

Introducción a los Sistemas Operativos

Práctica 2

# Objetivo

El objetivo de esta práctica es que el alumno comprenda los aspectos principales acerca de la estructura del sistema Operativo GNU/Linux en lo que respecta a procesos, usuarios, filesystems, permisos, etc.

1. Editor de textos:
   1. Nombre al menos 3 editores de texto que puede utilizar desde la línea de comandos.
   2. ¿En qué se diferencia un editor de texto de los comandos cat, more o less? Enumere los modos de operación que posee el editor de textos vi.
   3. Nombre los comandos más comunes que se le pueden enviar al editor de textos vi.
2. Nombre al menos 3 editores de texto que puede utilizar desde la línea de comandos

Vim: Un editor de texto en modo texto ampliamente utilizado en sistemas Unix y Linux.

Nano: Un editor de texto simple y amigable para usuarios nuevos en la línea de comandos.

Emacs: Otro editor de texto altamente personalizable y potente.

1. ¿En qué se diferencia un editor de texto de los comandos cat, more o less? Enumere los modos de operación que posee el editor de textos vi.

Un editor de texto y los comandos cat, more y less tienen propósitos y funcionalidades diferentes en relación con la manipulación y visualización de archivos de texto en sistemas Unix y Linux.

**CAT**: concatena archivos y los muestra por la salida estándar

**More**: permite visualizar el contenido de un archivo página por página. Puedes avanzar página a página presionando la barra espaciadora y salir del visor presionando q. Es útil para visualizar archivos mas largos de manera más cómoda.

**Less**: similar a more, pero con funcionalidades adicionales, como retroceder páginas, buscar texto y navegar por el contenido de manera mas versátil. less es especialmente útil para examinar archivos largos y realizar búsquedas con ellos.

VIM posee 3 modos de ejecución:

-Modo Insert (Ins o i)

-Modo visual (v)

-Modo de Ordenes o Normal (Esc)

1. Nombre los comandos más comunes que se le pueden enviar al editor de textos vi.

**w**: escribir cambios

**q** o **q**!:salir del editor

**dd**: cortar

**y**: copiar al portapapeles

**p**: pegar desde el portapapeles

**u**: deshacer

**/frase**: buscar “frase” dentro del archivo

1. Proceso de Arranque *SystemV*(https://github.com/systeminit/si):
   1. Enumere los pasos del proceso de inicio de un sistema GNU/Linux, desde que se prende la PC hasta que se logra obtener el login en el sistema.
   2. Proceso **INIT**. ¿Quién lo ejecuta? ¿Cuál es su objetivo?
   3. RunLevels. ¿Qué son? ¿Cuál es su objetivo?
   4. ¿A qué hace referencia cada nivel de ejecución según el estándar? ¿Dónde se define qué Runlevel ejecutar al iniciar el sistema operativo? ¿Todas las distribuciones respetan estos estándares?
   5. Archivo /etc/inittab. ¿Cuál es su finalidad? ¿Qué tipo de información se almacena en el? ¿Cuál es la estructura de la información que en él se almacena?
   6. Suponga que se encuentra en el runlevel <X>. Indique qué comando(s) ejecutaría para cambiar al runlevel <Y>. ¿Este cambio es permanente? ¿Por qué?
   7. Scripts RC. ¿Cuál es su finalidad? ¿Dónde se almacenan? Cuando un sistema GNU/Linux arranca o se detiene se ejecutan scripts, indique cómo determina qué script ejecutar ante cada acción. ¿Existe un orden para llamarlos? Justifique.
2. Enumere los pasos del proceso de inicio de un sistema GNU/Linux, desde que se prende la PC hasta que se logra obtener el login en el sistema.
3. Se empieza a ejecutar el código del BIOS
4. El BIOS lee el sector de arranque (MBR)
5. El BIOS lee el sector de arranque (MBR)
6. Se carga el gestor de arranque (MBC)
7. El bootloader carga el kernel y el initrd
8. Se monta el initrd como sistema de archivos raíz y se inicializan componentes esenciales (ej: scheduler)
9. El kernel ejecuta el proceso init y se desmonta el initrd
10. Se lee el /etc/inittab
11. Se ejecutan los scripts apuntados por el runlevel 1
12. El final del runlevel 1 le indica que vaya al runlevel por defecto
13. Se ejecutan los scripts apuntados por el runlevel por defecto
14. El sistema está listo para usarse
15. Proceso **INIT**. ¿Quién lo ejecuta? ¿Cuál es su objetivo?

Su función es cargar todos los subprocesos necesarios para el correcto funcionamiento del SO

El proceso init posee el PID 1 y se encuentra en /sbin/init

En SysV se lo configura a través del archivo /etc/inittab

No tiene padre y es el padre de todos los procesos (pstree)

Es el encargado de montar los filesystems de hacer disponible los demás dispositivos

El proceso init es el padre de todos los procesos (todos lo demás procesos son subprocesos de init). Es el proceso inicial con PDI 1. Este proceso va a leer el archivo /etc/inittab (en etc están todos los archivos que son de configuración). Es el encargado de montar los sistemas de archivos(filesystems).

1. RunLevels. ¿Qué son? ¿Cuál es su objetivo?

* Es el modo en que arranca Linux (3 en Red hat, 2 en Debian)
* El proceso de arranque lo dividimos en niveles,
* Cada uno es responsable de levantar(iniciar) o bajar (parar) una serie de servicios

Se encuentran definidos en /etc/inittab.

1. ¿A qué hace referencia cada nivel de ejecución según el estándar? ¿Dónde se define qué Runlevel ejecutar al iniciar el sistema operativo? ¿Todas las distribuciones respetan estos estándares?

Existen 7, y permiten iniciar un conjunto de procesos al

arranque o apagado del sistema

Segun el estandar:

0: halt (parada)

1: single user mode (monousuario)

2: multiuser, without NFS (modo multiusuario sin soporte de red)

3: full multiuser mode console (modo multiusuario completo por consola)

4: no se utiliza

5: X11 (modo multiusuario completo con login grafico basado en X)

6: reboot

La definición de qué runlevel ejecutar al iniciar el sistema operativo se especifica en el archivo /etc/inittab en sistemas que utilizan el sistema de inicio SysV. En sistemas que utilizan systemd, la configuración del runlevel se gestiona de manera diferente, generalmente a través del comando systemctl. En muchos sistemas modernos, especialmente en distribuciones populares como Ubuntu, CentOS y Debian, se ha adoptado systemd como el sistema de inicio por defecto, lo que ha llevado a cambios en la forma en que se gestionan los runlevels.

Es importante destacar que no todas las distribuciones de Linux respetan estrictamente el estándar de runlevels. Algunas distribuciones pueden tener configuraciones y convenciones diferentes para el inicio y la gestión de servicios. Por lo tanto, las prácticas pueden variar entre diferentes distribuciones, y es importante consultar la documentación específica de la distribución para comprender cómo se manejan los runlevels y el inicio del sistema en esa distribución en particular.

1. Archivo /etc/inittab. ¿Cuál es su finalidad? ¿Qué tipo de información se almacena en el? ¿Cuál es la estructura de la información que en él se almacena?

El archivo /etc/inittab es un archivo de configuración que se utiliza en sistemas Unix y sistemas Unix-like, como Linux, para definir el comportamiento del sistema durante el proceso de inicio y el cambio de niveles de ejecución (runlevels). Su finalidad principal es configurar cómo se inicia y se gestiona el sistema, incluyendo la definición de runlevels, la ejecución de scripts y la asignación de procesos al inicio.

id:nivelesEjecucion:acción:proceso

* Id: identifica la entrada en inittab( 1 a 4 caracteres)
* NivelesEjecucion: el/los niveles de ejecución en los que se realiza la acción
* Acción: describe la acción a realizar
  + wait:inicia cuando entra al runlevel e init espera a que termine
  + initdefault
  + ctrlaltdel: se ejecutará cuando init reciba la señal SIGINT
  + off,respawn,once,sysinit,boot,bootwait,powerwait,etc
* Proceso: el proceso exacto que será ejecutado

1. Suponga que se encuentra en el runlevel <X>. Indique qué comando(s) ejecutaría para cambiar al runlevel <Y>. ¿Este cambio es permanente? ¿Por qué?

Cambiar de runlevel en ejecución:

Existe una utilidad para línea de comandos que permite cambiar de un nivel de ejecución a otro. Esta es la herramienta init. Para cambiar de nivel de ejecución sólo hay que ejecutar init seguido del número del runlevel. Por ejemplo:

init 0: Cambia al runlevel 0 (se apaga el sistema, equivalente al comando halt).

init 2: Cambia al runlevel 2.

init 6: Cambia al runlevel 6 (reinicia el sistema, equivalente al comando reboot).

Estos cambios no son permanentes.

Modificar el runlevel por defecto:

Por defecto, el sistema suele arrancar en el nivel de ejecución 5 (modo gráfico). Si se quisiera modificar este comportamiento, habría que editar el fichero /etc/inittab.

Más concretamente, habría que modificar en el fichero /etc/inittab la línea

id:5:initdefault:

donde el número 5 indica que el nivel de ejecución por defecto es el 5. Este número es el que hay que modificar para cambiar el nivel de ejecución en el que arranca el sistema por defecto.

Estos cambios si son permanentes

1. Scripts RC. ¿Cuál es su finalidad? ¿Dónde se almacenan? Cuando un sistema GNU/Linux arranca o se detiene se ejecutan scripts, indique cómo determina qué script ejecutar ante cada acción. ¿Existe un orden para llamarlos? Justifique

Finalidad: Los scripts RC son un conjunto de scripts que se ejecutan en el arranque o apagado del sistema para iniciar o detener servicios específicos.

