Introducción a Linux

Para cada pregunta, añade capturas de pantalla donde se muestre la ejecución de os comandos que se pide. Entrega la práctica en formato pdf . El archivo llámalo con el número de la práctica + tu nombre, por ejemplo: PR_01.1_Juan_García.pdf

1.- Conceptos Básicos y Variables

1. Muestra el contenido de tu variable de entorno HOME . Luego, usa cd junto con esa variable para navegar a dicho directorio y verifica con pwd que te encuentras en la ubicación correcta.

```
brayan1@bserver:~$ cd /
brayan1@bserver:/$ ls
bin                boot   dev  home  lib.usr-is-merged  lost+found  mnt  proc  run   sbin.usr-is-merged  srv       sys  usr
bin.usr-is-merged  cdrom  etc  lib   lib64              media       opt  root  sbin  snap                swap.img  tmp  var
brayan1@bserver:/$ cd /home
brayan1@bserver:/home$ ls
brayan1
brayan1@bserver:/home$ pwd
/home
brayan1@bserver:/home$ _
```

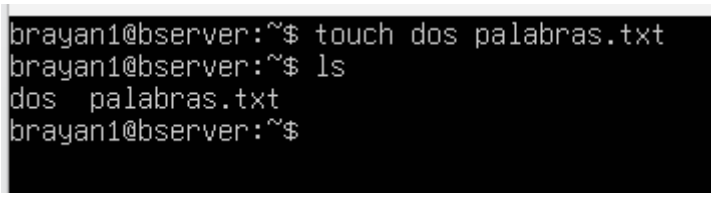
2. Ejecuta el comando whoami . Ahora, crea una variable local llamada USUARIO_ACTUAL que contenga el resultado del comando anterior y muéstrala en la terminal.

Ubuntu (Instantánea 3) [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

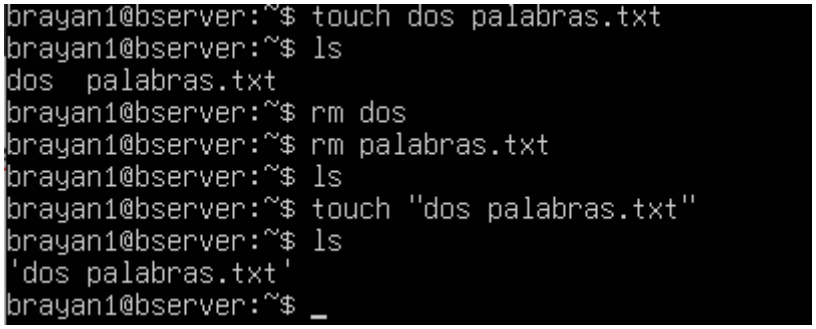
```
brayan1@bserver:~$ who
brayan1  tty1         2025-10-20 07:27
brayan1@bserver:~$ w
 08:23:47 up 58 min,  1 user,  load average: 0.06, 0.03, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU  WHAT
brayan1  tty1     -                07:27    2.00s  0.22s   ?    w
brayan1@bserver:~$ whoami
brayan1
brayan1@bserver:~$ USUARIO_ACTUAL=`whoami`
brayan1@bserver:~$ echo USUARIO_ACTUAL
USUARIO_ACTUAL
brayan1@bserver:~$ echo $USUARIO_ACTUAL
brayan1
brayan1@bserver:~$
```

3. Intenta crear un archivo llamado dos palabras.txt sin usar comillas. Observa el resultado con ls . ¿Qué ha ocurrido y por qué? Ahora, bórralo(s) y créalo correctamente.

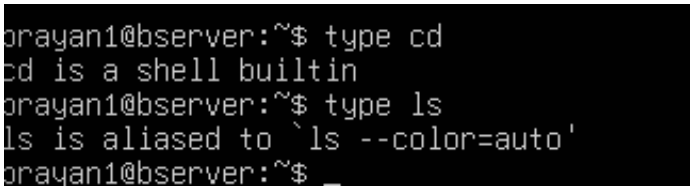


En Linux el espacio en blanco lo interpreta como un carácter por lo tanto Linux interpreta cada palabra como un argumento separado. En lugar de crear un único archivo llamado "dos palabras.txt".

Borramos y creamos con millas



4. Usa el comando type para averiguar si ls y cd son internos o externos al shell. ¿Qué diferencia práctica crees que implica esto?



cd (change directory) es un comando interno del shell Interno (Built-in). El shell ejecuta la lógica de cd directamente sin buscar ningún archivo externo.

ls (list directory contents) es un comando externo (Archivo ejecutable). El resultado indica la ruta completa (/usr/bin/ls) del archivo ejecutable que el shell debe cargar y ejecutar.

5. Muestra tu PATH actual. Crea un directorio ~/mi_bin y añádelo temporalmente al principio de tu PATH . Verifica que el cambio se ha realizado correctamente.

2.- Obtener Ayuda y Localizar Archivos

1. Abre la página del manual para el comando chmod . ¿En qué sección del manual se encuentra? ¿Qué indica ese número de sección sobre el tipo de comando?

