

Introducción a Linux

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.

```
servidor@servidor:~$ echo $HOME  
/home/servidor  
servidor@servidor:~$ pwd  
/home/servidor  
servidor@servidor:~$ _
```

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

```
servidor@servidor:~$ whoami  
servidor  
servidor@servidor:~$ USUARIO_ACTUAL=servidor  
servidor@servidor:~$ echo $USUARIO_ACTUAL  
servidor  
servidor@servidor:~$ _
```

- 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.

```
servidor@servidor:~$ touch dos palabras.txt  
servidor@servidor:~$ ls  
dos palabras.txt  
servidor@servidor:~$
```

Al no usar comillas, le hemos pasado dos argumentos al comando touch: dos y palabras.txt. Esto ha hecho que se creen dos archivos diferentes.

```
servidor@servidor:~$ rm dos  
servidor@servidor:~$ rm palabras.txt  
servidor@servidor:~$ ls  
servidor@servidor:~$ touch "dos palabras.txt"  
servidor@servidor:~$ ls  
'dos palabras.txt'  
servidor@servidor:~$
```

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

```

servidor@servidor:~$ type cd
cd is a shell builtin
servidor@servidor:~$ type ls
ls is aliased to `ls --color=auto'
servidor@servidor:~$ _

```

Cd es interno y ls es externo. En la practica supone que, al contrario de cd, ls puede no encontrarse en ciertas distribuciones y que tengamos que instalarlo antes de comenzar a utilizarlo.

- 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.

```

servidor@servidor:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
servidor@servidor:~$ mkdir ./mi_bin
servidor@servidor:~$ ls
'dos palabras.txt'  mi_bin
servidor@servidor:~$ pwd ./mi_bin
/home/servidor
servidor@servidor:~$ PATH=/home/servidor/mi_bin:$PATH
servidor@servidor:~$ echo $PATH
/home/servidor/mi_bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
servidor@servidor:~$ 

```

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?

Para abrirla usé man chmod

```

CHMOD(1)                                         User Commands                                         CHMOD(1)

NAME
    chmod - change file mode bits

SYNOPSIS
    chmod [OPTION]... MODE[.MODE]... FILE...
    chmod [OPTION]... OCTAL-MODE FILE...
    chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION
    This manual page documents the GNU version of chmod.  chmod changes the file mode bits of each given file according to mode, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new mode bits.

    The format of a symbolic mode is [ugo...] [+|-][perms...]..., where perms is either zero or more letters from the set rwxXst, or a single letter from the set ugo.  Multiple symbolic modes can be given, separated by commas.

    A combination of the letters ugoa controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a).  If none of these are given, the effect is as if (a) were given, but bits that are set in the umask are not affected.

    The operator + causes the selected file mode bits to be added to the existing file mode bits of each file; - causes them to be removed; and = causes them to be added and causes unmentioned bits to be removed except that a directory's unmentioned set user and group ID bits are not affected.

    The letters rwxXst select file mode bits for the affected users: read (r), write (w), execute (x), execute/search only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution (s), restricted deletion flag or sticky bit (t).  Instead of one or more of these letters, you can specify exactly one of the letters ugo: the permissions granted to the user who owns the file (u), the permissions granted to other users who are members of the file's group (g), and the permissions granted to users that are in neither of the two preceding categories (o).

    A numeric mode is from one to four octal digits (0-7), derived by adding up the bits with values 4, 2, and 1.  Octal digits are assumed to be leading zeros.  The first digit selects the set user ID (4) and set group ID (2) and restricted deletion or sticky (1) attributes.  The second digit selects permissions for the user who owns the file: read (4), write (2), and execute (1); the third selects permissions for other users in the file's group, with the same values; and the fourth for other users not in the file's group, with the same values.

    chmod never changes the permissions of symbolic links: the chmod system call cannot change their permissions.  This is not a problem since the permissions of symbolic links are never used.  However, for each symbolic link listed on the command line, chmod changes the permissions of the pointed-to file.  In contrast, chmod ignores symbolic links encountered during recursive directory traversals.

SETUID AND SETGID BITS
    chmod clears the set-group-ID bit of a regular file if the file's group ID does not match the user's effective group ID or one of the user's supplementary group IDs, unless the user has appropriate privileges.  Additional restrictions may cause the set-user-ID and set-group-ID bits of MODE or RFILE to be ignored.  This behavior depends on the policy and functionality of the underlying chmod system call.  When in doubt, check the underlying system behavior.

    For directories, chmod preserves set-user-ID and set-group-ID bits unless you explicitly specify otherwise.  You can set or clear the bits with symbolic modes like u+s and g-s.  To clear these bits for directories with a numeric mode requires an additional leading zero like 00755, leading minus like -6000, or leading equals like =755.

Manual name chmod(1) line 1 (press h for help or q to quit)

```

Se encuentra en la sección 1 que es la de comandos de usuario.