Dónde se almacenan: Generalmente, se encuentran en el directorio /etc/rc.d/ o en directorios similares como /etc/init.d/.

Cómo determina qué script ejecutar: Cada runlevel tiene su propio conjunto de scripts. Por ejemplo, para el runlevel 3, los scripts podrían estar en /etc/rc.d/rc3.d/. Dentro de este directorio, los scripts que comienzan con S son para iniciar servicios y los que comienzan con K son para detener servicios.

¿Existe un orden para llamarlos?: Sí, hay un orden. Los scripts suelen tener números después de la 'S' o 'K' que determinan el orden. Por ejemplo, S01script se ejecutará antes que S02another\_script

1. *SystemD*(https://github.com/systemd/systemd):
   1. ¿Qué es *sytemd*?
   2. ¿A qué hace referencia el concepto de *Unit* en SystemD?
   3. ¿Para que sirve el comando *systemctl* en SystemD?
   4. ¿A qué hace referencia el concepto de *target* en SystemD?
   5. Ejecute el comando pstree. ¿Qué es lo que se puede observar a partir de la ejecución de este comando?
2. ¿Qué es sytemd?

*systemd* es un sistema de inicio (init system) y gestor de servicios para sistemas operativos basados en Linux. Se encarga de gestionar el arranque, la configuración y la supervisión de servicios y recursos en el sistema. *systemd* reemplaza al tradicional *System V init* y proporciona características adicionales como:

* Paralelización del arranque.
* Monitoreo de servicios y reinicio automático en caso de fallo.
* Agrupación de procesos en "cgroups" para una gestión de recursos más eficiente.
* Registro centralizado de eventos mediante *journald*.

1. ¿A qué hace referencia el concepto de *Unit* en SystemD?

Una *Unit* en *systemd* es el recurso básico que *systemd* administra. Cada *Unit* define un servicio, un punto de montaje, un dispositivo, un socket u otros tipos de recursos que *systemd* puede gestionar. Las *Units* se definen a través de archivos de configuración con la extensión .service, .mount, .socket, entre otras. Algunas de las principales clases de *Units* son:

* service: Define servicios o procesos de fondo (daemons).
* target: Agrupa varias *Units* bajo un objetivo común.
* mount: Gestiona puntos de montaje.
* socket: Define sockets para comunicación entre procesos.

1. ¿Para que sirve el comando *systemctl* en SystemD?

El comando systemctl es una herramienta utilizada para interactuar con *systemd*. Sirve para gestionar los servicios y unidades del sistema, incluyendo el arranque y parada de servicios, habilitación o deshabilitación en el inicio, y revisar el estado de las *Units*. Algunos comandos comunes son:

* systemctl start <nombre.service>: Inicia un servicio.
* systemctl stop <nombre.service>: Detiene un servicio.
* systemctl restart <nombre.service>: Reinicia un servicio.
* systemctl status <nombre.service>: Muestra el estado de un servicio.
* systemctl enable <nombre.service>: Habilita un servicio para que se inicie en el arranque.
* systemctl disable <nombre.service>: Deshabilita un servicio

1. ¿A qué hace referencia el concepto de *target* en SystemD?

Un *target* en *systemd* es una unidad que agrupa otras *Units* bajo un objetivo común para definir estados del sistema. Los *targets* suelen representar etapas del proceso de arranque o apagado, y permiten gestionar de manera más eficiente qué servicios o recursos deben estar disponibles en un momento dado. Ejemplos de *targets* incluyen:

* multi-user.target: Similar al nivel de ejecución 3 de *System V*, proporciona un entorno multiusuario con servicios de red.
* graphical.target: Proporciona un entorno gráfico, similar al nivel de ejecución 5 de *System V*.
* rescue.target: Proporciona un entorno de rescate para recuperación de fallos, similar al nivel de ejecución 1.

1. Ejecute el comando pstree. ¿Qué es lo que se puede observar a partir de la ejecución de este comando?

El comando pstree muestra los procesos activos en el sistema en forma de árbol jerárquico. Este árbol refleja la relación de paternidad entre los procesos, donde cada proceso es hijo de otro. En sistemas gestionados por *systemd*, se puede observar que el proceso principal de *systemd* (PID 1) es el proceso raíz, y todos los demás procesos del sistema son hijos directos o indirectos de este proceso. pstree permite visualizar cómo los servicios y procesos están organizados y cómo interactúan dentro del sistema

1. Usuarios:
   1. ¿Qué archivos son utilizados en un sistema GNU/Linux para guardar la información de los usuarios?
   2. ¿A qué hacen referencia las siglas *UID* y *GID*? ¿Pueden coexistir UIDs iguales en un sistema GNU/Linux? Justifique.
   3. ¿Qué es el usuario root? ¿Puede existir más de un usuario con este perfil en GNU/Linux? ¿Cuál es la *UID* del *root*?.
   4. Agregue un nuevo usuario llamado iso2017 a su instalación de GNU/Linux, especifique que su home sea creada en /home/iso\_2017, y hágalo miembro del grupo catedra (si no existe, deberá crearlo). Luego, sin iniciar sesión como este usuario cree un archivo en su home personal que le pertenezca. Luego de todo esto, borre el usuario y verifique que no queden registros de él en los archivos de información de los usuarios y grupos.
   5. Investigue la funcionalidad y parámetros de los siguientes comandos:

useradd ó adduser

usermod

userdel

su

groupadd

who

groupdel

passwd

1. ¿Qué archivos son utilizados en un sistema GNU/Linux para guardar la información de los usuarios?

En un sistema GNU/Linux, los archivos utilizados ap´ra guardar la información de los usuarios son:

/etc/password: Almacena detalles sobre cada usuario, como el nombre de usuario, ID de usuario (UID) ,directorio principal y la Shell predeterminada. No alamacena contraseñas en texto claro (las contraseñas solían almacenarse aquí en sistemas muy antiguos, pero ya no es el caso por razones de seguridad).

/etc/shadow: Guarda las contraseñas de los usuarios en formato cifrado, junto con información sobre el envejecimiento de la contraseña.

/etc/group: Almacena información sobre los grupos del sistema y sus miembros

Estos archivos son fundamentales para la gestión de usuarios y grupos en un sistema GNU/Linux.

1. ¿A qué hacen referencia las siglas UID y GID? ¿Pueden coexistir UIDs iguales en un sistema GNU/Linux? Justifique.

UID (User Identifier): Es un número único que se asigna a cada usuario en el sistema y es utilizado para identificar al usuario dentro del sistema operativo. El UID es una forma de representar a los usuarios de manera que el sistema pueda diferenciarlos sin depender de nombres de usuario, que podrían ser ambiguos.

GID (Group Identifier): Es un número único que se asigna a cada grupo en el sistema. Al igual que el UID, el GID se utiliza para identificar grupos de manera única en el sistema.

Respecto a si pueden coexistir UIDs iguales en un sistema GNU/Linux: No, no pueden. Cada UID debe ser único para cada usuario. La razón es que el sistema operativo usa el UID (y no el nombre de usuario) para gestionar los permisos, el acceso a recursos, y otras operaciones. Si tuvieras dos usuarios con el mismo UID, el sistema no podría diferenciarlos y esto llevaría a ambigüedades y problemas potenciales en la administración de permisos y recursos.

1. ¿Qué es el usuario root? ¿Puede existir más de un usuario con este perfil en GNU/Linux? ¿Cuál es la *UID* del *root?*

Usuario root: Es el superusuario o administrador en un sistema GNU/Linux. Tiene permisos totales y puede realizar cualquier acción en el sistema, desde instalar software hasta eliminar archivos críticos.

No puede, hay un solo usuario "root" como tal, pero puedes conceder permisos de superusuario a otros usuarios, permitiéndoles realizar tareas con privilegios de root mediante comandos como "sudo".

 La UID del usuario root siempre es 0.

Entonces, el usuario root es el administrador supremo en Linux, y tener acceso a este usuario significa tener control total sobre el sistema.

d) **useradd / adduser:**

Uso: Agrega usuarios.

Ejemplo: useradd -m juan (crea un usuario llamado Juan con su directorio home).

**usermod:**

Uso: Modifica usuarios existentes.

Ejemplo: usermod -d /home/nuevojuan juan (cambia el directorio home de Juan).

**userdel:**

Uso: Elimina usuarios.

Ejemplo: userdel -r juan (elimina a Juan y su directorio home).

**su:**

Uso: Cambia al usuario que estás usando.

Ejemplo: su juan (cambia a Juan).

**groupadd:**

Uso: Crea grupos.

Ejemplo: groupadd amigos (crea un grupo llamado amigos).

**who:**

Uso: Muestra quién está en el sistema.

Ejemplo: who (verás una lista de usuarios conectados).

**groupdel:**

Uso: Elimina grupos.

Ejemplo: groupdel amigos (elimina el grupo amigos).

**passwd:**

Uso: Cambia contraseñas.