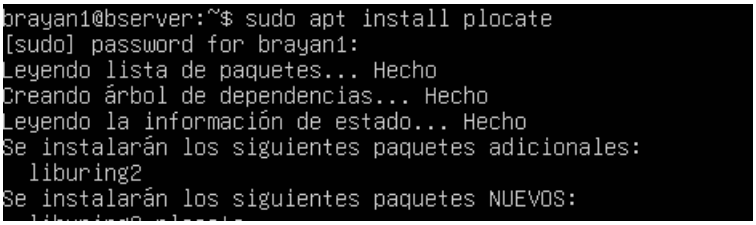
Aparece en la página 395 (pdf) en la sección 5 5 Seguridad y permisos de archivos, en el punto 5.3 Gestión de los permisos y la propiedad de los archivos

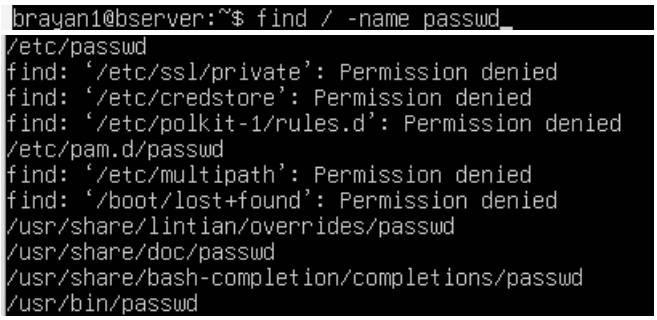
2. Usando la función de búsqueda dentro de la página del manual de ls , encuentra la opción que ordena los archivos por tamaño.

`ls -lX`

Combinar *long list* con la opción de ordenar por *file eXtension*. Esto agrupará todos los archivos que terminen con `.txt`, todos los que terminen con `.jpg` y así sucesivamente.

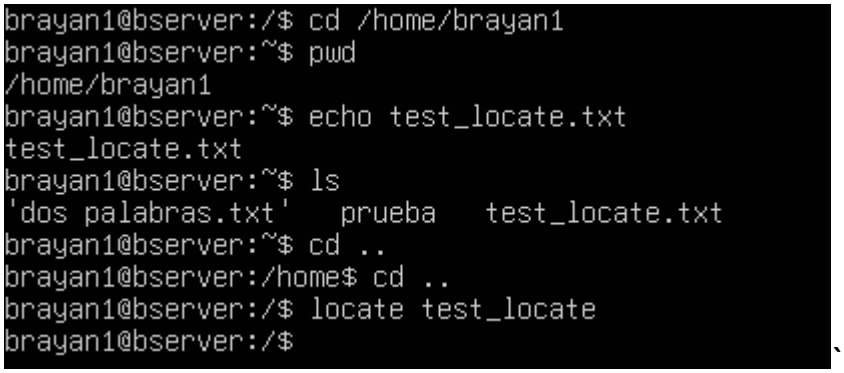
3. Imagina que has olvidado dónde se guarda el archivo de configuración de usuarios. Sabiendo que se llama passwd , usa find para buscarlo desde el directorio raíz (/). Anota la ruta completa que has encontrado.





4. Crea un archivo vacío llamado test_locate.txt en tu directorio home. Inmediatamente después, búscalo con locate . ¿Aparece en los resultados?

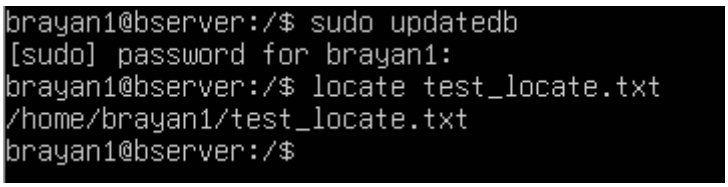
¿Por qué sí o por qué no?



No aparecen los resultados porque el comando locate utiliza una base de datos para localizar los archivos y como lo acabamos de crear no está indexado. Locate no realiza una búsqueda en tiempo real.

5. Basado en el ejercicio anterior, ¿qué comando (probablemente con sudo) necesitas ejecutar para que locate sí encuentre tu archivo? Ejecútalo y verifica que ahora sí lo encuentras.

El comando sudo updatebd



Después de forzar una actualización de la base de datos, ya aparece el archivo de texto.

3.- Navegación y Listado de Archivos

1. Navega al directorio /etc . Desde ahí, sin usar cd , lista el contenido de tu

directorio home usando una ruta con el atajo ~ .

```
brayan1@bserver:/$ cd /etc
brayan1@bserver:/etc$ pwd
/etc
brayan1@bserver:/etc$ ls ~
'dos palabras.txt'   prueba   test_locate.txt
```

2. Desde tu directorio home , navega a / y luego a var y finalmente a log

usando una sola línea de comando y rutas relativas.

```
brayan1@bserver:~$ pwd
/home/brayan1
brayan1@bserver:~$ cd ../../var/log
brayan1@bserver:/var/log$ ls
README            apport.log  bootstrap.log  cloud-init-output.log  dist-
alternatives.log  apt         btmp           cloud-init.log         dpkg.
brayan1@bserver:/var/log$ _
```

3. Lista el contenido de /etc en formato largo. En la salida, identifica el

propietario, el grupo y los permisos del archivo passwd .

```
brayan1@bserver:/$ ls -l /etc/passwd
-rw-r--r-- 1 root root 1442 Oct 20 06:53 /etc/passwd
brayan1@bserver:/$ _
```