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

```
-S      sort by file size, largest first

--sort=WORD
        sort by WORD instead of name: none (-U), size (-S)

--time=WORD
        select which timestamp used to display or sort; a
        (default): mtime, modification; birth time: birth,
        with -l, WORD determines which time to show; with

--time-style=TIME_STYLE
        time/date format with -l; see TIME_STYLE below

-t      sort by time, newest first; see --time

-T, --tabsize=COLS
/sort by file size
```

- 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.

```
servidor@servidor:/$ sudo find / -name passwd
/usr/share/lintian/overrides/passwd
/usr/share/doc/passwd
/usr/share/bash-completion/completions/passwd
/usr/bin/passwd
/etc/pam.d/passwd
/etc/passwd
```

- 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?

```
servidor@servidor:/$ sudo touch home/test_locate.txt
servidor@servidor:/$ sudo locate test_locate.txt
/home/test_locate.txt
```

Lo encuentra porque utilicé sudo updatedb para actualizar la base de datos de locate.

- 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.

Se debe utilizar sudo updatedb.

```
servidor@servidor:/$ sudo updatedb
servidor@servidor:/$ sudo locate test_locate.txt
/home/test_locate.txt
```

3) Navegación y Listado de Archivos

- Navega al directorio /etc. Desde ahí, sin usar cd, lista el contenido de tu directorio home usando una ruta con el atajo ~.

```
servidor@servidor:~$ cd /etc
servidor@servidor:/etc$ ls ~
total 44
drwxr-x--- 6 servidor servidor 4096 Oct 16 08:34 .
drwxr-xr-x  3 root      root    4096 Oct 16 09:02 ..
-rw-----  1 servidor servidor  918 Oct 16 07:39 .bash_history
-rw-r--r--  1 servidor servidor 220 Mar 31 2024 .bash_logout
-rw-r--r--  1 servidor servidor 3771 Mar 31 2024 .bashrc
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rw-rw-r--  1 servidor servidor     0 Oct 16 08:14 'dos palabras.txt'
-rw-----  1 servidor servidor   66 Oct 16 08:34 .lesshst
drwxrwxr-x  3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x  2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-r--r--  1 servidor servidor  807 Mar 31 2024 .profile
drwx----- 2 servidor servidor 4096 Oct 16 06:58 .ssh/
-rw-r--r--  1 servidor servidor     0 Oct 15 09:21 .sudo_as_admin_successful
```

- Desde tu directorio home, navega a / y luego a var y finalmente a log usando una sola línea de comando y rutas relativas.

```
servidor@servidor:/home$ cd ../../var/log
servidor@servidor:/var/log$ pwd
/var/log
```

- Lista el contenido de /etc en formato largo. En la salida, identifica el propietario, el grupo y los permisos del archivo passwd.

Para listar en formato largo utilizamos ll /etc.

```
-rw-r--r--  1 root root          8 Aug  5 17:02 timezone
drwxr-xr-x  2 root root        4096 Aug  5 17:14 tmpfiles.d/
drwxr-xr-x  2 root root        4096 Aug  5 17:14 ubuntu Advantage/
-rw-r--r--  1 root root       1260 Jan 27 2023 ucf.conf
drwxr-xr-x  4 root root        4096 Aug  5 17:02 udev/
drwxr-xr-x  2 root root        4096 Oct 15 09:17 udisks2/
drwxr-xr-x  3 root root        4096 Aug  5 17:14 ufw/
-rw-r--r--  1 root root       208 Aug  5 16:54 .updated
-rw-r--r--  1 root root       583 Jul 31 2023 updatedb.conf
drwxr-xr-x  3 root root        4096 Aug  5 17:02 update-manager/
drwxr-xr-x  2 root root        4096 Aug  5 17:14 update-motd.d/
drwxr-xr-x  2 root root        4096 Aug  5 17:14 update-notifier/
drwxr-xr-x  2 root root        4096 Oct 15 09:16 UPower/
-rw-r--r--  1 root root      1523 Aug  5 17:14 usb_modeswitch.conf
drwxr-xr-x  2 root root        4096 Aug  5 17:14 usb_modeswitch.d/
lrwxrwxrwx  1 root root          16 Aug  5 17:02 vconsole.conf -> default/keyboard
drwxr-xr-x  2 root root        4096 Oct 15 09:17 vim/
drwxr-xr-x  4 root root        4096 Oct 15 09:17 vmware-tools/
lrwxrwxrwx  1 root root          23 Feb 26 2024 vtrgb -> /etc/alternatives/vtrgb
-rw-r--r--  1 root root       4942 Aug  5 17:14 wgetrc
drwxr-xr-x  4 root root        4096 Aug  5 17:02 X11/
-rw-r--r--  1 root root        681 Apr  8 2024 xattr.conf
drwxr-xr-x  4 root root        4096 Aug  5 17:02 xdg/
drwxr-xr-x  2 root root        4096 Aug  5 17:02 xml/
-rw-r--r--  1 root root        460 Aug  5 17:14 zsh_command_not_found
```

