Laboratorio: Fundamentos de Linux y FHS

Curso: Seguridad y Privacidad en TI

Docente: Fabian Eduardo Sierra

Estudiante: Esteban Aguilera

Estudiante: Carlos Barrero

Programa/Grupo: SPTI

Fecha: 26/08/ 2025

Duración sugerida: 2 h 30 min

Introducción

Linux es un sistema operativo que se distingue por su kernel, encargado de controlar el hardware y mantener estable y seguro el funcionamiento del sistema. A partir de este núcleo surgen las distintas distribuciones, que incluyen herramientas y utilidades adicionales para facilitar su uso en diferentes contextos.

En este laboratorio se trabajará sobre aspectos básicos y prácticos de Linux, empezando por reconocer la versión del kernel y la distribución utilizada. Luego, se explorará la jerarquía de directorios (FHS) con el principio de que “todo es un archivo”, lo que permite entender cómo se organiza el sistema. También se practicarán comandos para manipular archivos y rutas, además del uso de comodines y utilidades de texto.

Otro punto importante es el manejo de permisos, ownership, umask y setuid, fundamentales para la seguridad del sistema. También se abordará el uso de /tmp con sticky bit y la diferencia entre los directorios /media y /mnt al momento de montar sistemas de archivos de prueba.

El entorno de trabajo será una máquina virtual con Ubuntu 22.04 LTS y algunos paquetes adicionales que facilitan la exploración. Al final, se busca no solo aplicar los comandos, sino también reflexionar sobre su función y buenas prácticas de uso, dejando como producto un informe con capturas, explicaciones y resultados claros.

RESULTADOS DE APRENDIZAJE

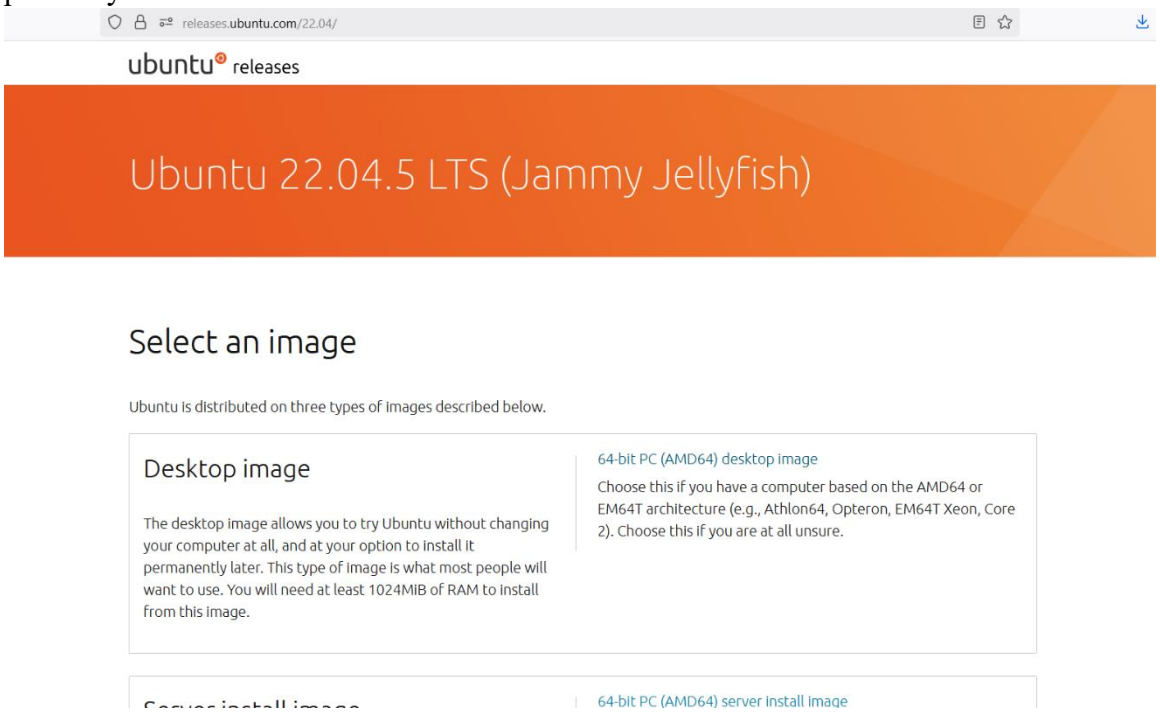
- Comprender qué es Linux, su kernel y el concepto de distribución.
- Reconocer la jerarquía de directorios (FHS) y el principio “todo es un archivo”.
- Manipular archivos, rutas absolutas/relativas, comodines y utilidades de texto.
- Explorar permisos, ownership, umask y setuid; y /tmp con sticky bit.
- Montar un sistema de archivos de prueba y diferenciar /media y /mnt.

ENTORNO RECOMENDADO

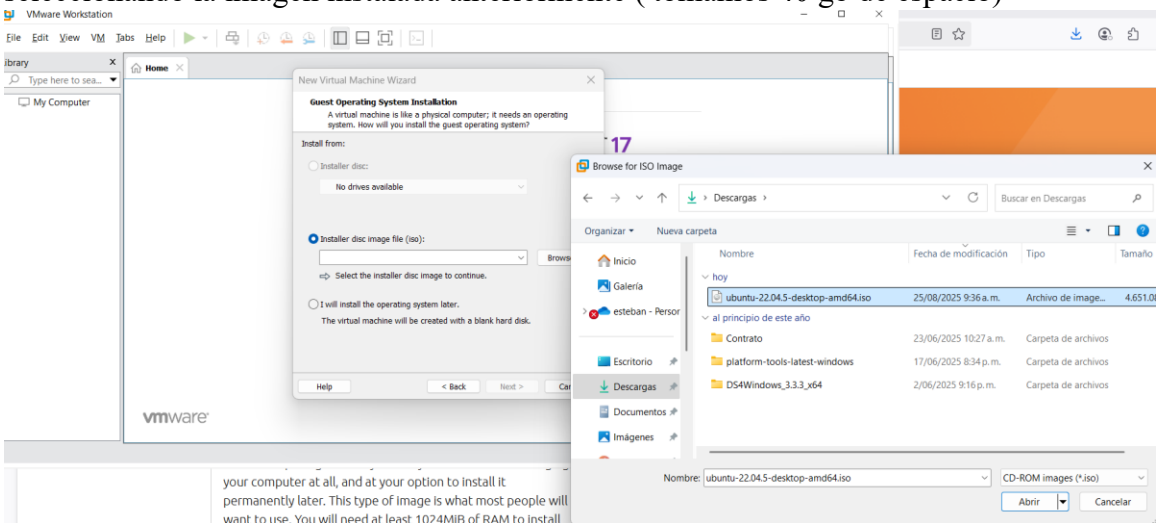
- VM Ubuntu 22.04 LTS (VirtualBox/VmWS) con usuario con sudo
- Paquetes: htop, tree, lsb-release, file, lsof, jq