4. Compara la salida de ls -l /etc y ls -lh /etc . ¿Qué hace la opción h y por qué es útil para las personas?

```
drwxr-xr-x 2 root root   4.0K Aug  5 17:02 rc0.d
drwxr-xr-x 2 root root   4.0K Aug  5 17:02 rc1.d
drwxr-xr-x 2 root root   4.0K Oct 20 06:53 rc2.d
drwxr-xr-x 2 root root   4.0K Oct 20 06:53 rc3.d
drwxr-xr-x 2 root root   4.0K Oct 20 06:53 rc4.d
drwxr-xr-x 2 root root   4.0K Oct 20 06:53 rc5.d
drwxr-xr-x 2 root root   4.0K Aug  5 17:02 rc6.d
drwxr-xr-x 2 root root   4.0K Aug  5 17:02 rcS.d
lrwxrwxrwx 1 root root     39 Aug  5 17:02 resolv.conf -> ../run/syste
lrwxrwxrwx 1 root root     13 Apr  8  2024 rmt -> /usr/sbin/rmt
-rw-r--r-- 1 root root    911 Oct 17  2022 rpc
drwxr-xr-x 2 root root   4.0K Aug  5 17:02 rsyslog.d
drwxr-xr-x 4 root root   4.0K Oct 20 07:15 security
drwxr-xr-x 2 root root   4.0K Aug  5 16:59 selinux
```

Nos indica el tamaño de los archivos en KB, MB o GB que podeos entender fácilmente porque los redondea.

5. Ejecuta ls -R ~ . ¿Qué hace la opción R ? ¿Por qué podría ser peligroso usarla

en el directorio raíz (/)?

Ls -R sobre ~ nos da la ruta completa del directorio (~)

- **Muestra los archivos y subdirectorios en el directorio actual.**
- **Luego, entra en cada subdirectorio y muestra su contenido.**
- **Repite este proceso para todos los subdirectorios que encuentre, sin importar la profundidad.**

Ls -R sobre el directorio raíz / genera un bucle infinito que deja la máquina sin funcionar

4.- Manipulación de Archivos y Directorios

1. Crea la estructura de directorios proyecto/src , proyecto/doc y proyecto/bin usando un único comando mkdir .

```
brayan1@bserver:~$ mkdir -p proyecto/{src,doc,bin}
brayan1@bserver:~$ ls
'dos palabras.txt'   proyecto   test_locate.txt
brayan1@bserver:~$ ls -l
total 4
-rw-rw-r-- 1 brayan1 brayan1    0 Oct 20 08:43 'dos palabras.txt'
drwxrwxr-x 5 brayan1 brayan1 4096 Oct 22 10:04  proyecto
-rw-rw-r-- 1 brayan1 brayan1    0 Oct 21 08:27  test_locate.txt
brayan1@bserver:~$ ls -R
.:
'dos palabras.txt'   proyecto   test_locate.txt

./proyecto:
bin  doc  src

./proyecto/bin:

./proyecto/doc:

./proyecto/src:
brayan1@bserver:~$ _
```

2. Crea un archivo ~/notas.txt . Muévelo a ~/proyecto/doc y, en el mismo comando, renómbralo a README.md .

```
brayan1@bserver:/$ touch ~/notas.txt
brayan1@bserver:/$ ls ~/
dos  mi_bin  notas.txt  palabras.txt  proyecto  test_locate.tx
brayan1@bserver:/$ mv ~/notas.txt ~/proyecto/doc/README.md
brayan1@bserver:/$ ls ~/proyecto/doc
notas.txt  README.md
brayan1@bserver:/$ re
```

3. Copia el archivo README.md de proyecto/doc a proyecto/bin . Luego, borra el archivo original de la carpeta doc .

```
brayan1@bserver:/$ cp ~/proyecto/doc/README.md ~/proyecto/bin
brayan1@bserver:/$ ls ~/proyecto/bin
README.md
```

```
brayan1@bserver:/$ rm ~/proyecto/doc/README.md
brayan1@bserver:/$ ls ~/proyecto/doc/
notas.txt
brayan1@bserver:/$ _
```

4. Intenta borrar el directorio proyecto con rmdir . ¿Qué error obtienes? Ahora, usa rm con la opción correcta para borrar el directorio y todo lo que contiene.

```
brayan1@bserver:/$ rmdir ~/proyecto
rmdir: failed to remove '/home/brayan1/proyecto': Directory not empty
brayan1@bserver:/$ rm ~/proyecto
rm: cannot remove '/home/brayan1/proyecto': Is a directory
brayan1@bserver:/$ rm -r ~/proyecto
brayan1@bserver:/$ ls ~/
dos  mi_bin  palabras.txt  test_locate.txt
brayan1@bserver:/$ _
```

Sale que no se puede borrar porque no esta vacia, para borrar carpteas con contenido se añade el **comando -r**

5. Navega a /etc . Usando un solo comando ls con globbing , lista todos los archivos que empiecen con la letra s y terminen con .conf .

ls /etc/s*.conf

s*.conf: La expresión de globbing que filtra los archivos deseados:

s: Coincide con la letra 's' al principio del nombre del archivo.

* (asterisco): Un comodín de globbing que coincide con cualquier cadena de cero o más caracteres.