Filtrando la respuesta podemos ver el propietario, el grupo y los permisos de passwd

```
servidor@servidor:~/etc | grep "passwd"
-rw-r--r-- 1 root root 1786 Oct 15 09:19 passwd
-rw-r--r-- 1 root root 1734 Oct 15 09:16 passwd-
```

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

La única diferencia está en que ls -l muestra el tamaño de los ficheros en bytes mientras que añadirle -h los formatea mostrándolo más legible para el usuario.

- 5) Ejecuta ls -R ~. Explica qué hace la opción -R y por qué podría ser peligroso usarla en el directorio raíz (/).

```
servidor@servidor:~$ ls -R ~
/home/servidor:
'dos palabras.txt'    mi_bin

/home/servidor/mi_bin:
```

La opción -R lista de forma recursiva, esto significa que mostrará cada fichero, directorio y subdirectorio. Es peligroso usarlo en la raíz ya que mostraría absolutamente todos los archivos del sistema pudiendo sobrepasar las capacidades de la maquina.

4) Manipulación de Archivos y Directorios

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

```
servidor@servidor:~$ mkdir -p proyecto/src proyecto/doc proyecto/bin
servidor@servidor:~$ tree proyecto
proyecto
└── bin
└── doc
└── src

4 directories, 0 files
```

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

```
servidor@servidor:~$ touch notas.txt
servidor@servidor:~$ mv notas.txt ./proyecto/doc/README.d
servidor@servidor:~$ ll proyecto/doc
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 16 09:52 .
drwxrwxr-x 5 servidor servidor 4096 Oct 16 09:50 ..
-rw-rw-r-- 1 servidor servidor 0 Oct 16 09:51 README.d
```

- 3) Copia el archivo README.md de proyecto/doc a proyecto/bin.

Luego, borra el archivo original de la carpeta doc.

```
servidor@servidor:~$ cp ./proyecto/doc/README.md ./proyecto/bin
servidor@servidor:~$ ll ./proyecto/bin
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 16 09:54 ../
drwxrwxr-x 5 servidor servidor 4096 Oct 16 09:50 ...
-rw-rw-r-- 1 servidor servidor 0 Oct 16 09:54 README.md
servidor@servidor:~$ rm ./proyecto/doc/README.md
servidor@servidor:~$ ll ./proyecto/doc/
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 16 09:54 ../
drwxrwxr-x 5 servidor servidor 4096 Oct 16 09:50 ...
```

- 4) Intenta borrar el directorio proyecto con rmdir. Explica el error obtenido. Luego, usa rm con la opción correcta para borrar el directorio y todo su contenido.

```
servidor@servidor:~$ rmdir proyecto/
rmdir: failed to remove 'proyecto/': Directory not empty
servidor@servidor:~$ rm -r proyecto/
servidor@servidor:~$ ll
total 44
drwxr-x--- 6 servidor servidor 4096 Oct 16 09:56 ../
drwxr-xr-x 3 root      root    4096 Oct 16 09:02 ...
-rw----- 1 servidor servidor 918 Oct 16 07:39 .bash_history
-rw-r--r-- 1 servidor servidor 220 Mar 31 2024 .bash_logout
-rw-r--r-- 1 servidor servidor 3771 Mar 31 2024 .bashrc
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rw-rw-r-- 1 servidor servidor 0 Oct 16 08:14 'dos palabras.txt'
-rw----- 1 servidor servidor 66 Oct 16 08:34 .lessht
drwxrwxr-x 3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-r--r-- 1 servidor servidor 807 Mar 31 2024 .profile
drwx----- 2 servidor servidor 4096 Oct 16 06:58 .ssh/
-rw-r--r-- 1 servidor servidor 0 Oct 15 09:21 .sudo_as_admin_successful
```

El error indica que el directorio no se puede borrar porque otros subdirectorios cuelgan de él.

- 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.

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

5) Archivado y Compresión

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

```
servidor@servidor:/$ tar cvf log_backup.tar /var/log
tar: log_backup.tar: Cannot open: Permission denied
tar: Error is not recoverable: exiting now
```

Aparecen porque no estamos usando permisos de administrador, con usar sudo se arreglaría.