Ejemplo: passwd juan (cambia la contraseña de Juan)

1. FileSystem:
   1. ¿Cómo son definidos los permisos sobre archivos en un sistema GNU/Linux?
   2. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con los permisos en GNU/Linux: chmod chown chgrp.
   3. Al utilizar el comando chmod generalmente se utiliza una notación octal asociada para definir permisos. ¿Qué significa esto? ¿A qué hace referencia cada valor?
   4. ¿Existe la posibilidad de que algún usuario del sistema pueda acceder a determinado archivo para el cual no posee permisos? Nombrelo, y realice las pruebas correspondientes.
   5. Explique los conceptos de “full path name” y “relative path name”. De ejemplos claros de cada uno de ellos.
   6. ¿Con qué comando puede determinar en qué directorio se encuentra actualmente? ¿Existe alguna forma de ingresar a su directorio personal sin necesidad de escribir todo el path completo? ¿Podría utilizar la misma idea para acceder a otros directorios? ¿Cómo? Explique con un ejemplo.
   7. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con el uso del FileSystem:

cd umount mkdir

du

rmdir

df

mount

ln

ls

pwd

cp

mv

1. ¿Cómo son definidos los permisos sobre archivos en un sistema GNU/Linux?

Los permisos sobre archivos en un sistema GNU/Linux están definidos mediante un sistema de control de acceso discrecional (DAC). Cada archivo y directorio tiene tres conjuntos de permisos: para el propietario (u), el grupo (g) y otros (o). Cada conjunto puede tener permisos de lectura (r), escritura (w) y ejecución (x), lo que se representa como una serie de caracteres, como "rw-r--r--".

1. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con los permisos en GNU/Linux: chmod chown chgrp.

**chmod**:

Función: Cambiar permisos de archivos o directorios.

Ejemplo: “chmod 755 archivo” establece permisos de lectura, escritura y ejecución para el propietario, y permisos de lectura y ejecución para el grupo y otros.

**chown**:

Función: Cambiar propietario de un archivo o directorio.

Ejemplo: “chown nuevo\_usuario archivo” cambia el propietario del archivo al "nuevo\_usuario".

**chgrp**:

Función: Cambiar grupo de un archivo o directorio.

Ejemplo: “chgrp nuevo\_grupo archivo” cambia el grupo del archivo al "nuevo\_grupo”

1. Al utilizar el comando chmod generalmente se utiliza una notación octal asociada para definir permisos. ¿Qué significa esto? ¿A qué hace referencia cada valor?

La notación octal en chmod utiliza números para representar permisos: 4 para lectura (r), 2 para escritura (w), 1 para ejecución (x). El valor octal total define los permisos para el propietario, grupo y otros en ese orden. Por ejemplo, 755 significa rwx para el propietario, rx para el grupo y otros.

1. ¿Existe la posibilidad de que algún usuario del sistema pueda acceder a determinado archivo para el cual no posee permisos? Nombrelo, y realice las pruebas correspondientes.

Sí, existe esa posibilidad, el usuario root o superusuario.

El usuario root es el administrador del sistema en un sistema GNU/Linux. Tiene poderes supremos y puede acceder a cualquier archivo o directorio, independientemente de los permisos establecidos. Esto es porque el usuario root tiene privilegios completos sobre el sistema y puede superar cualquier restricción de permisos

1. Explique los conceptos de “full path name” y “relative path name”. De ejemplos claros de cada uno de ellos.

Full Path Name (Ruta absoluta):

Se refiere al camino completo para localizar un archivo o directorio en el sistema. Empieza desde el directorio raíz, representado por /, y sigue a través de todos los subdirectorios hasta llegar al archivo o carpeta deseado.

Ejemplo de Full Path Name:

/home/usuario/documentos/archivo.txt

En este ejemplo, se parte del directorio raíz (/), se sigue al subdirectorio home, luego a usuario, luego a documentos, y finalmente al archivo archivo.txt.

Relative Path Name (Ruta relativa):

Es el camino a un archivo o directorio en relación al directorio actual en el que te encuentras. No comienza desde el directorio raíz.

Ejemplo de Relative Path Name:

Supongamos que estás actualmente en el directorio /home/usuario/. Si quieres referirte al archivo.txt en el directorio documentos, simplemente usarías:

documentos/archivo.txt

Aquí, no necesitas empezar desde el directorio raíz porque ya estás en /home/usuario/

1. ¿Con qué comando puede determinar en qué directorio se encuentra actualmente? ¿Existe alguna forma de ingresar a su directorio personal sin necesidad de escribir todo el path completo? ¿Podría utilizar la misma idea para acceder a otros directorios? ¿Cómo? Explique con un ejemplo.

El comando que te dice en qué directorio estás actualmente es pwd (print working directory).

¿Existe alguna forma de ingresar a su directorio personal sin necesidad de escribir todo el path completo?

Sí, puedes usar el comando cd sin argumentos o cd ~ para volver a tu directorio personal (home).

¿Podría utilizar la misma idea para acceder a otros directorios? ¿Cómo?

¡Sí! Puedes usar rutas relativas para moverte entre directorios. Si estás en un directorio y quieres moverte a un subdirectorio dentro de él, solo escribes cd nombre\_del\_subdirectorio.

Ejemplo:

Supongamos que estás en /home/usuario y hay un directorio llamado documentos dentro. Puedes hacer:

$ cd documentos

Además, puedes usar .. para ir al directorio padre.

Ejemplo:

Si estás en /home/usuario/documentos y quieres volver a /home/usuario, puedes hacer:

$ cd ..

1. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con el uso del FileSystem

**cd:**

Uso: Cambia el directorio actual.

Ejemplo: cd documentos

"Ingresa al directorio llamado documentos".

**umount:**

Uso: Desmonta sistemas de archivos.

Ejemplo: umount /dev/sda1

"Desmonta el dispositivo en /dev/sda1".

**mkdir:**

Uso: Crea un directorio.

Ejemplo: mkdir nuevo\_directorio

"Crea un directorio llamado 'nuevo\_directorio'".

**du:**

Uso: Muestra el uso de disco de un directorio.

Ejemplo: du -sh documentos

"Muestra el uso total de espacio del directorio documentos en un formato legible".

**rmdir:**

Uso: Elimina un directorio vacío.

Ejemplo: rmdir directorio\_vacio

"Elimina el directorio vacío llamado 'directorio\_vacio'".

**df:**

Uso: Muestra el espacio usado y disponible en el sistema de archivos.

Ejemplo: df -h

"Muestra el espacio en un formato legible".

**mount:**

Uso: Monta sistemas de archivos.

Ejemplo: mount /dev/sda1 /mnt/mi\_disco

"Monta el dispositivo /dev/sda1 en el directorio /mnt/mi\_disco".

**ln:**

Uso: Crea enlaces entre archivos.

Ejemplo: ln archivo\_original enlace\_archivo

"Crea un enlace llamado 'enlace\_archivo' que apunta a 'archivo\_original'".

**ls:**

Uso: Lista los archivos en un directorio.

Ejemplo: ls -l

"Lista los archivos en formato largo, mostrando permisos y propietarios".

**pwd:**

Uso: Muestra el directorio actual.

Ejemplo: pwd

Resultado: "/home/usuario".

**cp:**

Uso: Copia archivos o directorios.

Ejemplo: cp fuente.txt destino.txt

"Copia 'fuente.txt' a 'destino.txt'".

**mv:**

Uso: Mueve o renombra archivos o directorios.

Ejemplo: mv viejo.txt nuevo.txt

"Renombra el archivo 'viejo.txt' a 'nuevo.txt'"

1. Procesos:
   1. ¿Qué es un proceso? ¿A que hacen referencia las siglas *PID* y *PPID*? ¿Todos los procesos tienen estos atributos en GNU/Linux? Justifique. Indique qué otros atributos tiene un proceso.
   2. Indique qué comandos se podrían utilizar para ver qué procesos están en ejecución en un sistema GNU/Linux.
   3. ¿Qué significa que un proceso se está ejecutando en Background? ¿Y en Foreground?
   4. ¿Cómo puedo hacer para ejecutar un proceso en Background? ¿Como puedo hacer para pasar un proceso de background a foreground y viceversa?
   5. Pipe ( **|** ). ¿Cuál es su finalidad? Cite ejemplos de su utilización.
   6. Redirección. ¿Qué tipo de redirecciones existen? ¿Cuál es su finalidad? Cite ejemplos de utilización.
   7. Comando *kill*. ¿Cuál es su funcionalidad? Cite ejemplos.
   8. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con el manejo de procesos en GNU/Linux. Además, compárelos entre ellos:

ps

kill

pstree

killall

top

nice

1. ¿Qué es un proceso? ¿A que hacen referencia las siglas *PID* y *PPID*? ¿Todos los procesos tienen estos atributos en GNU/Linux? Justifique. Indique qué otros atributos tiene un proceso.

Proceso:

Un proceso es un programa en ejecución. Es una instancia activa de un programa que incluye el contador de programa, registros del procesador y variables.

PID y PPID:

PID (Process ID): Es un número único que identifica un proceso en el sistema.

Ejemplo: Imagina tener varios programas abiertos y cada uno tiene un "número de identificación" para diferenciarlos.

PPID (Parent Process ID): Es el ID del proceso padre que inició el proceso actual.

Ejemplo: Si abres un programa (proceso padre) y este abre otro programa (proceso hijo), el hijo tendrá su propio PID pero su PPID será el PID del padre.

¿Todos los procesos tienen PID y PPID en GNU/Linux?

Sí, todos los procesos en GNU/Linux tienen un PID y un PPID. El sistema necesita estas identificaciones para manejar y organizar los procesos correctamente.

Otros atributos de un proceso:

Estado: Define si un proceso está en ejecución, dormido, detenido, etc.

Prioridad: Define el orden en el que el proceso recibirá recursos de CPU.