.conf: Coincide con los archivos que terminan exactamente con esa extensión.

```
brayan1@bserver:/$ cd etc
brayan1@bserver:/etc$ ls s*.conf
sensors3.conf  sudo.conf  sudo_logsrvd.conf  sysctl.conf
brayan1@bserver:/etc$
```

5.- Archivado y Compresión

1. Crea un archivo tar llamado log_backup.tar que contenga todos los archivos del directorio /var/log . ¿Qué advertencias de “permiso denegado” aparecen y por qué?

```
brayan1@bserver:~$ tar -cvf log_bakup.tar /var/log
/var/log/installer/device-map.json
tar: /var/log/installer/subiquity-server-debug.log.1433: Cannot open: Permission denied
/var/log/installer/subiquity-client-info.log
tar: /var/log/installer/cloud-init-output.log: Cannot open: Permission denied
tar: /var/log/installer/subiquity-client-debug.log.1396: Cannot open: Permission denied
tar: Exiting with failure status due to previous errors
brayan1@bserver:~$
```

Aparece un error de permiso denegados para poder ejecutar tenemos que aplicar permisos de administrador con el comando **sudo tar -cvf log_backup.tar /var/log**

tar: El comando para crear y manipular archivos tar.

-c: Crea un nuevo archivo.

-v: Muestra el progreso detallado (verbose), lo cual es útil para ver qué archivos se están archivando y cuáles causan errores.

-f: Indica que el siguiente argumento es el nombre del archivo de destino.

2. Comprime el archivo log_backup.tar con gzip . Compara el tamaño del archivo original y el comprimido usando ls -lh .

```
brayan1@bserver:/$ cd ~/
brayan1@bserver:~$ ls
dos  log_bakup.tar  mi_bin  palabras.txt  test_locate.txt
brayan1@bserver:~$ gzip log_bakup.tar
brayan1@bserver:~$ ls
dos  log_bakup.tar.gz  mi_bin  palabras.txt  test_locate.txt
brayan1@bserver:~$ _
```

```
brayan1@bserver:~$ ls -lh
total 68M
-rw-rw-r-- 1 brayan1 brayan1    0 oct 23 12:41 dos
-rw-rw-r-- 1 brayan1 brayan1  67M oct 23 17:07 log_bakup.tar
-rw-rw-r-- 1 brayan1 brayan1 1,1M oct 23 13:37 log_bakup.tar.gz
drwxrwxr-x 2 brayan1 brayan1 4,0K oct 20 12:29 mi_bin
-rw-rw-r-- 1 brayan1 brayan1    0 oct 23 12:41 palabras.txt
-rw-rw-r-- 1 brayan1 brayan1    0 oct 23 12:42 test_locate.txt
brayan1@bserver:~$
```