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

```
servidor@servidor:~$ sudo gzip log_backup.tar
servidor@servidor:~$ ls -lh log_backup.tar.gz
-rw-r--r-- 1 root root 3.8M Oct 16 10:07 log_backup.tar.gz
```

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

Usamos tar tfz log backup.tar.gz

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

```
servidor@servidor:/ $ sudo tar zxt log_backup.tar.gz -C /tmp syslog
servidor@servidor:/ $ ll /tmp
total 52
drwxrwxrwt 13 root      root      4096 Oct 16 10:15 .
drwxr-xr-x 23 root      root      4096 Oct 16 10:07 ..
drwxrwxrwt  2 root      root      4096 Oct 16 08:00 .font-unix/
drwxrwxrwt  2 root      root      4096 Oct 16 08:00 .ICE-unix/
drwx-----  2 root      root      4096 Oct 16 08:00 snap-private-tmp/
-rw-rw-r--   1 servidor  servidor  0 Oct 16 10:15 syslog
drwx-----  3 root      root      4096 Oct 16 08:00 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-ModemManager.service-KBMISY/
drwx-----  3 root      root      4096 Oct 16 08:00 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-polkit.service-VtIITT/
drwx-----  3 root      root      4096 Oct 16 08:00 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-systemd-logind.service-pNKhA/
drwx-----  3 root      root      4096 Oct 16 08:00 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-systemd-resolved.service-P3RHEE/
drwx-----  3 root      root      4096 Oct 16 08:00 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-timesyncd.service-hj3CNV/
drwx-----  3 root      root      4096 Oct 16 08:33 systemd-private-2e107ae1e1af4dbb8057a8d238292cb3-upower.service-2KckG6/
drwxrwxrwt  2 root      root      4096 Oct 16 08:00 .X11-unix/
drwxrwxrwt  2 root      root      4096 Oct 16 08:00 .XIM-unix/
```

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

```
servidor@servidor:~$ touch a.txt b.log c.jpg
servidor@servidor:~$ zip letras.zip a.txt b.log c.jpg
  adding: a.txt (stored 0%)
  adding: b.log (stored 0%)
  adding: c.jpg (stored 0%)
servidor@servidor:~$ ll
total 48
drwxr-x--- 6 servidor servidor 4096 Oct 16 10:19 .
drwxr-xr-x 3 root      root    4096 Oct 16 09:02 ..
-rw-rw-r-- 1 servidor servidor  0 Oct 16 10:19 a.txt
-rw----- 1 servidor servidor 918 Oct 16 07:39 .bash_history
-rw-r--r-- 1 servidor servidor 220 Mar 31 2024 .bash_logout
-rw-r--r-- 1 servidor servidor 3771 Mar 31 2024 .bashrc
-rw-rw-r-- 1 servidor servidor  0 Oct 16 10:19 b.log
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rw-rw-r-- 1 servidor servidor  0 Oct 16 10:19 c.jpg
-rw-rw-r-- 1 servidor servidor  0 Oct 16 08:14 'dos palabras.txt'
-rw----- 1 servidor servidor 66 Oct 16 08:34 .lessht
-rw-r--r-- 1 servidor servidor 436 Oct 16 10:19 letras.zip
drwxrwxr-x 3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-r--r-- 1 servidor servidor 807 Mar 31 2024 .profile
drwx----- 2 servidor servidor 4096 Oct 16 06:58 .ssh/
-rw-r--r-- 1 servidor servidor  0 Oct 15 09:21 .sudo_as_admin_successful
```