Usuario: Identifica qué usuario inició el proceso. (en caso de ambientes multiusuario, desde que terminal y quien lo ejecutó)

Tiempo de CPU: Cuánto tiempo ha consumido el proceso en la CPU.

Grupo: si hay estructura de grupos, define grupo que lo disparó

1. Indique qué comandos se podrían utilizar para ver qué procesos están en ejecución en un sistema GNU/Linux

Comandos para visualizar procesos en ejecución en GNU/Linux:

**ps:**

Uso: Muestra los procesos en ejecución para una sesión.

Ejemplo: ps aux (muestra todos los procesos del sistema).

**top:**

Uso: Muestra los procesos en tiempo real y permite la interacción.

Ejemplo: Simplemente escribe top.

**htop:**

Uso: Similar a top pero con una interfaz más amigable y colorida.

Ejemplo: Primero debes instalarlo en algunas distribuciones, luego solo escribe htop.

**pstree:**

Uso: Muestra los procesos en forma de árbol.

Ejemplo: pstree -p (muestra el árbol de procesos con IDs).

**pgrep:**

Uso: Busca procesos basados en un nombre.

Ejemplo: pgrep firefox (muestra los IDs de todos los procesos relacionados con Firefox)

1. ¿Qué significa que un proceso se está ejecutando en Background? ¿Y en Foreground?

**Background (Fondo):**

Cuando un proceso se ejecuta en "background", está corriendo, pero no está ocupando la terminal activa. Puedes seguir usando la terminal para otros comandos mientras el proceso en background sigue su ejecución.

Ejemplo: Imagina que pones a reproducir música y luego minimizas el reproductor para hacer otras cosas en tu computadora. La música (proceso) sigue sonando en el background.

**Foreground (Primer Plano):**

Un proceso en "foreground" es el que está ocupando la terminal activa. No puedes usar esa terminal para otra cosa hasta que el proceso en foreground termine o se mueva al background.

Ejemplo: Si abres un programa desde la terminal y este programa muestra su interfaz en dicha terminal, entonces ese programa está en foreground y tienes que esperar a que termine o enviarlo al background para usar esa terminal nuevamente.

1. ¿Cómo puedo hacer para ejecutar un proceso en Background? ¿Como puedo hacer para pasar un proceso de background a foreground y viceversa?

Para ejecutar un proceso en segundo plano (background), simplemente añade un "&" al final del comando cuando lo ejecutes. Por ejemplo, para ejecutar el comando "mi\_programa" en background:

"mi\_programa &"

Pasar un proceso de Background a Foreground: Utiliza el comando fg seguido del número de trabajo del proceso que deseas traer al primer plano. Por ejemplo:

"fg %1"

Donde %1 se refiere al primer trabajo en background. Si tienes varios procesos en background, puedes usar jobs para listarlos y luego usar fg %n para traer el proceso deseado al foreground, donde "n" es el número de trabajo del proceso.

1. Pipe ( **|** ). ¿Cuál es su finalidad? Cite ejemplos de su utilización.

El operador “|” te permite ejecutar dos comandos de forma secuencial, donde la salida del primero se convierte automáticamente en la entrada del segundo. Esto te permite realizar dos acciones al mismo tiempo, donde el resultado del primer comando se utiliza como entrada para el segundo. Por ejemplo:

"ls | grep archivo"

Este comando lista archivos en un directorio y luego filtra esa lista para mostrar solo los archivos que contienen la palabra "archivo"

1. Redirección. ¿Qué tipo de redirecciones existen? ¿Cuál es su finalidad? Cite ejemplos de utilización.

Tipos de Redirección:

**Redirección de Salida (>)**: Envía la salida estándar de un comando a un archivo.

Ejemplo: ls > lista.txt

**Redirección de Entrada (<)**: Toma la entrada estándar de un comando desde un archivo.

Ejemplo: sort < archivo.txt

**Redirección de Salida y Error (2> o &>)**: Envía la salida estándar y los mensajes de error a un archivo.

Ejemplo: comando\_inexistente 2> errores.txt

**Redirección de Salida al Final (>>)**: Agrega la salida al final de un archivo sin borrar su contenido.

Ejemplo: echo "Nueva línea" >> archivo.txt

**Redirección de Entrada y Salida (<>)**: Abre un archivo para lectura y escritura simultánea.

Ejemplo: cat <> archivo.txt

Estos métodos permiten controlar cómo los comandos interactúan con archivos y flujos de datos en Linux.

1. Comando kill. ¿Cuál es su funcionalidad? Cite ejemplos

El comando kill se utiliza para enviar señales a procesos en ejecución en sistemas Linux. Su función principal es detener o interactuar con procesos de manera controlada.

Ejemplos:

Detener un proceso específico por su ID de proceso (PID):

“kill 1234”

Enviar la señal SIGTERM (15) para solicitar una terminación suave de un proceso (el 5678):

“kill -15 5678”

Enviar la señal SIGKILL (9) para forzar la terminación inmediata de un proceso (el 7890):

“kill -9 7890”

El comando kill es útil para administrar procesos y puede usarse para detener o interactuar con ellos según sea necesario.

1. Investigue la funcionalidad y parámetros de los siguientes comandos relacionados con el manejo de procesos en GNU/Linux. Además, compárelos entre ellos.

**ps:**

Función: Muestra información sobre los procesos en ejecución.

Ejemplo: “ps aux” muestra una lista detallada de todos los procesos en el sistema.

**kill:**

Función: Envía señales a procesos para controlar su comportamiento.

Ejemplo: “kill -15 1234” envía la señal SIGTERM al proceso con PID 1234 para una terminación suave.

**pstree:**

Función: Muestra los procesos en forma de árbol, mostrando relaciones entre ellos.

Ejemplo: “pstree -p” muestra un árbol de procesos con sus PID.

**killall:**

Función: Envía señales a procesos según su nombre en lugar de su PID.

Ejemplo: “killall -TERM nombre\_proceso” envía la señal SIGTERM a todos los procesos con el nombre "nombre\_proceso".

**top:**

Función: Proporciona una vista en tiempo real de los procesos y recursos del sistema.

Ejemplo: Simplemente ejecuta top para ver una lista en tiempo real de procesos y recursos.

**nice:**

Función: Ajusta la prioridad de ejecución de un proceso.

Ejemplo: “nice -n 10 comando” ejecuta "comando" con una prioridad más baja (mayor número) de 10.

Estos comandos son esenciales para administrar y supervisar procesos en sistemas GNU/Linux

1. Otros comandos de Linux (Indique funcionalidad y parámetros):
   1. ¿A qué hace referencia el concepto de empaquetar archivos en GNU/Linux?
   2. Seleccione 4 archivos dentro de algún directorio al que tenga permiso y sume el tamaño de cada uno de estos archivos. Cree un archivo empaquetado conteniendo estos 4 archivos y compare los tamaños de los mismos. ¿Qué característica nota?
   3. ¿Qué acciones debe llevar a cabo para comprimir 4 archivos en uno solo? Indique la secuencia de comandos ejecutados.
   4. ¿Pueden comprimirse un conjunto de archivos utilizando un único comando?
   5. Investigue la funcionalidad de los siguientes comandos:

tar

grep

gzip zgrep

wc

1. ¿A qué hace referencia el concepto de empaquetar archivos en GNU/Linux?

El concepto de empaquetar archivos en GNU/Linux se refiere a la acción de agrupar uno o varios archivos y directorios en un solo archivo comprimido. Esto se hace principalmente para ahorrar espacio en disco y facilitar la distribución y transferencia de archivos. Los archivos empaquetados suelen comprimirse para reducir su tamaño, lo que facilita su manejo y transferencia.

1. Seleccione 4 archivos dentro de algún directorio al que tenga permiso y sume el tamaño de cada uno de estos archivos. Cree un archivo empaquetado conteniendo estos 4 archivos y compare los tamaños de los mismos. ¿Qué característica nota?

Crear archivos de texto vacíos:

"touch archivo1.txt archivo2.txt archivo3.txt archivo4.txt"

Verificar el tamaño de los archivos individuales y sumarlos:

"du -ch archivo1.txt archivo2.txt archivo3.txt archivo4.txt"

Esto mostrará el tamaño de cada archivo individualmente y la suma total de sus tamaños.

Empaquetar los archivos en un archivo tar (el c significa crear, la v significa "verbose" (detallado), muestra informacion detallada sobre el proceso en la salida estandar mientas se crea el archivo tar. la f significa "file", indica que se va a especificar el nombre del archivo tar despues de esta opcion):

"tar -cvf archivos\_empaquetados.tar archivo1.txt archivo2.txt archivo3.txt archivo4.txt"

Comprimir el archivo tar con gzip (este paso se puede ahorrar agregando una z a -cvf del paso anterior, osea -czvf):

"gzip archivos\_empaquetados.tar"

Verificar el tamaño del archivo comprimido:

"du -h archivos\_empaquetados.tar.gz"

Esto mostrará el tamaño del archivo empaquetado y comprimido.

Generalmente el archivo empaquetado y comprimido es más pequeño que la suma de los tamaños de los archivos individuales debido a la compresión.

1. ¿Qué acciones debe llevar a cabo para comprimir 4 archivos en uno solo? Indique la secuencia de comandos ejecutados.