3. Lista el contenido del archivo log_backup.tar.gz sin extraerlo para verificar que los archivos están dentro.

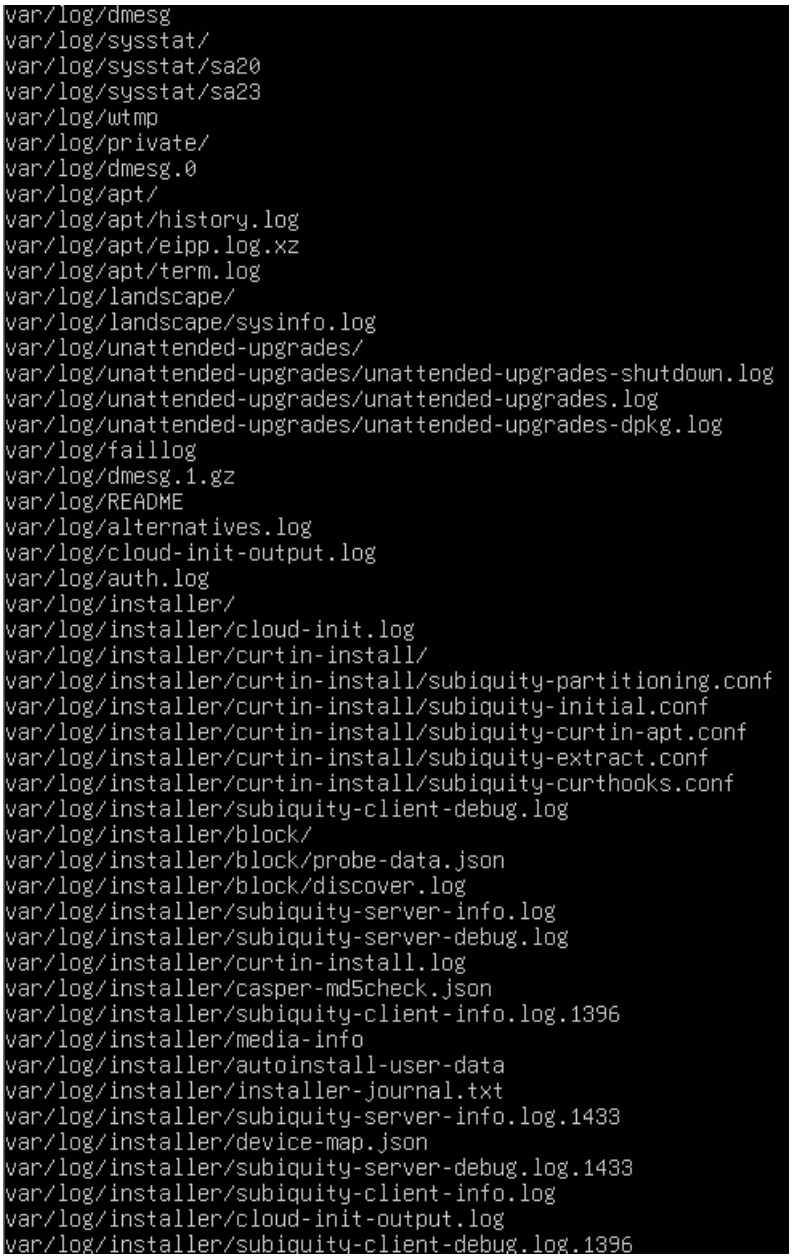
Comando tar -tzf log_backup.tar.gz

-t (list): Le indica a tar que debe listar los contenidos del archivo, en lugar de extraerlos.

-z (gzip): Especifica que el archivo está comprimido con gzip.

-v (verbose): Muestra el listado detallado (con más información como los permisos, propietario, tamaño y fecha) de cada archivo dentro del archivo comprimido.

-f (file): Indica que a continuación se especificará el nombre del archivo con el que se va a trabajar.



4. Extrae únicamente el archivo syslog (o messages) de log_backup.tar.gz a tu directorio /tmp .

```
brayan1@bserver:~$ pwd
/home/brayan1
brayan1@bserver:~$ tar -xzf log_bakup.tar.gz -C /tmp var/log/messages
tar: var/log/messages: Not found in archive
tar: Exiting with failure status due to previous errors
brayan1@bserver:~$ tar -xzf log_bakup.tar.gz -C /tmp var/log/syslog
brayan1@bserver:~$
```

5. Crea tres archivos (a.txt , b.log , c.jpg) y luego crea un archivo zip que los contenga.

```
brayan1@bserver:~$ ls
dos  log_bakup.tar  log_bakup.tar.gz  mi_bin  palabras.txt  test_locate.txt
brayan1@bserver:~$ touch a.txt b.log c.jpg
brayan1@bserver:~$ ls
a.txt  b.log  c.jpg  dos  log_bakup.tar  log_bakup.tar.gz  mi_bin  palabras.txt  test_locate.txt
```

Con gzip de Linux

tar -czvf mis_archivos.tar.gz a.txt b.log c.jpg

-c: Crea un nuevo archivo .tar.

-z: Comprime el archivo con gzip.

-v: Muestra los archivos que se están procesando (verbose).

-f: Indica que el siguiente argumento es el nombre del archivo de salida.

```
brayan1@bserver:~$ tar -czvf log_bakip.tar.gz a.txt b.log c.jpg
a.txt
b.log
c.jpg
```
rm

6. Elimina los tres archivos originales y luego recupéralos desde el archivo zip .´

```
brayan1@bserver:~$ ls
a.txt  b.log  c.jpg  dos  log_bakup.tar  log_bakup.tar.gz  mi_bin  palabras.txt  test_loc
brayan1@bserver:~$ rm a.txt b.log c.jpg
brayan1@bserver:~$ ls
dos  log_bakup.tar  log_bakup.tar.gz  mi_bin  palabras.txt  test_locate.txt
brayan1@bserver:~$
```

```
brayan1@bserver:~$ ls
dos  log_bakup.tar  mi_bin  palabras.txt  test_locate.txt  v  var
brayan1@bserver:~$ tar -xf log_bakup.tar
brayan1@bserver:~$ ls
a.txt  b.log  c.jpg  dos  log_bakup.tar  mi_bin  palabras.txt  test_locate.txt  v  var
brayan1@bserver:~$
```

7. Usa zcat (o gzcat) para leer el contenido de un archivo de log comprimido (ej: en /var/log , busca uno que termine en .gz) sin crear un archivo descomprimido.

```
brayan1@bserver:/var/log$ zcat dmesg.2.gz | tail
[   71.174366] kernel: audit: type=1400 audit(1760960568.085:117): apparmor="STATUS" operation="profile_load" profile=
mm="apparmor_parser"
[   71.195234] kernel: audit: type=1400 audit(1760960568.106:118): apparmor="STATUS" operation="profile_load" profile=
parmor_parser"
[   71.213163] kernel: audit: type=1400 audit(1760960568.124:119): apparmor="STATUS" operation="profile_load" profile=
armor_parser"
[   71.228130] kernel: audit: type=1400 audit(1760960568.139:120): apparmor="STATUS" operation="profile_load" profile=
parmor_parser"
[   89.703978] kernel: workqueue: drm_fb_helper_damage_work hogged CPU for >10000us 64 times, consider switching to WQ
[   95.290872] kernel: cfg80211: Loading compiled-in X.509 certificates for regulatory database
[   95.302482] kernel: Loaded X.509 cert 'sforshee: 00b28ddf47aef9cea7'
[   95.305890] kernel: Loaded X.509 cert 'wens: 61c038651aabdcf94bd0ac7ff06c7248db18c600'
[   95.397593] kernel: e1000: enp0s3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[  129.413081] kernel: workqueue: drm_fb_helper_damage_work hogged CPU for >10000us 128 times, consider switching to W
brayan1@bserver:/var/log$
```