INSTALACION UBUNTU

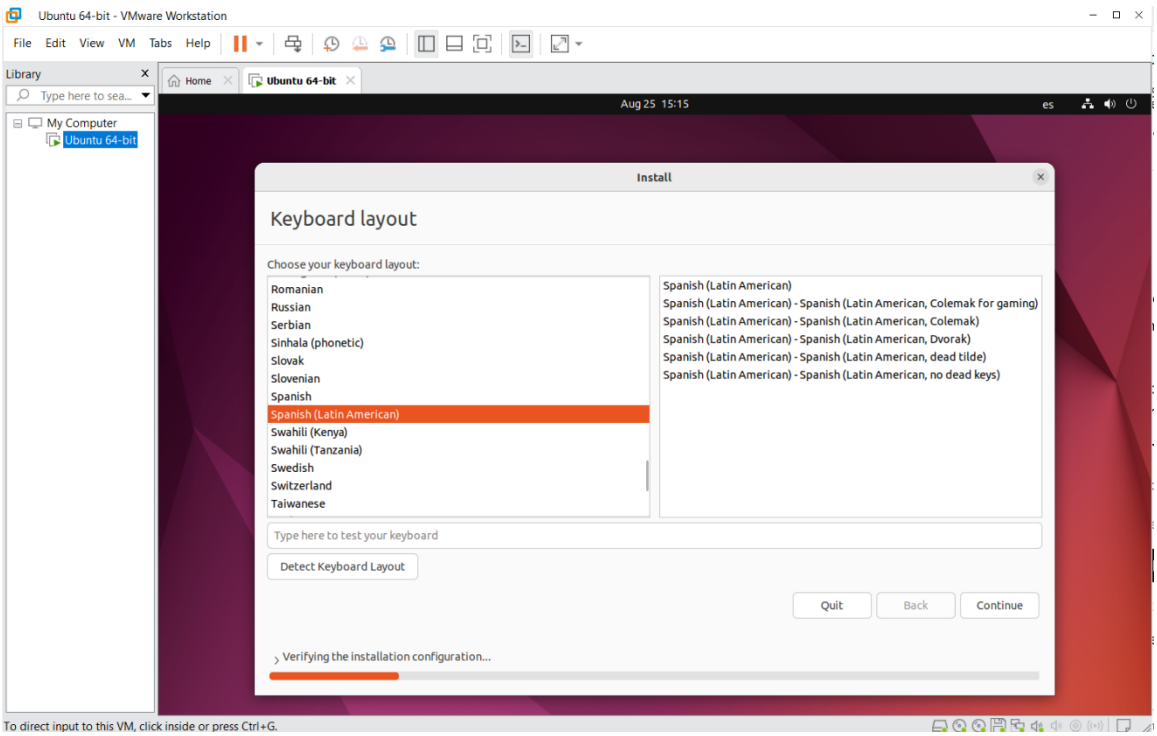
- Primero buscamos la imagen del disco de UBUNTU 22.04 LTS en la página oficial releases.ubuntu.com/22.04/. En este caso seleccionamos la Desktop image para mayor facilidad



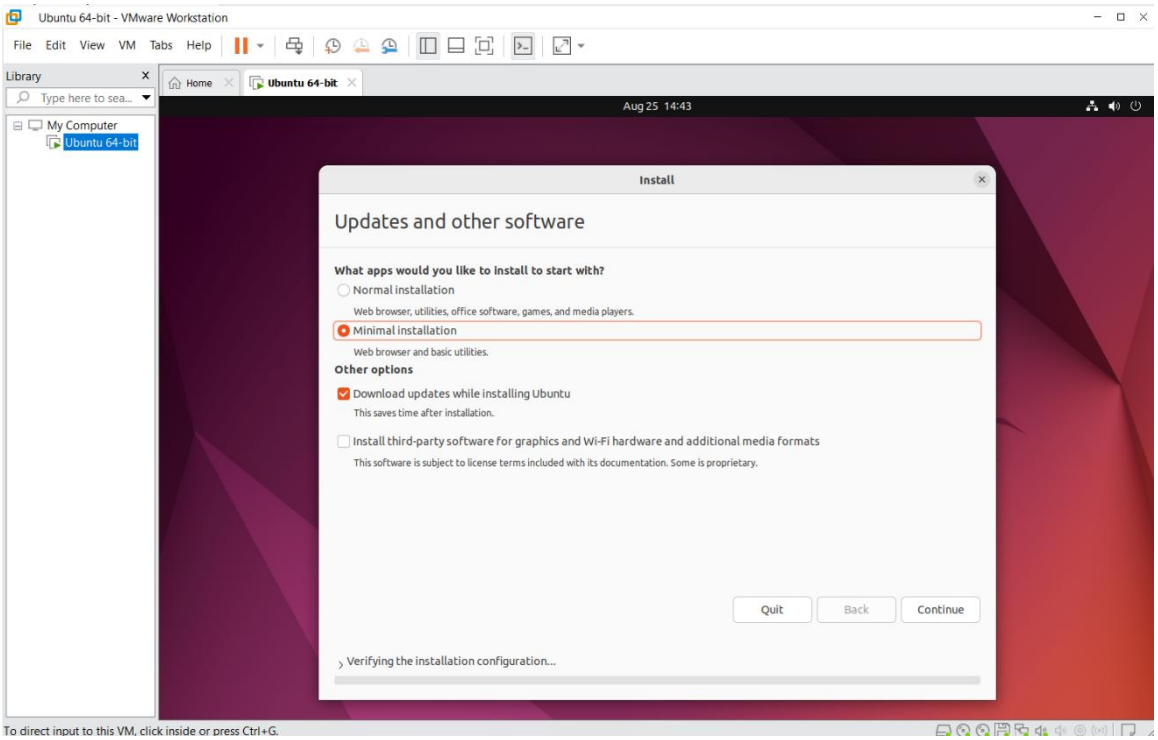
- Ahora mediante la aplicación de VMWare , creamos una nueva maquina virtual , seleccionando la imagen instalada anteriormente (tomamos 40 gb de espacio)