Para comprimir cuatro archivos en uno sólo, puedes utilizar el comando tar para empaquetarlos en un archivo tar y luego usar gzip para comprimir ese archivo tar. Aquí está la secuencia de comandos que debes ejecutar:

# Empaquetar los archivos en un archivo tar

"tar -cvf archivos\_empaquetados.tar archivo1.txt archivo2.txt archivo3.txt archivo4.txt"

# Comprimir el archivo tar con gzip

"gzip archivos\_empaquetados.tar"

Estos comandos crearán un archivo tar llamado "archivos\_empaquetados.tar" que contiene los cuatro archivos especificados y luego lo comprimirán con gzip, resultando en un archivo comprimido llamado "archivos\_empaquetados.tar.gz" que contiene los cuatro archivos originales.

1. ¿Pueden comprimirse un conjunto de archivos utilizando un único comando?

Sí, es posible comprimir un conjunto de archivos utilizando un solo comando. En Linux, puedes usar el comando tar junto con la opción -z para comprimir los archivos mientras los empaqueta en un archivo tar en un solo paso. Aquí está el comando:

"tar -czvf archivos\_empaquetados.tar.gz archivo1.txt archivo2.txt archivo3.txt archivo4.txt"

En este comando:

-c indica que se está creando un nuevo archivo tar.

-z indica que se utilizará compresión gzip.

-v muestra información detallada sobre el proceso.

-f especifica el nombre del archivo de salida.

Luego, se enumeran los nombres de los archivos que deseas empaquetar y comprimir.

Con este comando, los archivos se empaquetarán y comprimirán en un solo paso, y se creará un archivo llamado "archivos\_empaquetados.tar.gz". Esto puede simplificar el proceso si deseas realizar ambas operaciones al mismo tiempo

1. Investigue la funcionalidad de los siguientes comandos:

**tar (Archiver):**

Función: Comprime y archiva uno o varios archivos o directorios en un solo archivo.

Ejemplo: Para crear un archivo tar de un directorio llamado "mi\_directorio": tar -cvf mi\_directorio.tar mi\_directorio/

**grep (Global Regular Expression Print):**

Función: Busca patrones de texto en archivos o en la salida de otros comandos.

Ejemplo: Para buscar la palabra "buscar" en un archivo llamado "texto.txt": grep "buscar" texto.txt

**gzip (GNU Zip):**

Función: Comprime archivos usando el algoritmo de compresión gzip.

Ejemplo: Para comprimir un archivo llamado "archivo.txt" y crear "archivo.txt.gz": gzip archivo.txt

**zgrep:**

Función: Busca patrones de texto en archivos comprimidos con gzip.

Ejemplo: Para buscar la palabra "buscar" en un archivo comprimido llamado "archivo.txt.gz":

zgrep "buscar" archivo.txt.gz

**wc (Word Count):**

Función: Cuenta palabras, líneas y bytes en archivos o en la entrada estándar.

Ejemplo: Para contar las líneas en un archivo llamado "texto.txt": wc -l texto.txt

Estos comandos son herramientas útiles para trabajar con archivos, buscar información específica y realizar tareas de compresión y conteo de contenido en sistemas Linux.

1. Indique qué acción realiza cada uno de los comandos indicados a continuación considerando su orden. Suponga que se ejecutan desde un usuario que no es root ni pertenece al grupo de root. (Asuma que se encuentra posicionado en el directorio de trabajo del usuario con el que se logueó). En caso de no poder ejecutarse el comando, indique la razón:

ls −l > prueba

ps > PRUEBA

chmod 710 prueba

chown root : root PRUEBA

chmod 777 PRUEBA

chmod 700 /etc/passwd

passwd root

rm PRUEBA

man /etc/shadow

find / −name ∗. conf

usermod root −d /home/newroot −L

cd /root

rm ∗

cd /etc

cp ∗ /home −R

shutdown

"ls -l > prueba": Este comando redirige la salida del comando ls -l al archivo llamado "prueba". Listará el contenido del directorio actual en un formato largo y guardará esa lista en el archivo "prueba".

"ps > PRUEBA": Este comando redirige la salida del comando ps al archivo llamado "PRUEBA". Mostrará una lista de los procesos actuales y guardará esa lista en el archivo "PRUEBA".

"chmod 710 prueba": Cambia los permisos del archivo "prueba" para que el propietario tenga permisos de lectura, escritura y ejecución, mientras que el grupo solo tiene permiso de ejecución. Los demás no tienen permisos.

"chown root:root PRUEBA": Cambia el propietario y el grupo del archivo "PRUEBA" a "root". (es una operacion no permitida, se necesitan privilegios de administrador).

"chmod 777 PRUEBA: Cambia los permisos del archivo "PRUEBA" para que todos tengan permisos de lectura, escritura y ejecución. Esto no es una buena práctica de seguridad. No me deja, esto lo estoy haciendo con mi user normal valentino por eso

"chmod 700 /etc/passwd": Cambia los permisos del archivo /etc/passwd para que solo el propietario tenga acceso de lectura y escritura, mientras que los demás no tienen acceso.(es una operacion no permitida, solo se puede hacer con privilegios de administrados) Lo mismo

"passwd root": Intenta cambiar la contraseña del usuario "root". Esto solo funcionará si el usuario actual tiene los permisos adecuados para realizar esta acción. De lo contrario, se requerirán privilegios de root.

"rm PRUEBA": Elimina el archivo "PRUEBA" del directorio actual.

"man /etc/shadow": Intenta ver el manual del archivo /etc/shadow. Esto solo funcionará si el usuario actual tiene los permisos adecuados para acceder a ese archivo. De lo contrario, se requerirán privilegios de root.

"find / -name \*.conf": Busca archivos con extensión ".conf" en todo el sistema de archivos, comenzando desde la raíz ("/"). Esto puede llevar tiempo y generará resultados en función de los permisos de lectura del usuario.(algunos me deja y otros me aparecen con acceso denegado).

"usermod root -d /home/newroot -L": Intenta modificar el usuario "root" para cambiar su directorio de inicio a "/home/newroot" y bloquearlo. Esto solo funcionará si el usuario actual tiene los permisos adecuados y si el sistema permite realizar esta acción en el usuario "root".

"cd /root": Cambia el directorio de trabajo actual al directorio "/root". Esto solo funcionará si el usuario actual tiene permisos para acceder a ese directorio. Denegadisimo

"rm \*": Elimina todos los archivos en el directorio actual. Esto puede ser peligroso si se ejecuta en un directorio equivocado. (me deja) Hice una instantánea por las dudas

"cd /etc": Cambia el directorio de trabajo actual al directorio "/etc". Esto solo funcionará si el usuario actual tiene permisos para acceder a ese directorio. (me deja)

"cp \* /home -R": Copia todos los archivos y directorios del directorio actual al directorio "/home" de manera recursiva. Esto puede ser peligroso si se ejecuta en un directorio equivocado.

"shutdown": Intenta apagar o reiniciar el sistema. Esto solo funcionará si el usuario actual tiene los permisos adecuados para realizar esta acción, generalmente se requieren privilegios de root.

1. Indique qué comando sería necesario ejecutar para realizar cada una de las siguientes acciones:
   1. Terminar el proceso con *PID* 23.
   2. Terminar el proceso llamado *init* o *systemd*. ¿Qué resultados obtuvo?
   3. Buscar todos los archivos de usuarios en los que su nombre contiene la cadena “.conf” (d) Guardar una lista de procesos en ejecución el archivo **/home/<su nombre de usuario>/procesos**
   4. Cambiar los permisos del archivo **/home/<su nombre de usuario>/xxxx** a:

Usuario: Lectura, escritura, ejecución

Grupo: Lectura, ejecución

Otros: ejecución

* 1. Cambiar los permisos del archivo **/home/<su nombre de usuario>/yyyy** a:

Usuario: Lectura, escritura.

Grupo: Lectura, ejecución

Otros: Ninguno

* 1. Borrar todos los archivos del directorio **/tmp**
  2. Cambiar el propietario del archivo **/opt/isodata** al usuario **iso2010**
  3. Guardar en el archivo **/home/<su nombre de usuario>/donde** el directorio donde me encuentro en este momento, en caso de que el archivo exista no se debe eliminar su contenido anterior.

1. Terminar el proceso con PID 23.

Para terminar el proceso con PID 23, puedes usar el siguiente comando:

"kill 23"

Sin embargo, es importante mencionar que en algunos casos, el proceso puede no responder a la señal estándar SIGTERM enviada por kill. En tales situaciones, puedes agregar la opción -9 al comando kill para enviar la señal SIGKILL, lo que forzará la terminación del proceso de la siguiente manera:

"kill -9 23"

La señal SIGKILL es más contundente y garantiza que el proceso se detenga, pero también puede dejar recursos no liberados. Por lo tanto, se recomienda usarla con precaución y sólo cuando sea necesario cuando otros métodos de terminación no funcionan.

1. Terminar el proceso llamado init. ¿Qué resultados obtuvo?

Terminar el proceso llamado "init" puede tener graves consecuencias en un sistema Linux, ya que "init" es el proceso principal (PID 1) y es esencial para el funcionamiento del sistema. Cerrar este proceso generalmente provocará un reinicio o un colapso del sistema.

Por lo tanto, la respuesta a esta pregunta es que no se recomienda terminar el proceso "init" en un sistema en funcionamiento, a menos que sea necesario como parte de un proceso de reinicio o apagado del sistema. Terminar "init" podría resultar en la pérdida de datos y la inestabilidad del sistema. Si necesitas reiniciar el sistema, es mejor utilizar comandos apropiados para reiniciar o apagar el sistema en lugar de intentar terminar "init" directamente.

Si estás utilizando un entorno de virtualización, como una máquina virtual, es posible que el sistema operativo anfitrión haya proporcionado una capa de aislamiento que evitó que el comando afectará al sistema anfitrión.