- 6) Elimina los tres archivos originales y luego recuperálos desde el archivo zip.

```
servidor@servidor:~$ rm a.txt b.log c.jpg
servidor@servidor:~$ ll
total 48
drwxr-x--- 6 servidor servidor 4096 Oct 16 10:20 .
drwxr-xr-x 3 root      root    4096 Oct 16 09:02 ..
-rw----- 1 servidor servidor 918 Oct 16 07:39 .bash_history
-rw-r--r-- 1 servidor servidor 220 Mar 31 2024 .bash_logout
-rw-r--r-- 1 servidor servidor 3771 Mar 31 2024 .bashrc
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rw-rw-r-- 1 servidor servidor 0 Oct 16 08:14 'dos palabras.txt'
-rw----- 1 servidor servidor 66 Oct 16 08:34 .lessht
-rw-rw-r-- 1 servidor servidor 436 Oct 16 10:19 letras.zip
drwxrwxr-x 3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-r--r-- 1 servidor servidor 807 Mar 31 2024 .profile
drwx----- 2 servidor servidor 4096 Oct 16 06:58 .ssh/
-rw-r--r-- 1 servidor servidor 0 Oct 15 09:21 .sudo_as_admin_successful
servidor@servidor:~$ unzip letras.zip
Archive: letras.zip
  extracting: a.txt
  extracting: b.log
  extracting: c.jpg
servidor@servidor:~$ ll
total 48
drwxr-x--- 6 servidor servidor 4096 Oct 16 10:21 .
drwxr-xr-x 3 root      root    4096 Oct 16 09:02 ..
-rw-rw-r-- 1 servidor servidor 0 Oct 16 10:19 a.txt
-rw----- 1 servidor servidor 918 Oct 16 07:39 .bash_history
-rw-r--r-- 1 servidor servidor 220 Mar 31 2024 .bash_logout
-rw-r--r-- 1 servidor servidor 3771 Mar 31 2024 .bashrc
-rw-rw-r-- 1 servidor servidor 0 Oct 16 10:19 b.log
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rw-rw-r-- 1 servidor servidor 0 Oct 16 10:19 c.jpg
-rw-rw-r-- 1 servidor servidor 0 Oct 16 08:14 'dos palabras.txt'
-rw----- 1 servidor servidor 66 Oct 16 08:34 .lessht
-rw-rw-r-- 1 servidor servidor 436 Oct 16 10:19 letras.zip
drwxrwxr-x 3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-r--r-- 1 servidor servidor 807 Mar 31 2024 .profile
drwx----- 2 servidor servidor 4096 Oct 16 06:58 .ssh/
-rw-r--r-- 1 servidor servidor 0 Oct 15 09:21 .sudo_as_admin_successful
```

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

```
servidor@servidor:~$ cd /var/log
servidor@servidor:/var/log$ ll
total 3272
drwxrwxr-x 10 root      syslog          4096 Oct 16 08:01 .
drwxr-xr-x 13 root      root            4096 Oct 15 09:19 ..
-rw-r--r--  1 root      root            34416 Oct 16 09:02 alternatives.log
-rw-r----- 1 root      adm             0 Oct 15 09:19 apport.log
drwxr-xr-x  2 root      root            4096 Oct 16 10:19 apt/
-rw-r----- 1 syslog    adm             41657 Oct 16 10:19 auth.log
-rw-r----- 1 root      root            61229 Aug  5 16:54 bootstrap.log
-rw-rw----  1 root      utmp            768 Oct 15 09:29 btmp
-rw-r----- 1 syslog    adm             72202 Oct 15 09:19 cloud-init.log
-rw-r----- 1 root      adm             4443 Oct 15 09:19 cloud-init-output.log
drwxr-xr-x  2 root      root            4096 Jul 25 16:08 dist-upgrade/
-rw-r----- 1 root      adm             48221 Oct 16 08:01 dmesg
-rw-r----- 1 root      adm             48615 Oct 16 07:39 dmesg.0
-rw-r----- 1 root      adm             15531 Oct 16 07:35 dmesg.1.gz
-rw-r----- 1 root      adm             15431 Oct 16 07:34 dmesg.2.gz
-rw-r----- 1 root      adm             15442 Oct 16 07:33 dmesg.3.gz
-rw-r----- 1 root      adm             15426 Oct 16 07:29 dmesg.4.gz
-rw-r----- 1 root      root            675741 Oct 16 10:19 dpkg.log
-rw-r----- 1 root      root            0 Aug   5 16:54 faillog
drwxrwx---  4 root      adm             4096 Oct 15 09:18 installer/
drwxr-sr-x+ 3 root      systemd-journal 4096 Oct 15 09:19 journal/
-rw-r----- 1 syslog    adm             749333 Oct 16 08:52 kern.log
drwxr-xr-x  2 landscape  landscape       4096 Aug  5 17:14 landscape/
-rw-rw-r--  1 root      utmp            0 Aug   5 16:54 lastlog
drwxr----- 2 root      root            4096 Aug  5 17:02 private/
lrwxrwxrwx  1 root      root            39 Aug   5 17:02 README -> ../../usr/share/doc/systemd/README.logs
-rw-r----- 1 syslog    adm             1453299 Oct 16 10:20 syslog
drwxr-xr-x  2 root      root            4096 Oct 16 06:29 sysstat/
drwxr-x---  2 root      adm             4096 Oct 15 09:19 unattended-upgrades/
-rw-rw-r--  1 root      utmp            23040 Oct 16 08:01 wtmp
servidor@servidor:/var/log$ gzip dmesg.1.gz
gzip: dmesg.1.gz already has .gz suffix -- unchanged
```