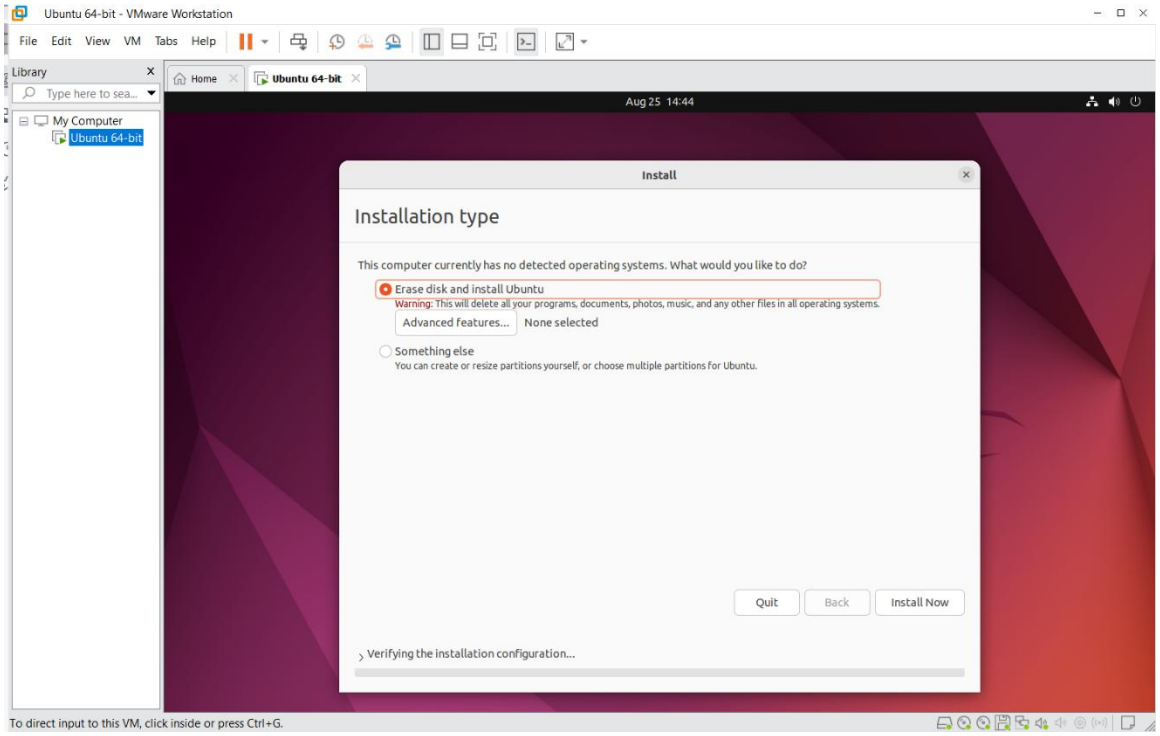
- Una vez insertado el disco e iniciada la máquina virtual nos saldrá la siguiente pantalla. Seleccionamos la distribución del teclado Spanish (Latin American)



