > ejecicioF

23. **Indique qué acciones realizan los siguientes comandos**

a. crea un directorio llamado iso
b. accede al directorio iso e imprime la información de los procesos que se encuentran en ejecución en un archivo llamado f0
c. lista el contenido del directorio iso en un archivo llamado f1
d. accede al directorio raíz
e. imprime en la terminal la ruta del directorio personal.
f. Imprime el contenido del directorio personal en formato de lista detallada en un archivo llamado ls dentro del directorio iso.
g. accede al directorio personal y crea el directorio f2.
h. Imprime en consola la información del directorio f2 en formato de lista detallada.

i. Otorga permisos de escritura y ejecución al usuario propietario, de lectura al grupo y de ejecución a los demás usuarios.
j. Crea un archivo llamado dir
k. Accede al directorio f2
l. Accede al directorio iso que se encuentra dentro del directorio personal.
m. Imprime el directorio actual de trabajo dentro de un archivo llamado f3.
n. Imprime la cantidad de procesos activos que contienen el texto “ps” dentro del archivo f3 que se encuentra en el directorio f2, dentro del directorio personal del usuario, sin sobrescribir la información en caso de que ya exista.
o. Otorga permisos de lectura, escritura y ejecución al usuario propietario y ningún tipo de permisos ni al grupo ni a los demás usuarios sobre el directorio f2 que se ubica dentro del directorio personal y luego accede al directorio personal.
p. No puede ejecutarse debido a que no funciona con directorios completos (/etc/passwd) sino con basenames, que sería el último componente de un path (en este caso passwd).
q. Sucede lo mismo que en el caso anterior.

23.2. **Completar los comandos r y s**

r.

24. **Cree la estructura que se indica e indiqué que comandos son necesarios**

a. mv ./dir1/f3 ~
b. cp ./dir2/f4 ./dir1/dir1
c. cp ./dir2/f4 ./dir1/dir1/f7
d. mkdir copia; cp -r ./dir1/* ./copia
e. mv f1 archivo; ls -l archive
f. chmod 617 archivo
g. mv f3 f3.exe; mv ./dir1/dir1/f4 ./dir1/dir1/f4.exe
h. chmod 023 f3.exe ./dir1/dir1/f4.exe

25. **Indique los comandos necesarios**

a. mkdir /tmp/logs
b. cp -r /var/log/* /tmp/logs
c. tar -cvf misLogs.tar logs (parado sobre el directorio tmp)
d. tar -cvzf misLogs.tar.gz logs (parado sobre el directorio tmp)

e. cp misLogs.tar misLogs.tar.gz ~ (parado sobre temp)
f. rm -r logs (parado sobre temp)
g. mkdir -p misLogsEmpaquetados && tar -xf misLogs.tar -C misLogsEmpaquetados
   mkdir -p misLogsComprimidos && tar -xzf misLogs.tar.gz -C misLogsComprimidos