6) Redirección, Tuberías y Filtros

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

```
servidor@servidor:~$ ls -l ~ > mis_archivos.txt
servidor@servidor:~$ cat mis_archivos.txt
total 8
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 a.txt
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 b.log
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 c.jpg
-rw-rw-r-- 1 servidor servidor    0 Oct 16 08:14 dos palabras.txt
-rw-rw-r-- 1 servidor servidor  436 Oct 16 10:19 letras.zip
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin
-rw-rw-r-- 1 servidor servidor    0 Oct 17 06:37 mis_archivos.txt
```

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

```
servidor@servidor:~$ date >> mis_archivos.txt
servidor@servidor:~$ cat mis_archivos.txt
total 8
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 a.txt
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 b.log
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 c.jpg
-rw-rw-r-- 1 servidor servidor    0 Oct 16 08:14 dos palabras.txt
-rw-rw-r-- 1 servidor servidor  436 Oct 16 10:19 letras.zip
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin
-rw-rw-r-- 1 servidor servidor    0 Oct 17 06:37 mis_archivos.txt
Fri Oct 17 06:38:34 AM UTC 2025
```

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

Utilizamos ls -l para sacar la lista detallada con el contenido de /etc. Luego, filtro con grep '^d' para que solo muestre las líneas que comiencen por "d", que son las que muestran datos de directorios. Por último, contamos la cantidad de líneas resultantes con wc -l. El resultado es 106.

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

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

Para lo primero, sencillamente utilizamos el comando tail. Para mostrar los usuarios, utilizamos el comando cut indicando que ":" es el delimitador y que queremos la fila 1.

```
servidor@servidor:~$ tail /etc/passwd  
polkitd:x:991:991:User for polkitd:::/usr/sbin/nologin  
syslog:x:103:104::/nonexistent:/usr/sbin/nologin  
uuidd:x:104:105::/run/uuidd:/usr/sbin/nologin  
tcpdump:x:105:107::/nonexistent:/usr/sbin/nologin  
tss:x:106:108:TPM software stack,,,:/var/lib/tpm:/bin/false  
landscape:x:107:109::/var/lib/landscape:/usr/sbin/nologin  
fwupd-refresh:x:989:989:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin  
usbmux:x:108:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin  
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin  
servidor:x:1000:1000:Pablo:/home/servidor:/bin/bash  
servidor@servidor:~$ cut -d: -f1 /etc/passwd  
root  
daemon  
bin  
sys  
sync  
games  
man  
lp  
mail  
news  
uucp  
proxy  
www-data  
backup  
list  
irc  
_apt  
nobody  
systemd-network  
systemd-timesync  
dhcpcd  
messagebus  
systemd-resolve  
pollinate  
polkitd  
syslog  
uuidd  
tcpdump  
tss  
landscape  
fwupd-refresh  
usbmux  
sshd  
servidor
```

- 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.

Mostramos los procesos con ps aux, los ordeno por la tercera columna en orden número inverso utilizando el comando sort y saco solo las 5 líneas superiores con head.

```
servidor@servidor:~$ ps aux | sort -k 3 -nr | head -n 5
root      71  0.2  0.0      0      0 ?        I   06:36  0:04 [kworker/0:3-cgwb_release]
root     138  0.2  0.0      0      0 ?        I   06:36  0:03 [kworker/0:4-events]
root      1  0.1  0.6  22064 13204 ?      Ss  06:36  0:01 /sbin/init
USER     PID %CPU %MEM   VSZ   RSS TTY STAT START TIME COMMAND
systemd+ 529  0.0  0.3  91024  7808 ?      Ssl 06:36  0:00 /usr/lib/systemd/systemd-timesyncd
```

- 6) Explica la diferencia entre usar > y >> para redirigir la salida de un comando a un archivo. Da un ejemplo.

Al utilizar > se establece el valor dado como contenido del archivo, reemplazando su contenido anterior si lo tuviese. En cambio, >> añade el valor dado al contenido del archivo sin eliminar su contenido.

Ejemplo >:

```
servidor@servidor:~$ echo hola > ejemplo_redireccion.txt
servidor@servidor:~$ echo adios > ejemplo_redireccion.txt
servidor@servidor:~$ cat ejemplo_redireccion.txt
adios
```

Ejemplo >>:

```
servidor@servidor:~$ echo hola >> ejemplo_redireccion.txt
servidor@servidor:~$ echo adios >> ejemplo_redireccion.txt
servidor@servidor:~$ cat ejemplo_redireccion.txt
hola
adios
```

- 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.