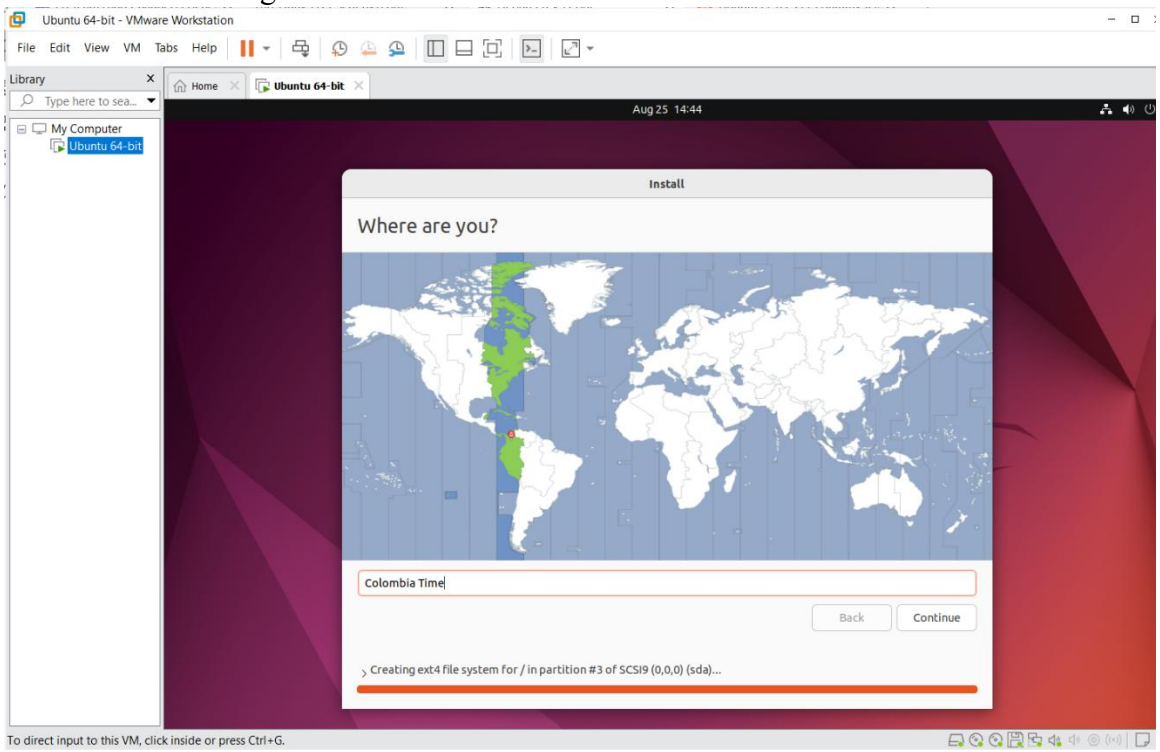
- Seleccionamos la instalación mínima



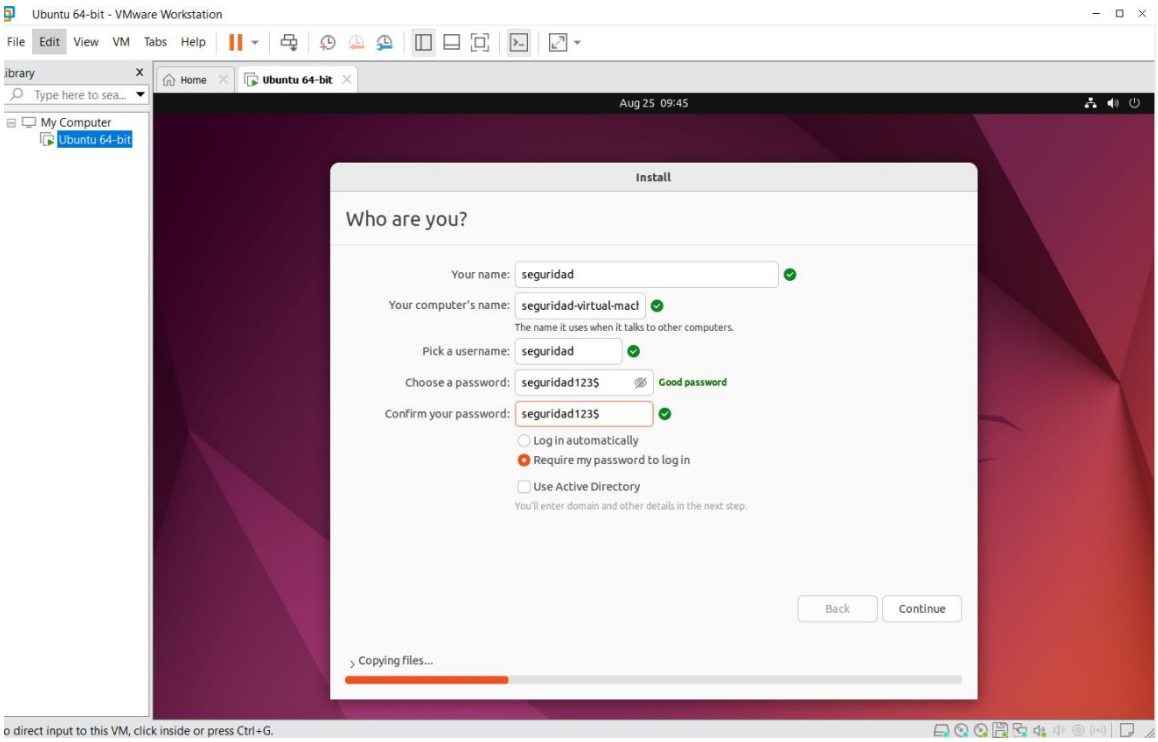
- Seleccionamos la opción “Erase disk and install Ubuntu”



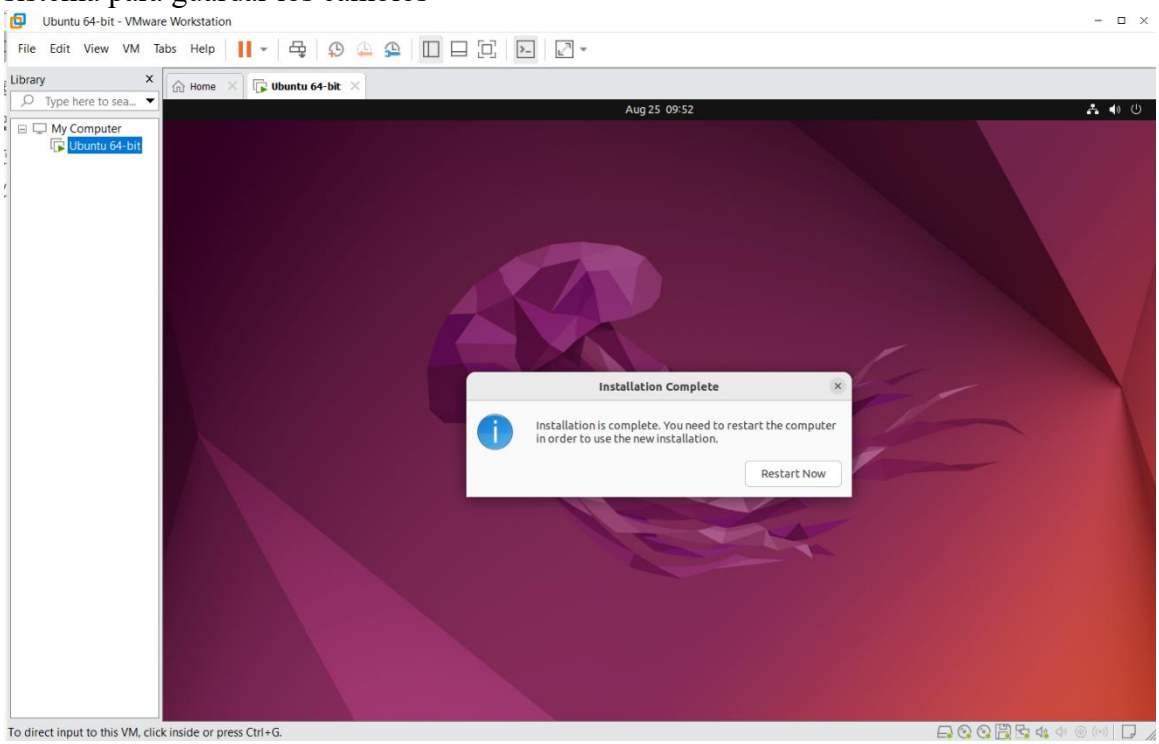
- Seleccionamos la región Colombia Time



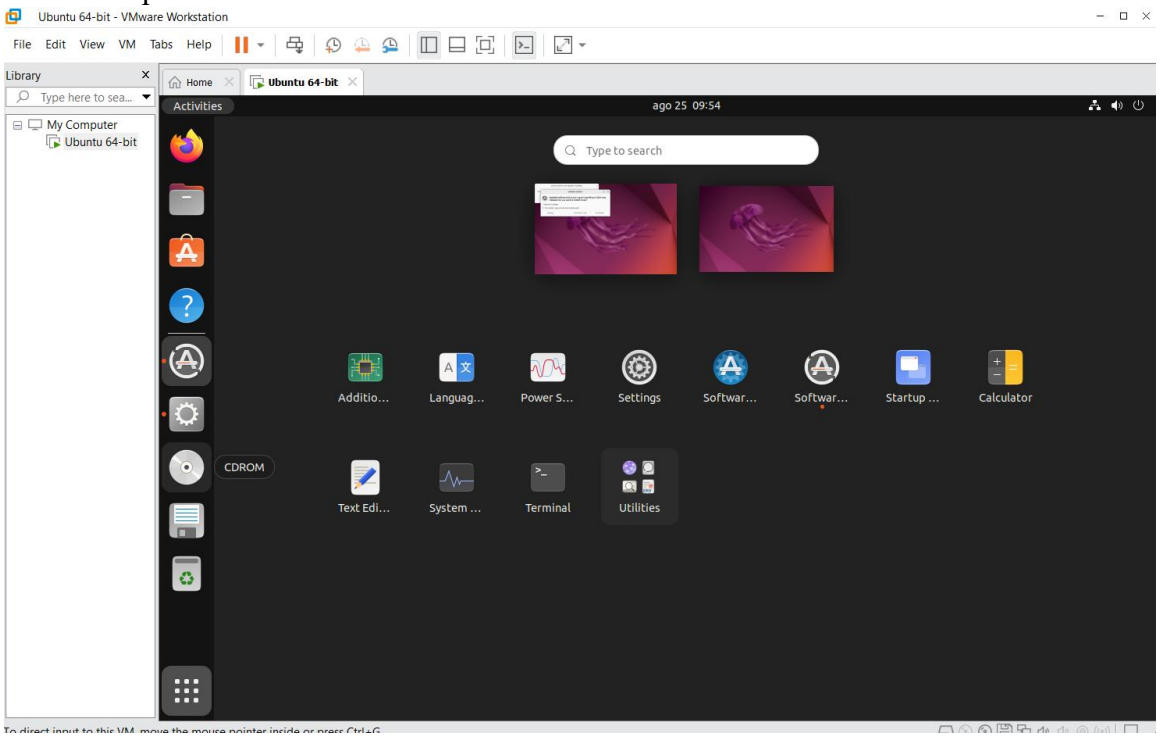
- Digitamos los datos de inicio de sesión, en nuestro caso serán los siguientes:



- Una vez hecho esto, la instalación estaría completa y solo nos queda reiniciar el sistema para guardar los cambios



- Una vez reiniciado el sistema, accedemos al menú y abrimos la terminal de comandos para realizar el laboratorio.



DESARROLLO

Actividad 0 — Preparación del entorno (10 min)

Objetivo: confirmar sistema, privilegios y preparar utilidades.

- Whoami && id
  - Este comando muestra el usuario actual (whoami) y los identificadores de usuarios y grupo (id)

```
seguridad@seguridad-virtual-machine:~$ whoami && id
seguridad
uid=1000(seguridad) gid=1000(seguridad) groups=1000(seguridad),4(adm),24(cdrom),
27(sudo),30(dip),46(plugdev),122(lpadmin),135(lxd),136(sambashare)
```

- uname-a
  - Este comando muestra información del Kernel, arquitectura y compilación del sistema

```
seguridad@seguridad-virtual-machine:~$ uname -a
Linux seguridad-virtual-machine 6.8.0-65-generic #68~22.04.1-Ubuntu SMP PREEMPT_
DYNAMIC Tue Jul 15 18:06:34 UTC 2 x86_64 x86_64 x86_64 GNU/Linux
```

- cat /etc/os-release
  - Este comando muestra los metadatos de la distribución de Linux usada (nombre, versión, id, enlaces)

```
seguridad@seguridad-virtual-machine:~$ cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.5 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.5 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-poli
cy"
UBUNTU_CODENAME=jammy
```