1. Buscar todos los archivos de usuarios en los que su nombre contiene la cadena “.conf”

Para buscar todos los archivos de usuarios en los que su nombre contiene la cadena ".conf", puedes utilizar el comando find de la siguiente manera:

"find /home -type f -name "\*.conf" "

Este comando buscará en el directorio /home y sus subdirectorios (-type f indica que se buscan archivos) y mostrará una lista de todos los archivos cuyos nombres contienen la cadena ".conf". Los resultados incluirán archivos de usuarios que se encuentren en el directorio /home.

1. Guardar una lista de procesos en ejecución el archivo /home/<su nombre de usuario>/procesos

Para guardar una lista de procesos en ejecución en un archivo llamado "procesos" en el directorio "/home/tu\_usuario", puedes usar el siguiente comando:

"ps aux > /home/tu\_usuario/procesos"

"ps aux": Este comando muestra una lista de procesos en ejecución en el sistema y envía esta lista a la salida estándar, que normalmente es la pantalla.

">": Este símbolo se llama "redirección de salida". Cuando se utiliza después de un comando como ps aux, indica que la salida del comando debe ser redirigida hacia un archivo en lugar de mostrarse en la pantalla.

"/home/tu\_usuario/procesos": Es la ruta completa del archivo donde se almacenará la salida del comando. En este caso, el archivo se llamará "procesos" y se ubicará en el directorio "/home/tu\_usuario".

1. Cambiar los permisos del archivo /home//xxxx a:

* Usuario: Lectura, escritura, ejecución
* Grupo: Lectura, ejecución
* Otros: ejecución

Para cambiar los permisos del archivo "/home/tu\_usuario/xxxx" de acuerdo a las especificaciones dadas:

"chmod 751 /home/tu\_usuario/xxxx"

Este comando establecerá los permisos como se especificó anteriormente:

Usuario (propietario): Lectura, escritura, ejecución (7).

Grupo: Lectura, ejecución (5).

Otros: Ejecución (1).

Así, los permisos del archivo "/home/tu\_usuario/xxxx" se habrán cambiado según las especificaciones dadas. Asegúrate de reemplazar "tu\_usuario" y "xxxx" con tus valores reales.

los números para los permisos en Linux son:

Lectura = 4

Escritura = 2

Ejecución = 1

Puedes sumar estos números para asignar permisos. Por ejemplo, 7 significa lectura (4) + escritura (2) + ejecución (1).

1. Cambiar los permisos del archivo /home//yyyy a: Usuario: Lectura, escritura.

Grupo: Lectura, ejecución

Otros: Ninguno

“chmod 650 /home/tu\_usuario/yyyy”

1. Borrar todos los archivos del directorio /tmp

Para borrar todos los archivos del directorio "/tmp", puedes utilizar el comando rm de la siguiente manera:

“rm /tmp/\*”

Este comando eliminará todos los archivos en el directorio "/tmp".

1. Cambiar el propietario del archivo /opt/isodata al usuario iso2010

Para cambiar el propietario del archivo "/opt/isodata" al usuario "iso2010", puedes utilizar el comando chown de la siguiente manera:

"sudo chown iso2010: /opt/isodata"

Este comando cambiará el propietario del archivo "/opt/isodata" al usuario "iso2010".

1. Guardar en el archivo /home//donde el directorio donde me encuentro en este momento, en caso de que el archivo exista no se debe eliminar su contenido anterior.

Para guardar el directorio actual en un archivo llamado "donde" en "/home/tu\_usuario/" sin eliminar su contenido anterior, puedes utilizar el comando pwd para obtener la ruta del directorio actual y luego redirigir esa información al archivo. Aquí tienes el comando:

"pwd >> /home/tu\_usuario/donde"

Este comando tomará la ruta actual y la agregará al archivo "donde" en "/home/tu\_usuario/" sin borrar el contenido existente en ese archivo ya que uso >>.

1. Indique qué comando sería necesario ejecutar para realizar cada una de las siguientes acciones:
   1. Ingrese al sistema como usuario “root”
   2. Cree un usuario. Elija como nombre, por convención, la primer letra de su nombre seguida de su apellido. Asígnele una contraseña de acceso.
   3. ¿Qué archivos fueron modificados luego de crear el usuario y qué directorios se crearon?
   4. Crear un directorio en /tmp llamado cursada2017
   5. Copiar todos los archivos de /var/log al directorio antes creado.
   6. Para el directorio antes creado (y los archivos y subdirectorios contenidos en él) cambiar el propietario y grupo al usuario creado y grupo users.
   7. Agregue permiso total al dueño, de escritura al grupo y escritura y ejecución a todos los demás usuarios para todos los archivos dentro de un directorio en forma recursiva. (h) Acceda a otra terminal virtual para loguearse con el usuario antes creado.
   8. Una vez logueado con el usuario antes creado, averigüe cuál es el nombre de su terminal.
   9. Verifique la cantidad de procesos activos que hay en el sistema.
   10. Verifiqué la cantidad de usuarios conectados al sistema.
   11. Vuelva a la terminal del usuario root, y envíele un mensaje al usuario anteriormente creado, avisándole que el sistema va a ser apagado.
   12. Apague el sistema.
2. Ingrese al sistema como usuario “root”

Para ingresar al sistema como usuario "root", puedes utilizar el comando su (switch user) seguido de "root" como usuario:

"su root"

1. Cree un usuario. Elija como nombre, por convención, la primer letra de su nombre seguida de su apellido. Asígnele una contraseña de acceso.

Para crear un nuevo usuario con el nombre "NApellido" y asignarle una contraseña de acceso, puedes usar los siguientes comandos:

"sudo useradd -m -c "Nombre Completo" -s /bin/bash NApellido"

"sudo passwd NApellido"

Estos comandos hacen lo siguiente:

sudo useradd -m -c "Nombre Completo" -s /bin/bash NApellido: Crea un nuevo usuario con el nombre de usuario "NApellido". Los argumentos -m crean el directorio de inicio del usuario, -c establece un comentario con tu nombre completo y -s establece la shell (en este caso, /bin/bash) para el usuario.

sudo passwd NApellido: Establece una contraseña para el nuevo usuario "NApellido". Te pedirá que ingreses la contraseña dos veces para confirmarla.

1. ¿Qué archivos fueron modificados luego de crear el usuario y qué directorios se crearon?

Al crear un usuario "nombre\_usuario" en Linux, se realizan los siguientes cambios:

-Se crea un directorio de inicio en /home/nombre\_usuario.

-Se modifica el archivo /etc/passwd con información del usuario.

-Se actualiza el archivo /etc/shadow con la contraseña cifrada.

-Se asignan permisos al directorio de inicio para la privacidad del usuario.

Estos cambios permiten al usuario "nombre\_usuario" acceder al sistema y tener su propio espacio de trabajo en /home/nombre\_usuario

1. Crear un directorio en /tmp llamado cursada2017

Para crear un directorio llamado "cursada2017" en la ubicación /tmp, puedes usar el siguiente comando:

"sudo mkdir /tmp/cursada2017"

Este comando creará un directorio llamado "cursada2017" dentro del directorio /tmp

1. Copiar todos los archivos de /var/log al directorio antes creado

Para copiar todos los archivos de /var/log al directorio "cursada2017" que has creado en /tmp, puedes utilizar el comando cp con la opción -r para copiar recursivamente todos los archivos y directorios:

"sudo cp -r /var/log/\* /tmp/cursada2017/"

copiarlo de manera recursiva permite tambien copiar los archivos y subdirectorios de /var/log

1. Para el directorio antes creado (y los archivos y subdirectorios contenidos en él) cambiar el propietario y grupo al usuario creado y grupo users.

Para cambiar el propietario y el grupo del directorio "cursada2017" y todos los archivos y subdirectorios contenidos en él al usuario "nombre\_usuario" y al grupo "users", puedes usar el siguiente comando:

"sudo chown -R nombre\_usuario:users /tmp/cursada2017"

Este comando utiliza la opción -R para realizar el cambio de propietario y grupo de forma recursiva, lo que significa que afectará a todos los archivos y subdirectorios dentro de "/tmp/cursada2017".

Para verificar que se ha realizado correctamente el cambio de propietario y grupo del directorio "cursada2017" y su contenido, puedes utilizar el comando ls con la opción -l para listar los archivos y directorios en /tmp/cursada2017 y mostrar sus propietarios y grupos.

1. Agregue permiso total al dueño, de escritura al grupo y escritura y ejecución a todos los demás usuarios para todos los archivos dentro de un directorio en forma recursiva.

Para agregar permisos de escritura y ejecución a todos los demás usuarios, escritura al grupo y permiso total al dueño en forma recursiva a todos los archivos dentro de un directorio, puedes utilizar el comando chmod con la opción -R. Suponiendo que deseas aplicar estos permisos al directorio "/tmp/cursada2017" y su contenido, puedes ejecutar el siguiente comando:

"sudo chmod -R 723 /tmp/cursada2017"

El dueño del archivo tiene permisos:

Lectura (4)

Escritura (2)

Ejecución (1)

El grupo tiene permisos:

Escritura (2)

Otros usuarios tienen permisos:

Escritura (2)

Ejecución (1)

1. Acceda a otra terminal virtual para loguearse con el usuario antes creado.

Para acceder a otra terminal virtual y loguearse con un usuario específico en sistemas GNU/Linux, puedes seguir estos pasos:

Presiona Ctrl + Alt + F2 (o cualquier otra tecla de función, como F3, F4, etc.) para cambiar a otra terminal virtual. Esto te llevará a una pantalla de inicio de sesión en texto.