Ejecutamos find /etc -name "*.conf" redirigiendo su salida por estándar a config_files.txt y su salida secundaria, o de errores, a errors.txt.

```
servidor@servidor:~$ find /etc -name "*.conf" > config_files.txt 2> errors.txt
servidor@servidor:~$ head -n 5 config_files.txt
/etc/initramfs-tools/initramfs.conf
/etc/initramfs-tools/update-initramfs.conf
/etc/systemd/system.conf
/etc/systemd/journald.conf
/etc/systemd/user.conf
servidor@servidor:~$ cat errors.txt
find: '/etc/credstore': Permission denied
find: '/etc/polkit-1/rules.d': Permission denied
find: '/etc/multipath': Permission denied
find: '/etc/credstore.encrypted': Permission denied
find: '/etc/ssl/private': Permission denied
```

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.

```
servidor@servidor:~$ cat imprime_nombre.sh
#!/bin/bash

echo $USER
echo $PWD
servidor@servidor:~$ chmod +x imprime_nombre.sh
servidor@servidor:~$ ./imprime_nombre.sh
servidor
/home/servidor
```

- 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.

[En el caso de ejecutarlo con otro usuario daría un error de Permission denied.](#)

```
servidor@servidor:~$ chmod go-rwx imprime_nombre.sh
servidor@servidor:~$ ./imprime_nombre.sh
servidor
/home/servidor
```

- 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.

```
servidor@servidor:~$ cat imprime_nombre.sh
#!/bin/bash

echo $USER
echo $PWD

if [ "$1" == "hola" ]; then
    echo "mundo"
fi
servidor@servidor:~$ ./imprime_nombre.sh hola
servidor
/home/servidor
mundo
servidor@servidor:~$ ./imprime_nombre.sh adios
servidor
/home/servidor
servidor@servidor:~$ ./imprime_nombre.sh
servidor
/home/servidor
```

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

```
servidor@servidor:~$ cat imprime_nombre.sh
#!/bin/bash
if [ -z "$1" ]; then
    echo "ERROR: Debes introducir un argumento al comando"
else
    echo $USER
    echo $PWD

    if [ "$1" == "hola" ]; then
        echo "mundo"
    fi
fi
servidor@servidor:~$ ./imprime_nombre.sh
ERROR: Debes introducir un argumento al comando
servidor@servidor:~$ ./imprime_nombre.sh hola
servidor
/home/servidor
mundo
```

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

```
servidor@servidor:~$ cat compara_numeros.sh
#!/bin/bash

if [ "$1" -eq "$2" ]; then
    echo "iguales"
else
    echo "diferentes"
fi
servidor@servidor:~$ chmod +x compara_numeros.sh
servidor@servidor:~$ ./compara_numeros.sh 1 2
diferentes
servidor@servidor:~$ ./compara_numeros.sh 2 2
iguales
```

- 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.

```
servidor@servidor:~$ cat cambia_extension.sh
#!/bin/bash

for archivo in "$1"/*; do
    mv "$archivo" "$archivo.bak"
done
servidor@servidor:~$ ll documentos/
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 20 07:17 ../
drwxr-x--- 7 servidor servidor 4096 Oct 20 07:14 ej1.txt
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej1.txt.bak
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej2.txt
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej3.txt
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej4.txt
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej5.txt
servidor@servidor:~$ ./cambia_extension.sh ~/documentos/
servidor@servidor:~$ ll documentos/
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 20 07:18 ../
drwxr-x--- 7 servidor servidor 4096 Oct 20 07:14 ej1.txt.bak
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej2.txt.bak
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej3.txt.bak
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej4.txt.bak
-rw-rw-r-- 1 servidor servidor 0 Oct 20 07:16 ej5.txt.bak
```

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).

Extraemos la séptima columna de cada línea del archivo /etc/passwd, que es donde se guarda el Shell, usando ":" como delimitador. Luego, ordenamos alfabéticamente con sort. Por último, usamos uniq para eliminar las líneas duplicadas.

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

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

Mostramos todos los procesos que están siendo ejecutados por root.

Filtramos la salida para que elimine las líneas que comienzan por "USER", el encabezado. Contamos el número de líneas.

```
servidor@servidor:~$ ps -u root | grep -v "USER" | wc -l
88
```

- 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.