- sudo apt update
  - Este comando actualiza las versiones de los paquetes con el fin de poder instalar las más recientes

```
seguridad@seguridad-virtual-machine:~$ sudo apt update
[sudo] password for seguridad:
Hit:1 http://co.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://co.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://co.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
258 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

- sudo apt install -y htop tree lsb-release file lsof jq
    - Este comando instala las unidades que se necesitan para este laboratorio
        - Htop: monitor de procesos
        - Tree: muestra los directorios de forma jerárquica
        - Lbs-release: permite identificar la distribución
        - File: identifica el tipo de archivo
        - Lsof: lista de archivos abiertos por procesos
        - Jq: procesador de JSON por la línea de comandos

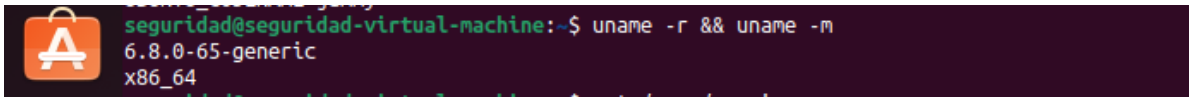
```
seguridad@seguridad-virtual-machine:~$ sudo apt install -y htop tree lsb-release
 file lsof jq
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
lsof is already the newest version (4.93.2+dfsg-1.1build2).
lsof set to manually installed.
file is already the newest version (1:5.41-3ubuntu0.1).
file set to manually installed.
```

Actividad 1 — ¿Qué es Linux? Kernel y módulos (20 min)

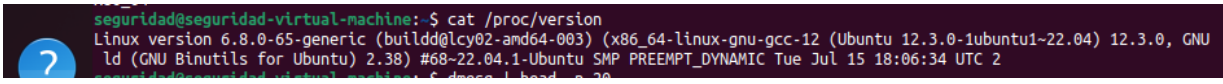
Objetivo: identificar versión del kernel, arquitectura y módulos cargados.

Evidencia: captura con versión del kernel y breve explicación del rol del kernel (3–5 líneas).

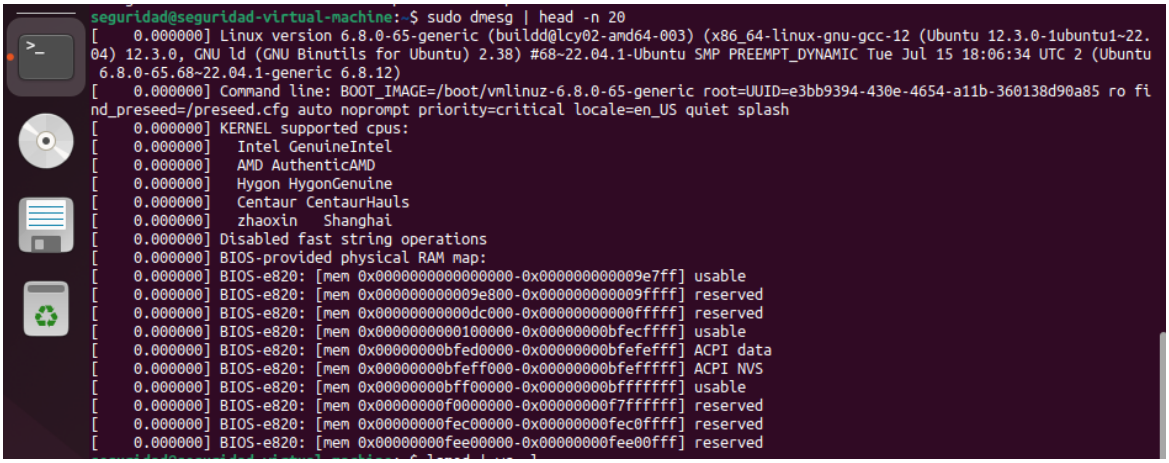
- uname -r && uname -m
    - Este comando muestra la versión del kernel en uso (-r) y la arquictura del procesador (-m)



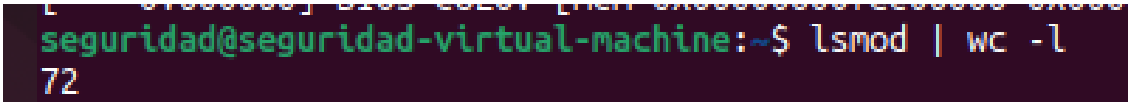
- cat /proc/version
    - Este comando muesta información detallada de la compilación del kernel (versión, compilador usado (gcc) y fecha de construcción



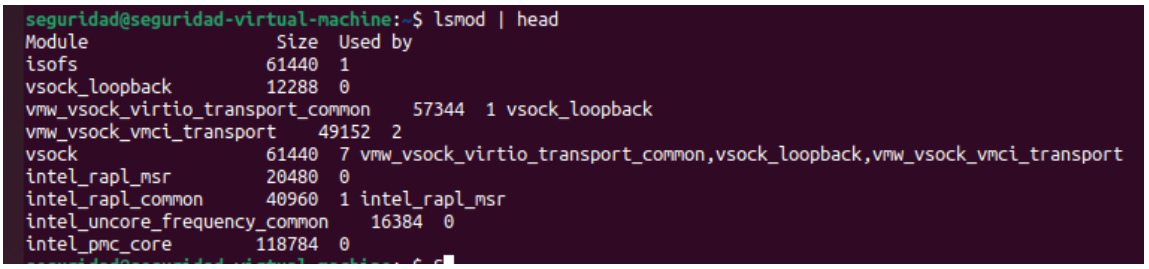
- dmesg | head -n 20
    - Este comando muestra los primeros mensajes del registro de kernel (dmesg), como la inicialización del hardware, memoria y drivers.



- lsmod | wc –l
  - Lista los modulos del kernel y cuenta cuantas lineas hay



- lsmod | head
  - Lista los 10 primeros modulos cargados por el kernel

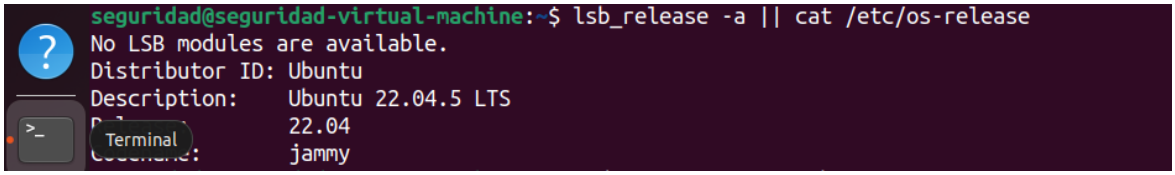


- Reflexión: ¿Por qué el kernel es “el cerebro” del SO y cómo afecta a seguridad/estabilidad?
  - El kernel es considerado “el cerebro” del sistema operativo ya que realiza la gestión de varios recursos fundamentales como la CPU, memoria, seguridad y dispositivos
  - Si el kernel llega a fallar puede afectar la estabilidad del sistema o exponer vulnerabilidades críticas

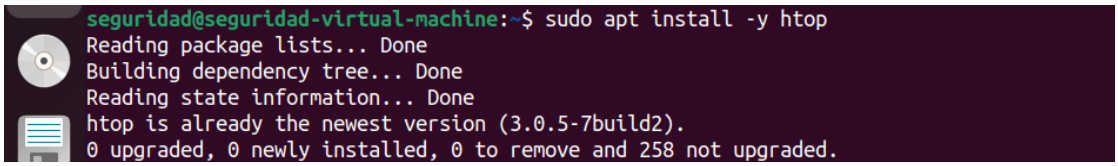
Actividad 2 — Distribución y gestor de paquetes (15 min)

Objetivo: reconocer la distro y explorar archivos instalados por un paquete.
Evidencia: listado parcial de archivos del paquete y explicación del rol del gestor de paquetes.

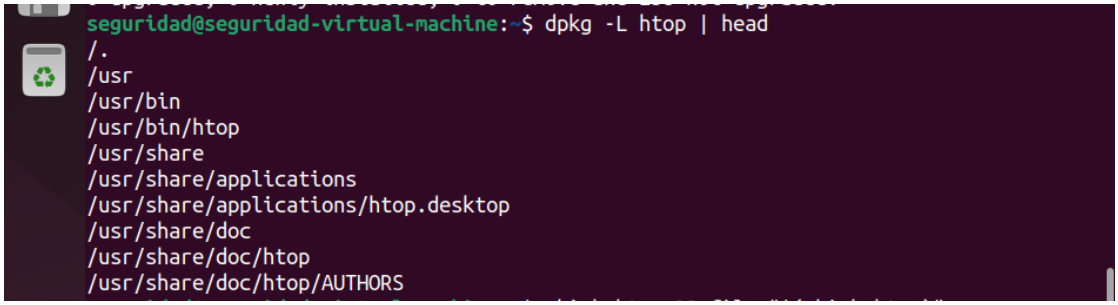
- lsb_release -a || cat /etc/os-release
  - Este comando permite identificar la distribución de Linux en uso



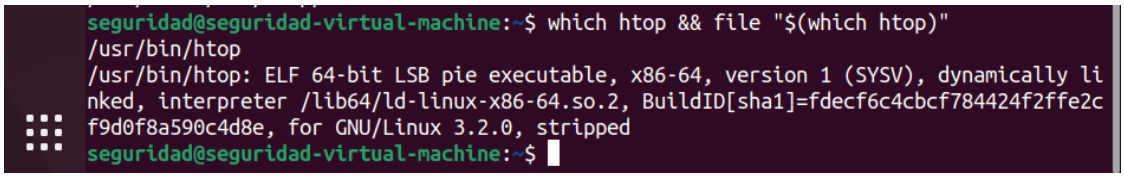
- sudo apt install -y htop
    - Este comando instala el paquete htop desde los repositorios oficiales usando el gestor de paquetes APT



- dpkg -L htop | head
    - Este comando lista los archivos instalados del paquete htop el cual contiene rutas como usr/bin/htop, archivos de documentación y páginas de manual



- which htop && file "$(which htop)"
    - Este comando usa which para localizar la ruta del binario htop (es decir usr/bin/htop) y con file analiza el tipo de archivo mostrando que es un binario de 64 bits



- Reflexión: diferencia entre /bin, /usr/bin y por qué algunas distros unifican rutas.
    - La diferencia principal es que /bin contiene comandos esenciales para el arranque mínimo del sistema y usr/bin guarda la mayoría de los programas y utilidades que no son indispensables para dicho arranque
    - Algunas distribuciones unifican las rutas para simplificar la jerarquía y con esto evitar la duplicación de archivos. Por eso se monta al inicio /usr ya que contiene /urs/bin y /usr/sbin

Actividad 3 — Tour FHS: “todo es un archivo” (45 min)

Objetivo: recorrer directorios clave del FHS y recopilar evidencias.

3.1 /bin y /sbin

Evidencia: librerías de las que depende ls (salida de ldd).

- ls -l /bin | head a
  - Este comando muestra los primeros archivos dentro /bin . En realidad, se ve que bin apunta a /usr/bin