En la pantalla de inicio de sesión en texto, ingresa el nombre de usuario que deseas usar y presiona Enter.

Luego, se te solicitará ingresar la contraseña del usuario. Ingresa la contraseña y presiona Enter.

Si las credenciales son correctas, deberías haber iniciado sesión en la nueva terminal virtual con el usuario especificado.

Para cambiar entre las terminales virtuales, puedes usar las combinaciones de teclas Ctrl + Alt + F1, Ctrl + Alt + F2, etc., donde cada tecla de función representa una terminal diferente.

1. Una vez logueado con el usuario antes creado, averigüe cuál es el nombre de su terminal.

Para averiguar el nombre de la terminal en la que estás logueado, puedes utilizar el siguiente comando en la terminal:

"tty"

Simplemente ejecuta ese comando y te mostrará el nombre de la terminal en la que estás logueado. Esto suele verse como algo similar a "/dev/pts/0" o "/dev/tty1", dependiendo de la terminal virtual en la que te encuentres.

1. Verifique la cantidad de procesos activos que hay en el sistema

Para verificar la cantidad de procesos activos en el sistema, puedes utilizar el comando ps en la terminal. Hay varias formas de usar ps para obtener información sobre los procesos. Una de las opciones más comunes es la siguiente:

"ps aux | wc -l"

Este comando muestra una lista de todos los procesos activos en el sistema y luego utiliza wc -l para contar el número de líneas, lo que equivale al número de procesos activos.

Si deseas ver una lista más detallada de los procesos activos junto con su información, puedes simplemente ejecutar:

"ps aux"

1. Verifique la cantidad de usuarios conectados al sistema.

puedes utilizar el comando who en la terminal. Simplemente ejecuta el siguiente comando:

"who"

Este comando te mostrará una lista de los usuarios que están actualmente conectados al sistema, junto con información sobre su sesión, como el nombre de usuario, la terminal en la que están conectados, la fecha y hora de inicio de sesión y la dirección IP si es aplicable.

Para contar la cantidad de usuarios conectados, puedes usar el comando wc -l para contar las líneas de la salida del comando who:

"who | wc -l"

Esto te dará el número total de usuarios conectados al sistema en ese momento.

1. Vuelva a la terminal del usuario root, y envíele un mensaje al usuario anteriormente creado, avisándole que el sistema va a ser apagado,

Abre una terminal y asegúrate de que estás logueado con el usuario desde el cual deseas enviar el mensaje.

Usa el comando who o w para verificar qué usuarios están actualmente conectados al sistema y cuál es su terminal. Por ejemplo, ejecuta:

"who"

Esto te mostrará una lista de usuarios y sus terminales activas.

Una vez que identifiques el usuario al que deseas enviar el mensaje y su terminal (por ejemplo, /dev/pts/1), puedes usar el comando write seguido del nombre de usuario y la terminal para iniciar la mensajería. Por ejemplo, si deseas enviar un mensaje al usuario "nombre\_usuario" en la terminal /dev/pts/1, ejecuta:

"write nombre\_usuario /dev/pts/1"

Ten en cuenta que debes especificar el nombre de usuario y la terminal exactamente como se muestra en la lista de who. También, asegúrate de que el usuario esté realmente usando esa terminal en ese momento.

Después de ejecutar el comando anterior, tu terminal cambiará a modo de escritura y podrás escribir el mensaje. Simplemente escribe tu mensaje y presiona Enter para enviarlo.

El usuario en la terminal especificada recibirá tu mensaje y podrá responder si lo desea.

1. Apague el sistema.

Para apagar el sistema desde la terminal como usuario root, puedes utilizar el comando shutdown.

Para apagar el sistema de inmediato, puedes usar el siguiente comando:

"sudo shutdown -h -now"

Este comando le dice al sistema que se apague (opción -h) inmediatamente (opción now).

También puedes especificar un tiempo de espera antes de que se apague. Por ejemplo, para apagar el sistema en 10 minutos, puedes usar:

"sudo shutdown -h +10"

1. Indique qué comando sería necesario ejecutar para realizar cada una de las siguientes acciones:
   1. Cree un directorio cuyo nombre sea su número de legajo e ingrese a él.
   2. Cree un archivo utilizando el editor de textos vi, e introduzca su información personal: Nombre, Apellido, Número de alumno y dirección de correo electrónico. El archivo debe llamarse "LEAME".
   3. Cambie los permisos del archivo LEAME, de manera que se puedan ver reflejados los siguientes permisos:

Dueño: ningún permiso

Grupo: permiso de ejecución

Otros: todos los permisos

* 1. Vaya al directorio /etc y verifique su contenido. Cree un archivo dentro de su directorio personal cuyo nombre sea leame donde el contenido del mismo sea el listado de todos los archivos y directorios contenidos en /etc. ¿Cuál es la razón por la cuál puede crear este archivo si ya existe un archivo llamado "LEAME.en este directorio?.
  2. ¿Qué comando utilizaría y de qué manera si tuviera que localizar un archivo dentro del filesystem? ¿Y si tuviera que localizar varios archivos con características similares? Explique el concepto teórico y ejemplifique.
  3. Utilizando los conceptos aprendidos en el punto e), busque todos los archivos cuya extensión sea .so y almacene el resultado de esta búsqueda en un archivo dentro del directorio creado en a). El archivo deberá llamarse .ejercicio\_f".

1. Cree un directorio cuyo nombre sea su número de legajo e ingrese a él.

El directorio se crea con el comando “mkdir nombre\_archivo”.

Y para ingresar a él se usa el comando “cd nombre\_archivo”.

1. Cree un archivo utilizando el editor de textos vi, e introduzca su información personal: Nombre, Apellido, Número de alumno y dirección de correo electrónico. El archivo debe llamarse "LEAME"

Para esto se usa el comando “vi LEAME”. esto crea el archivo. luego, ya en la ventana del editor de texto, se presiona la i para utilizar el modo inserción y se escriben los datos. Cuando ya se escribieron los datos, se presiona esc para salir del modo inserción y se escribe :wq para guardar y salir del editor

1. Cambie los permisos del archivo LEAME, de manera que se puedan ver reflejados los siguientes permisos: Dueño: ningún permiso Grupo: permiso de ejecución Otros: todos los permisos.

Esto se hace con “chmod 017 LEAME”. el 0 indica ningún permiso para el dueño, el 1 indica permiso de ejecución para el grupo y el 7 indica permiso de lectura,escritura y ejecución para otros.

1. Vaya al directorio /etc y verifique su contenido. Cree un archivo dentro de su directorio personal cuyo nombre sea leame donde el contenido del mismo sea el listado de todos los archivos y directorios contenidos en /etc. ¿Cuál es la razón por la cuál puede crear este archivo si ya existe un archivo llamado "LEAME.en este directorio?

ls /etc > ~/leame

El sistema de archivos en Linux distingue entre mayúsculas y minúsculas. Por lo tanto, "LEAME" y "leame" son considerados como dos archivos diferentes. en vez de ~ se puede poner la ruta donde se quiere guardar el archivo. el > es para que los elementos listados sean la entrada al archivo leame.

1. ¿Qué comando utilizaría y de qué manera si tuviera que localizar un archivo dentro del filesystem? ¿Y si tuviera que localizar varios archivos con características similares? Explique el concepto teórico y ejemplifique.

Para buscar un archivo específico:

“find / -name <nombre\_del\_archivo>”

Para buscar varios archivos con características similares:

“find / -name '<patrón\_de\_búsqueda>'”

Por ejemplo, para buscar todos los archivos .txt en el sistema: "find / -name '\*.txt'"

1. Utilizando los conceptos aprendidos en el punto e), busque todos los archivos cuya extensión sea .so y almacene el resultado de esta búsqueda en un archivo dentro del directorio creado en a). El archivo deberá llamarse .ejercicio\_f".

Esto se realiza con el comando:

“find / -name '\*.so' > .ejercicio\_f”

También se puede especificar la ruta donde se quiere guardar el archivo. por ej /home/usuario/ejercicio\_f

1. Indique qué acción realiza cada uno de los comandos indicados a continuación considerando su orden. Suponga que se ejecutan desde un usuario que no es root ni pertenece al grupo de root. (Asuma que se encuentra posicionado en el directorio de trabajo del usuario con el que se logueó). En caso de no poder ejecutarse el comando indique la razón:

mkdir iso

cd ./ iso ; ps > f0

ls > f1

cd /

echo $HOME

ls −l $> $HOME/ iso / ls

cd $HOME; mkdir f2

ls −ld f2

chmod 341 f2

touch dir

cd f2

cd ~/ iso

pwd >f3 ps | grep 'ps ' | wc −l >> ../ f2/f3

chmod 700 ../ f2 ; cd . .

find . −name etc/passwd

find / −name etc/passwd

mkdir ejercicio5

1. Inicie 2 sesiones utilizando su nombre de usuario y contraseña. En una sesión vaya siguiendo paso a paso las órdenes que se encuentran escritas en el cuadro superior. En la otra sesión, cree utilizando algún editor de textos un archivo que se llame .ejercicio10\_explicacion"dentro del directorio creado en el ejercicio 9.a) y, para cada una de las órdenes que ejecute en la otra sesión, realice una breve explicación de los resultados obtenidos.
2. Complete en el cuadro superior los comandos 19 y 20, de manera tal que realicen la siguiente acción:

19: Copiar el directorio iso y todo su contenido al directorio creado en el inciso

9.a).