6.- Redirección,Tuberías y Filtros

1. Guarda la lista de archivos de tu directorio home (formato largo) en un

archivo mis_archivos.txt .

```
brayan1@bserver:~$ ls -l ~ > mis_archivos.txt
brayan1@bserver:~$ ls
a.txt  b.log  c.jpg  dos  log_bakup.tar  mi_bin  mis_archivos.txt  palabras.txt  test_locate.txt
brayan1@bserver:~$ cat mis_archivos.txt
total 16
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 a.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 b.log
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 c.jpg
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:41 dos
-rw-rw-r-- 1 brayan1 brayan1 10240 oct 26 08:19 log_bakup.tar
drwxrwxr-x 2 brayan1 brayan1  4096 oct 20 12:29 mi_bin
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 09:10 mis_archivos.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:41 palabras.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:42 test_locate.txt
brayan1@bserver:~$ ls
```

2. Sin borrar el contenido anterior, añade la fecha y hora actual al final del archivo mis_archivos.txt .

```
brayan1@bserver:~$ echo "miercoles 22 octubre 2025 10:55" >>mis_archivos.txt
brayan1@bserver:~$ cat mis_archivos.txt
total 16
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 a.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 b.log
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 08:15 c.jpg
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:41 dos
-rw-rw-r-- 1 brayan1 brayan1 10240 oct 26 08:19 log_bakup.tar
drwxrwxr-x 2 brayan1 brayan1  4096 oct 20 12:29 mi_bin
-rw-rw-r-- 1 brayan1 brayan1     0 oct 26 09:10 mis_archivos.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:41 palabras.txt
-rw-rw-r-- 1 brayan1 brayan1     0 oct 23 12:42 test_locate.txt
miercoles 22 octubre 2025 10:55
brayan1@bserver:~$ _
```

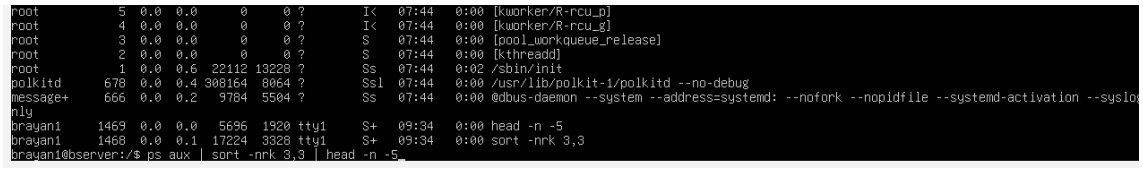
3. Usa grep y una tubería (|) para contar el número de directorios que hay en /etc . (Pista: ls -l | grep '^d').

```
brayan1@bserver:~$ ls -l /etc  | grep '^d' | wc -l
106
brayan1@bserver:~$
```

4. Muestra las 10 últimas líneas del archivo /etc/passwd y, usando otra tubería, extrae solo los nombres de usuario (el primer campo).

```
brayan1@bserver:/$ tail /etc/passwd | cut -d: -f1
polkitd
syslog
uuidd
tcpdump
tss
landscape
fwupd-refresh
usbmux
sshd
brayan1
brayan1@bserver:/$ _
```

5. Muestra una lista de todos los procesos del sistema (ps aux), ordénala por uso de CPU (tercera columna) y muestra solo las 5 líneas superiores.

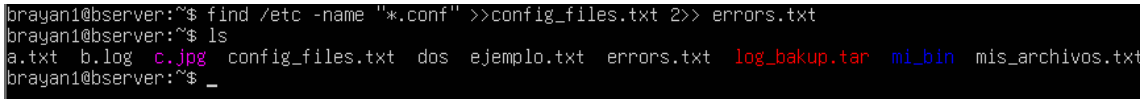


6. ¿Cuál es la diferencia entre usar > y >> para redirigir la salida de un comando a un archivo? Demuéstralo con un ejemplo.

> **Sobrescribir** Redirige la salida de un comando a un archivo. Si el archivo ya existe, su contenido es borrado y reemplazado por la nueva salida. Si el archivo no existe, lo crea.

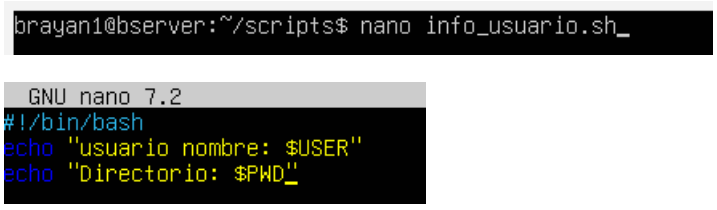
>> **Añadir** Redirige la salida de un comando y la añade al final del archivo. Preserva el contenido original del archivo. Si el archivo no existe, lo crea.

7. Ejecuta find /etc -name "*.conf" . Redirige la salida estándar a un archivo config_files.txt y los errores (si los hay) a errors.txt .