```
Trash dad@seguridad-virtual-machine:~$ ls -l /bin | head
lrwxrwxrwx 1 root root 7 ago 25 10:19 /bin -> usr/bin
```

- ls -l /sbin | head
  - Este comando muestra los primeros archivos dentro /sbin . En realidad, se ve que sbin apunta a /usr/sbin

```
seguridad@seguridad-virtual-machine:~$ ls -l /sbin | head
lrwxrwxrwx 1 root root 8 ago 25 10:19 /sbin -> usr/sbin
```

- which ls; file /bin/ls; ldd /bin/ls
  - Este comando localiza la ruta con el comando which ls, que está en usr/bin/ls, luego que dicho archivo sea ejecutable EFL de 64 bits con file /bin/ls y finalmente lista las bibliotecas compartidas de las que depende como libc.so.6, libselinux.so.1 …

```
seguridad@seguridad-virtual-machine:~$ which ls; file /bin/ls; ldd /bin/ls
/usr/bin/ls
/bin/ls: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked,
interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=36b86f957a1be53733633d184c3a335
4f3fc7b12, for GNU/Linux 3.2.0, stripped
        linux-vdso.so.1 (0x00007ffe4d38d000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007b7867eba000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007b7867c00000)
        libpcre2-8.so.0 => /lib/x86_64-linux-gnu/libpcre2-8.so.0 (0x00007b7867b69000)
        /lib64/ld-linux-x86-64.so.2 (0x00007b7867f19000)
seguridad@seguridad-virtual-machine:~$
```

3.2 /boot

Evidencia: identificar vmlinuz- e initrd.img-* y explicar su función.*

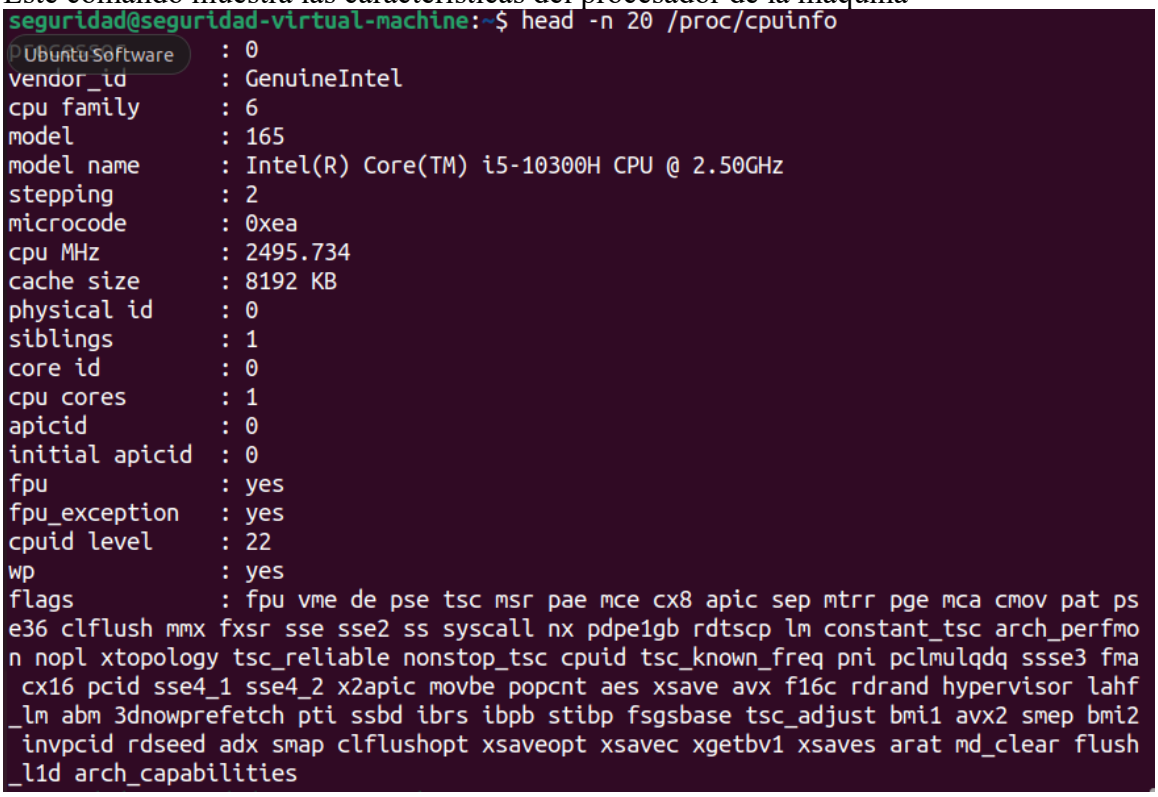
- ls -lh /boot
  - Este comando muestra el directorio /boot el cual contiene los archivos necesarios para arrancar el sistema operativo (vmlinuz-* es la imagen comprimida el kernel e initrd.img-*es la imagen del disco RAM inicial)