20: Copiar el resto de los archivos y directorios que se crearon en este ejercicio al directorio creado en el ejercicio 9.a).

1. Ejecute las órdenes 19 y 20 y comentelas en el archivo creado en el inciso a).

**mkdir iso**: Crea un directorio llamado "iso".

**cd ./iso; ps > f0**: Cambia al directorio "iso" y guarda la lista de procesos en un archivo llamado "f0".

**ls > f1**: Guarda el listado del directorio actual (iso) en un archivo llamado "f1".

**cd /**: Cambia al directorio raíz del sistema.

**echo $HOME**: Muestra el directorio home del usuario actual.

**ls -l &> $HOME/iso/ls**: Guarda el listado detallado del directorio raíz en un archivo llamado "ls" dentro del directorio "iso".

**cd $HOME; mkdir f2**: Regresa al directorio home del usuario y crea un directorio llamado "f2".

**ls -l df2**: Muestra el listado detallado del archivo o directorio "df2". Si no existe, dará error.

**chmod 341 f2**: Cambia los permisos del directorio "f2" dando al dueño escritura y ejecucion, al grupo lectura y a otros ejecucion.

**touch dir**: Crea un archivo vacío llamado "dir".

**cd f2**: Cambia al directorio "f2".

**cd ~/iso**: Cambia al directorio "iso" en el directorio home del usuario.

**pwd >f3**: Guarda la ruta del directorio actual en un archivo llamado "f3".

**ps | grep ' ps ' | wc -l >> ../f2/f3**: Cuenta cuántas veces 'ps' aparece en la lista de procesos y añade el número al archivo "f3" en el directorio "f2".

**chmod 700 ../f2; cd ..**: Cambia los permisos del directorio "f2" a lectura, escritura y ejecución solo para el dueño, y regresa al directorio superior.

**find . -name etc/passwd**: Busca el archivo "passwd" en el directorio "etc" dentro del directorio actual y sus subdirectorios.

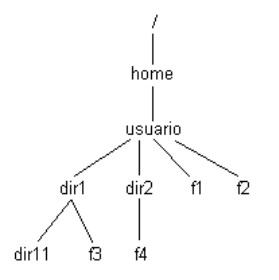
**find / -name etc/passwd**: Busca el archivo "passwd" en el directorio "etc" en todo el sistema.

**mkdir ejercicio5**: Crea un directorio llamado "ejercicio5".

b) Completando comandos 19 y 20:

19. **cp -r iso <ruta\_directorio\_ejercicio\_9.a>**: Copia el directorio "iso" y su contenido al directorio creado en el ejercicio 9.a.

20. **cp \* <ruta\_directorio\_ejercicio\_9.a>**: Copia todos los archivos y directorios (excepto "iso", que ya fue copiado) al directorio creado en el ejercicio 9.a.

1. Cree una estructura desde el directorio /home que incluya varios directorios, subdirectorios y archivos, según el esquema siguiente. Asuma que “usuario” indica cuál es su nombre de usuario. Además deberá tener en cuenta que dirX hace referencia a directorios y fX hace referencia a archivos: 
   1. Utilizando la estructura de directorios anteriormente creada, indique que comandos son necesarios para realizar las siguientes acciones:

Mueva el archivo "f3.al directorio de trabajo /home/usuario.

Copie el archivo "f4.en el directorio "dir11".

Haga los mismo que en el inciso anterior pero el archivo de destino, se debe llamar

"f7".

Cree el directorio copia dentro del directorio usuario y copie en él, el contenido de

"dir1".

Renombre el archivo "f1"por el nombre archivo y vea los permisos del mismo.

Cambie los permisos del archivo llamado archivo de manera de reflejar lo siguiente:

* + - Usuario: Permisos de lectura y escritura
    - Grupo: Permisos de ejecución
    - Otros: Todos los permisos

Renombre los archivos "f3 "f4"de manera que se llamen "f3.exe "f4.exerespectivamente.

2

2

Utilizando un único comando cambie los permisos de los dos archivos renombrados en el inciso anterior, de manera de reflejar lo siguiente:

* + - Usuario: Ningún permiso
    - Grupo: Permisos de escritura
    - Otros: Permisos de escritura y ejecución

1. Primero, se crea la estructura de directorios, subdirectorios y archivos. Asumiendo que el nombre del usuario es usuario y partiendo del directorio /home/usuario:

mkdir dir1 dir1/dir11 dir2

touch f1 f2 dir1/f3 dir2/f4

Comandos para las acciones solicitadas (suponiendo que estamos posicionados en /home/usuario):

Mover el archivo "f3" al directorio de trabajo /home/usuario: **mv dir1/f3** .

Copiar el archivo "f4" en el directorio "dir11": **cp dir2/f4 dir1/dir11/**

Copiar el archivo "f4" en el directorio "dir11" con el nombre "f7": **cp dir2/f4 dir1/dir11/f7**

Crear el directorio "copia" dentro del directorio "usuario" y copiar el contenido de "dir1" en él: **mkdir /home/usuario/copia**

**cp -r dir1/\* /home/usuario/copia/**

Renombrar el archivo "f1" a "archivo" y ver sus permisos: **mv dir1/f1 dir1/archivo**

**ls -l dir1/archivo**

Cambiar los permisos del archivo llamado "archivo": **chmod 617 dir1/archivo**

Renombrar los archivos "f3" y "f4" a "f3.exe" y "f4.exe" respectivamente: **mv f3 f3.exe**

**mv dir1/dir11/f4 dir1/dir11/f4.exe**

Cambiar permisos de los archivos "f3.exe" y "f4.exe": **chmod 023 f3.exe dir1/dir11/f4.exe**

1. Indique qué comando/s es necesario para realizar cada una de las acciones de la siguiente secuencia de pasos (considerando su orden de aparición):
   1. Cree un directorio llamado logs en el directorio /tmp.
   2. Copie todo el contenido del directorio /var/log en el directorio creado en el punto anterior.
   3. Empaquete el directorio creado en 1, el archivo resultante se debe llamar "misLogs.tar".
   4. Empaquete y comprima el directorio creado en 1, el archivo resultante se debe llamar "misLogs.tar.gz".
   5. Copie los archivos creados en 3 y 4 al directorio de trabajo de su usuario. (f) Elimine el directorio creado en 1, logs.

(g) Desempaquete los archivos creados en 3 y 4 en 2 directorios diferentes.

Comandos para las acciones solicitadas:

Cree un directorio llamado logs en el directorio /tmp: **mkdir /tmp/logs**

Copie todo el contenido del directorio /var/log en el directorio creado en el punto anterior: **sudo cp -r /var/log/\* /tmp/logs/**

(el -r es para que se haga recursivamente, ósea afecta a todos los directorios y sus respectivos subdirectorios,archivos,etc.)

Empaquete el directorio creado en 1, el archivo resultante se debe llamar "misLogs.tar":

**sudo tar -cf misLogs.tar /tmp/logs**

(-cf: Estas son dos opciones que se pasan a tar: -c: Significa "crear". Indica que se creará un nuevo archivo tar. -f: Significa "archivo". Indica que se proporcionará un nombre de archivo después de esta opción. En este caso, el nombre del archivo es "misLogs.tar".)

Empaquete y comprima el directorio creado en 1, el archivo resultante se debe llamar "misLogs.tar.gz": **sudo tar -czf misLogs.tar.gz /tmp/logs**

(-c: Indica que deseas crear un nuevo archivo de archivo. -z: Indica que deseas comprimir el archivo resultante usando gzip. -f misLogs.tar.gz: Especifica el nombre del archivo de salida como "misLogs.tar.gz". )

Copie los archivos creados en 3 y 4 al directorio de trabajo de su usuario:

**cp misLogs.tar misLogs.tar.gz /home/usuario/iso**

Elimine el directorio creado en 1, logs: **sudo rm -r /tmp/logs** (el -r es para recursividad y que elimine todo lo que hay debajo)

Desempaquete los archivos creados en 3 y 4 en 2 directorios diferentes:

**mkdir unpacked\_tar unpacked\_tar\_gz** (Este comando crea dos directorios llamados "unpacked\_tar" y "unpacked\_tar\_gz" en tu directorio de trabajo actual. Estos directorios se utilizarán para desempaquetar los archivos.)

**tar -xf misLogs.tar -C unpacked\_tar/** (Este comando desempaqueta el archivo "misLogs.tar" en el directorio "unpacked\_tar". La opción -x significa "extraer" o "desempaquetar", la opción -f especifica el archivo de origen ("misLogs.tar") y la opción -C indica el directorio de destino ("unpacked\_tar"). Después de ejecutar este comando, los archivos y directorios que estaban empaquetados en "misLogs.tar" se encontrarán en el directorio "unpacked\_tar".)

**tar -xzf misLogs.tar.gz -C unpacked\_tar\_gz/** (Este comando desempaqueta y descomprime el archivo "misLogs.tar.gz" en el directorio "unpacked\_tar\_gz". La opción -x significa "extraer", la opción -z indica que el archivo está comprimido con gzip, la opción -f especifica el archivo de origen ("misLogs.tar.gz") y la opción -C indica el directorio de destino ("unpacked\_tar\_gz"). Después de ejecutar este comando, los archivos y directorios que estaban empaquetados y comprimidos en "misLogs.tar.gz" estarán desempaquetados y descomprimidos en el directorio "unpacked\_tar\_gz".)