7.- Scripts Básicos

1. Crea un script que imprima tu nombre de usuario y el directorio de trabajo actual usando las variables de entorno correspondientes.



```
brayan1@bserver:~/scripts$ ls
info_usuario.sh
brayan1@bserver:~/scripts$ cat info_usuario.sh
#!/bin/bash
echo "usuario nombre: $USER"
echo "Directorio: $PWD"
brayan1@bserver:~/scripts$ ./info_usuario.sh
usuario nombre: brayan1
Directorio: /home/brayan1/scripts
brayan1@bserver:~/scripts$ _
```

2. Haz el script anterior ejecutable solo para ti (chmod u+x ...) y ejecútalo. Luego, intenta ejecutarlo como otro usuario (si es posible) o explica qué pasaría.

```
brayan1@bserver:~/scripts$ chmod u+x info_usuario.sh
brayan1@bserver:~/scripts$ ls -l
total 4
-rwxrwxr-x 1 brayan1 brayan1 65 oct 26 11:08 info_usuario.sh
brayan1@bserver:~/scripts$ ./info_usuario.sh
usuario nombre: brayan1
Directorio: /home/brayan1/scripts
brayan1@bserver:~/scripts$ ./info_usuario.sh_
```

Si el archivo tiene permisos solo para el usuario propietario (u+x), otro usuario no podrá ejecutarlo y obtendrá errores de permiso denegado.

3. Modifica el script para que acepte un argumento. Si el argumento es “hola”, debe imprimir “mundo”. Si es cualquier otra cosa, no debe imprimir nada.

```
 GNU nano 7.2
#!/bin/bash
if [ "$1" == "hola" ]; then
        echo "mundo"
fi
```

```
brayan1@bserver:~/scripts$ ./script_if.sh hola
mundo
brayan1@bserver:~/scripts$ ./script_if.sh ahora
brayan1@bserver:~/scripts$ _
```

4. Mejora el script anterior para que, si no se proporciona ningún argumento, muestre un mensaje de uso: “Error: Debes proporcionar un argumento.”

```
  GNU nano 7.2
#!/bin/bash
if [ -z "$1" ]; then
        echo "Error:Debes proporcionar un argumento"
        exit 1
fi

if [ "$1" == "hola" ]; then
        echo "mundo"
fi
```

```
brayan1@bserver:~/scripts$ ./script_if.sh
Error:Debes proporcionar un argumento
brayan1@bserver:~/scripts$ ./script_if.sh hola
mundo
brayan1@bserver:~/scripts$ ./script_if.sh adios
brayan1@bserver:~/scripts$
```

5. Escribe un script que reciba dos números. Debe imprimir “iguales” si son iguales y “diferentes” si no lo son.

```
  GNU nano 7.2
#!/bin/bash
if [ "$1" -eq "$2" ]; then
        echo "Iguales"
else
        echo "Diferentes"
fi
```

```
brayan1@bserver:~/scripts$ ls
info_usuario.sh  niguales.sh  script_if.sh
brayan1@bserver:~/scripts$ ./niguales.sh 4 5
bash: ./niguales.sh: Permission denied
brayan1@bserver:~/scripts$ chmod +x niguales.sh
brayan1@bserver:~/scripts$ ./niguales.sh 4 5
Diferentes
brayan1@bserver:~/scripts$ ./niguales.sh 4 4
Iguales
brayan1@bserver:~/scripts$
```

6. Escribe un script que, dado un directorio como argumento, use un bucle for para iterar sobre su contenido (ls $1) y añada la extensión .bak a cada archivo.

```
  GNU nano 7.2
#!/bin/bash

if [ -z "$1" ]; then
        echo "Proporciona un directorio"
        exit 1

fi

for archivo in $(ls "$1"); do
        mv "$1/$archivo" "$1/${archivo}.bak"
done
```

```
brayan1@bserver:~/scripts$ mkdir A
brayan1@bserver:~/scripts$ CD A
CD: command not found
brayan1@bserver:~/scripts$ cd A
brayan1@bserver:~/scripts/A$ touch a.txt b.txt c.txt
brayan1@bserver:~/scripts/A$ ls
a.txt  b.txt  c.txt
brayan1@bserver:~/scripts/A$ _
```

```
brayan1@bserver:~/scripts/A$ ls
a.txt  bak.sh  b.txt  c.txt
brayan1@bserver:~/scripts/A$ ./bak.sh
Proporciona un directorio
brayan1@bserver:~/scripts/A$ ./bak.sh ~/scripts/A
brayan1@bserver:~/scripts/A$ ls
a.txt.bak  bak.sh.bak  b.txt.bak  c.txt.bak
brayan1@bserver:~/scripts/A$
```

8.- Ejercicios Avanzados

1. Muestra los shells de los usuarios listados en /etc/passwd , elimina las líneas duplicadas y ordénalos alfabéticamente. (Pista: cut , sort , uniq).

```
brayan1@bserver:/$ cut -d: -f7 /etc/passwd | sort | uniq
/bin/bash
/bin/false
/bin/sync
/usr/sbin/nologin
brayan1@bserver:/$ _
```

- `cut -d:` → usa ":" como delimitador (en `/etc/passwd` los campos están separados por `:`).
- `-f7` → toma el **séptimo campo**, que es el **shell** del usuario.
- `sort` → ordena alfabéticamente.
- `uniq` → elimina duplicados.