Muestra los detalles de todos los archivos y directorios dentro de "/etc". Luego, filtramos los resultados para mostrar solo aquellos que contienen "Oct" en la fecha de modificación. Finalmente, redirige la salida del comando al archivo october_files.txt.

```
servidor@servidor:~$ ls -l /etc | grep "^.Oct" > october_files.txt
servidor@servidor:~$ cat october_files.txt
drwxr-xr-x 2 root root      4096 Oct 16 09:02 alternatives
drwxr-xr-x 9 root root      4096 Oct 15 09:12 apt
drwxr-xr-x 5 root root      4096 Oct 15 09:19 cloud
drwxr-xr-x 2 root root      4096 Oct 16 06:56 console-setup
drwxr-xr-x 2 root root      4096 Oct 16 09:02 cron.daily
drwxr-xr-x 3 root root      4096 Oct 16 06:56 default
drwxr-xr-x 4 root root      4096 Oct 15 09:17 dpkg
-rw-r--r-- 1 root root     1853 Oct 17 2022 ethertypes
drwxr-xr-x 4 root root      4096 Oct 15 09:16 fonts
-rw-r--r-- 1 root root      657 Oct 15 09:16 fstab
-rw-r--r-- 1 root root      837 Oct 16 09:02 group
-rw-r--r-- 1 root root      822 Oct 15 09:19 group-
drwxr-xr-x 2 root root      4096 Oct 15 09:13 grub.d
-rw-r----- 1 root shadow     711 Oct 16 09:02 gshadow
-rw-r----- 1 root shadow     699 Oct 15 09:19 gshadow-
-rw-r----- 1 root root          9 Oct 15 09:16 hostname
```

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

Imprimimos todos los archivos y filtramos aquellos que tengan un número en su nombre.

```
servidor@servidor:~$ 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
```

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

```
servidor@servidor:~$ sudo find /usr/bin -type f -executable ! -user root  
servidor@servidor:~$
```

- 6) Compara la diferencia de tamaño y velocidad al comprimir un archivo grande (puedes usar /var/log/syslog) con gzip y con bzip2.

```
servidor@servidor:~$ gzip -c /var/log/syslog > syslog.gz  
servidor@servidor:~$ bzip2 -c /var/log/syslog > syslog.bz2
```

```
-rw-rw-r-- 1 servidor servidor 130697 Oct 20 07:43 syslog.bz2  
-rw-rw-r-- 1 servidor servidor 375933 Oct 20 07:43 syslog.gz
```

Bzip2 ha tardado más en ejecutarse que gzip pero el directorio comprimido resultante pesa menos de la mitad que el de gzip.

- 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.

```
servidor@servidor:~$ ln -s ~/documentos/ ~/enlace
```

```
servidor@servidor:~$ tar -czhf home.tar.gz ~/
tar: Removing leading `/' from member names
tar: Removing leading `/' from hard link targets
servidor@servidor:~$ tar -tzf home.tar.gz
home/servidor/
home/servidor/dos palabras.txt
home/servidor/letras.zip
home/servidor/a.txt
home/servidor/home.tar.gz
home/servidor/ejemplo_redireccion.txt
home/servidor/.bashrc
home/servidor/.sudo_as_admin_successful
home/servidor/october_files.txt
```

La diferencia se encuentra en que, si le indicamos al comando que siga enlaces simbólicos, este se introducirá en el directorio de destino del enlace y comprimirá los archivos que allí se encuentren. En el caso contrario, comprimirá el enlace simbólico como un archivo más.

- 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 ...]).

```
servidor@servidor:~$ cat lee_archivo.sh
#!/bin/bash

if [ -f "$1" ]; then
    echo "$1 es un archivo regular."
elif [ -d "$1" ]; then
    echo "$1 es un directorio."
else
    echo "$1 no existe."
fi
servidor@servidor:~$ ./lee_archivo.sh ~/a.txt
/home/servidor/a.txt es un archivo regular.
servidor@servidor:~$ ./lee_archivo.sh ~/documentos/
/home/servidor/documentos/ es un directorio.
servidor@servidor:~$ ./lee_archivo.sh ~/gatitos.png
/home/servidor/gatitos.png no existe.
```

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.

```
servidor@servidor:~$ cat directorio_fallador.sh
#!/bin/bash

mkdir /test_dir

if [ $? -eq 0 ]; then
    echo "El directorio se creó exitosamente"
else
    echo "Falló la creación del directorio"
fi
servidor@servidor:~$ ./directorio_fallador.sh
mkdir: cannot create directory '/test_dir': Permission denied
Falló la creación del directorio
servidor@servidor:~$ sudo ./directorio_fallador.sh
[sudo] password for servidor:
El directorio se creó exitosamente
```

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.

```
servidor@servidor:~$ cat numeros_grandes.sh
#!/bin/bash

for numero in "$@"; do
    if [[ "$numero" =~ ^[0-9]+\$ ]] && [ "$numero" -gt 10 ]; then
        echo "$numero"
    fi
done
servidor@servidor:~$ ./numeros_grandes.sh 1 23 45 67 0
23
45
67
```