2. Usando ps , grep y wc , crea un comando de una sola línea que te diga cuántos procesos está ejecutando el usuario root actualmente.

```
brayan1@bserver:/$ ps -ef | grep '^root' | wc -l
85
brayan1@bserver:/$ _
```

3. Lista todos los archivos en /etc , filtra los resultados para mostrar solo

aquellos que han sido modificados en “Oct” (octubre) y guarda esa lista en

october_files.txt .

```
brayan1@bserver:/$ ls -l /etc | grep 'Oct' > october_files.txt
bash: october_files.txt: Permission denied
brayan1@bserver:/$ sudo su ls -l /etc | grep 'Oct' > october_files.txt
bash: october_files.txt: Permission denied
[sudo] password for brayan1:
Sorry, try again.
[sudo] password for brayan1:
su: user ls does not exist or the user entry does not contain all the required fields
brayan1@bserver:/$
```

4. Usando globbing , lista todos los archivos en /etc que contengan un número

en su nombre.

```
brayan1@bserver:/$ ls /etc | grep '[0-9]'
dbus-1
e2scrub.conf
iproute2
libnl-3
mke2fs.conf
polkit-1
python3
python3.12
rc0.d
rc1.d
rc2.d
rc3.d
rc4.d
rc5.d
rc6.d
sensors3.conf
udisks2
X11
brayan1@bserver:/$
```

5. Usando find , busca en /usr/bin todos los archivos que sean ejecutables, pero que no sean propiedad del usuario root .

```
brayan1@bserver:/$ find usr/bin -type f -perm /111 ! -user root
brayan1@bserver:/$ find /usr/bin -type f -perm /111 ! -user root
brayan1@bserver:/$ find /usr/bin -type f -perm /111 ! -user root
```

6. Compara la diferencia de tamaño y velocidad al comprimir un archivo

grande (puedes usar /var/log/syslog) con gzip y con bzip2 .

7. Crea un archivo tar de tu directorio home, pero esta vez, usa la opción para

seguir enlaces simbólicos. Antes, crea un enlace simbólico en tu home para

que puedas ver la diferencia.

```
rayan1@bserver:/$ ln -s /etc/host ~/enlace
rayan1@bserver:/$ tar -cvhf homelink.tar ~
ar: homelink.tar: Cannot open: Permission denied
ar: Error is not recoverable: exiting now
rayan1@bserver:/$ tar -cvhf homelink.tar ~
ar: homelink.tar: Cannot open: Permission denied
ar: Error is not recoverable: exiting now
rayan1@bserver:/$ ls
in              boot   dev  home  lib64             lost+found  mnt  proc  run   sbin.usr-is-merged  srv
ln.usr-is-merged  cdrom  etc  lib   lib.usr-is-merged  media       opt  root  sbin  snap                swap.
rayan1@bserver:/$ cd /home
rayan1@bserver:/home$ ls
rayan1
rayan1@bserver:/home$ cd ~/
rayan1@bserver:~$ ls
.txt  b.log  c.jpg  config_files.txt  dos  ejemplo.txt  enlace  errors.txt  log_bakup.tar  mi_bin  mis_archiv
```

8. Escribe un script que reciba una ruta a un archivo. Debe verificar si es un archivo regular, un directorio o si no existe, mostrando un mensaje diferente en cada caso. (Pista: if [-f ...] , if [-d ...]).

```
  GNU nano 7.2
#!/bin/bash
if [ -f "$1" ];then
        echo "$1 es un archivo regular."
elif [ -d "$1" ]; then
        echo "$1 es directorio"
else
        echo "$1 no existe"
fi
```

```
brayan1@bserver:~/scripts$ ls
A  info_usuario.sh  niguales.sh  script_if.sh  verificar_archivo.sh
brayan1@bserver:~/scripts$ ./verificar_archivo.sh /etc/passwd
bash: ./verificar_archivo.sh: Permission denied
brayan1@bserver:~/scripts$ chmod +x verificar_archivo.sh
brayan1@bserver:~/scripts$ ./verificar_archivo.sh /etc/passwd
/etc/passwd es un archivo regular.
brayan1@bserver:~/scripts$ ./verificar_archivo.sh /etc
/etc es directorio
brayan1@bserver:~/scripts$ ./verificar_archivo.sh /sddfsfsfs
/sddfsfsfs no existe
brayan1@bserver:~/scripts$
```

9. Crea un script que intente crear un directorio llamado test_dir en / . Usando el código de salida ($?), el script debe informar si tuvo éxito o si falló por un problema de permisos.

```
  GNU nano 7.2
#!/bin/bash
mkdir /test_dir 2>/dev/null
if [$? -eq 0 ]; then
        echo "Directorio creado"
else
        echo "Error no se creo"
fi
```

```
brayan1@bserver:~/scripts$ ls
A  info_usuario.sh  niguales.sh  script_if.sh  testdir.sh  verificar_archivo.s
brayan1@bserver:~/scripts$ chmod +x testdir.sh
brayan1@bserver:~/scripts$ ./testdir.sh
./testdir.sh: line 3: [1: command not found
Error no se creo
brayan1@bserver:~/scripts$ _
```

10. Escribe un script que reciba cualquier número de argumentos. El script debe iterar sobre ellos y solo imprimir aquellos que sean números mayores que 10

```
  GNU nano 7.2
#!/bin/bash
for arg in "$@"; do
        if [[ "$arg" =~ ^[0-9]+$ ]] && [ "$arg" -gt 10 ]; then
        echo "$arg"
        fi
done
```

```
brayan1@bserver:~/scripts$ ls
A  info_usuario.sh  mayor10.sh  niguales.sh  script_if.sh  testdir.sh  verificar_archivo.sh
brayan1@bserver:~/scripts$ ./mayor10.sh 3 58 62 14
58
62
14
brayan1@bserver:~/scripts$
```