```
seguridad@seguridad-virtual-machine:~$ ls -lh /boot
total 184M
-rw-r--r-- 1 root root 281K jul 30  2024 config-6.8.0-40-generic
-rw-r--r-- 1 root root 281K jul 15 11:33 config-6.8.0-65-generic
drwx------ 3 root root 4,0K dic 31  1969 efi
drwxr-xr-x 6 root root 4,0K ago 25 10:26 grub
lrwxrwxrwx 1 root root   27 ago 25 10:26 initrd.img -> initrd.img-6.8.0-65-generic
-rw-r--r-- 1 root root  69M ago 25 10:26 initrd.img-6.8.0-40-generic
-rw-r--r-- 1 root root  69M ago 25 10:26 initrd.img-6.8.0-65-generic
lrwxrwxrwx 1 root root   27 ago 25 10:19 initrd.img.old -> initrd.img-6.8.0-40-generic
-rw-r--r-- 1 root root 179K feb  6  2022 memtest86+.bin
-rw-r--r-- 1 root root 181K feb  6  2022 memtest86+.elf
-rw-r--r-- 1 root root 181K feb  6  2022 memtest86+_multiboot.bin
-rw------- 1 root root 8,3M jul 30  2024 System.map-6.8.0-40-generic
-rw------- 1 root root 8,4M jul 15 11:33 System.map-6.8.0-65-generic
lrwxrwxrwx 1 root root   24 ago 25 10:26 vmlinuz -> vmlinuz-6.8.0-65-generic
-rw-r--r-- 1 root root  15M sep 11  2024 vmlinuz-6.8.0-40-generic
-rw------- 1 root root  15M jul 15 11:36 vmlinuz-6.8.0-65-generic
lrwxrwxrwx 1 root root   24 ago 25 10:26 vmlinuz.old -> vmlinuz-6.8.0-40-generic
```

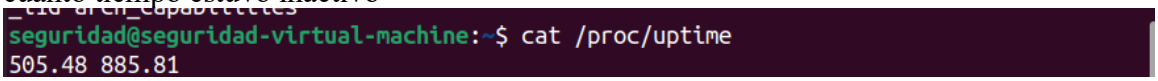
3.3 /proc (virtual)

Evidencia: captura de /proc/self y significado.

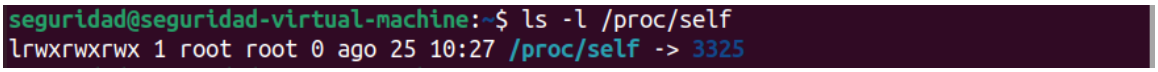
- head -n 20 /proc/cpuinfo
  - Este comando muestra las características del procesador de la maquina



- cat /proc/uptime
  - Este comando indica cuánto tiempo lleva encendido el sistema (segundos) y cuánto tiempo estuvo inactivo



- ls -l /proc/self
  - Este comando muestra a que directorio apunta el proceso que ejecuta el comando (esta información se guarda en el directorio /proc/self)



3.4 /dev (dispositivos)

Evidencia: ¿qué representa /dev/null? Relación con "todo es un archivo".

- ls -l /dev/null /dev/zero
  - Este comando muestra dos archivos.
    - /dev/null (agujero negro) -> todo lo que se escriba acá se descarta
    - /dev/zero -> usado para pruebas o inicializar archivos
  - /dev/null es un archivo especial que descarta todo lo que se escribe, mostrando cómo en Linux "todo es un archivo".

```
seguridad@seguridad-virtual-machine:~$ ls -l /dev/null /dev/zero
crw-rw-rw- 1 root root 1, 3 ago 25 10:27 /dev/null
crw-rw-rw- 1 root root 1, 5 ago 25 10:27 /dev/zero
```

- Lsblk
  - Ese comando lista los dispositivos de bloque (discos, particiones, unidades ópticas, imágenes montadas). Sda es el disco principal y sr0 la unidad CD-ROM virtual

```
seguridad@seguridad-virtual-machine:~$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
fd0      2:0    1     4K  0 disk
loop0    7:0    0  74,3M  1 loop /snap/core22/1612
loop1    7:1    0     4K  1 loop /snap/bare/5
loop2    7:2    0 271,2M  1 loop /snap/firefox/4848
loop3    7:3    0 505,1M  1 loop /snap/gnome-42-2204/176
loop4    7:4    0  91,7M  1 loop /snap/gtk-common-themes/1535
loop5    7:5    0  12,9M  1 loop /snap/snap-store/1113
loop6    7:6    0  38,8M  1 loop /snap/snapd/21759
loop7    7:7    0   500K  1 loop /snap/snapd-desktop-integration/178
sda      8:0    0    40G  0 disk
├─sda1   8:1    0     1M  0 part
├─sda2   8:2    0   513M  0 part /boot/efi
└─sda3   8:3    0  39,5G  0 part /
sr0     11:0    1 156,4M  0 rom  /media/seguridad/CDROM
sr1     11:1    1  1024M  0 rom
seguridad@seguridad-virtual-machine:~$ echo "prueba" > /dev/null
```

- echo "prueba" > /dev/null
  - Este comando envia el texto "prueba" a /dev/null , que lo descarta sin mostrar nada.

```
seguridad@seguridad-virtual-machine:~$ echo "prueba" > /dev/null
seguridad@seguridad-virtual-machine:~$
```

3.5 /etc (configuración del sistema)

Evidencia: propósito de passwd y group (sin exponer datos sensibles).

- ls -l /etc | wc -l head -
  n 5
  - Este comando muestra la cantidad de archivos en /etc el cual contiene la configuración global del sistema (servicios , usuarios, red …)

```
seguridad@seguridad-virtual-machine:~$ ls -l /etc | wc -l
225
```

- head -n 5 /etc/passwd
    - Este comando muestra las primeras líneas del archivo passwd el cual guarda la información básica de cada usuario del sistema (nombre, user id, group id principal, directorio home …)

```
seguridad@seguridad-virtual-machine:~$ head -n 5 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

- head -n 5 /etc/group
    - Este comando muestra las primeras líneas del archivo group el cual define los grupos de usuarios e indica nombre del grupo, id y lista de usuarios que pertenecen a el

```
seguridad@seguridad-virtual-machine:~$ head -n 5 /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,seguridad
seguridad@seguridad-virtual-machine:~$
```

3.6 /lib, /lib32, /lib64 (bibliotecas)

Evidencia: relación binarios ↔ bibliotecas.

- ldd /bin/bash | head
    - Este comando muestra las bibliotecas que necesita bash para poder ejecutarse. Sirve para mostrar la relación entre un programa y las librerías

```
seguridad@seguridad-virtual-machine:~$ ldd /bin/bash | head
        linux-vdso.so.1 (0x00007ffc89114000)
        libtinfo.so.6 => /lib/x86_64-linux-gnu/libtinfo.so.6 (0x00007def638a6000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007def63600000)
        /lib64/ld-linux-x86-64.so.2 (0x00007def63a48000)
```

Trash

3.7 /media y /mnt (montajes)

Evidencia: diferenciar montajes automáticos (/media) vs manuales (/mnt).

- lsblk -f
    - Este comando muestra los dispositivos de bloques del sistema (discos, particiones, usb …) junto con el tipo de sistema de archivos y puntos de montaje

```
seguridad@seguridad-virtual-machine:~$ lsblk -f
NAME FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
fd0
loop0
     squash 4.0                                                    0   100% /snap/core22/1612
loop1
     squash 4.0                                                    0   100% /snap/bare/5
loop2
     squash 4.0                                                    0   100% /snap/firefox/4848
loop3
     squash 4.0                                                    0   100% /snap/gnome-42-2204/176
loop4
     squash 4.0                                                    0   100% /snap/gtk-common-themes/1535
loop5
     squash 4.0                                                    0   100% /snap/snap-store/1113
loop6
     squash 4.0                                                    0   100% /snap/snapd/21759
loop7
     squash 4.0                                                    0   100% /snap/snapd-desktop-integration/178
sda
├─sda1
│
├─sda2
│    vfat   FAT32       F004-C27A                            505,9M     1% /boot/efi
└─sda3
     ext4   1.0         e3bb9394-430e-4654-a11b-360138d90a85     24G    33% /
sr0  iso966       CDROM 2025-08-25-10-17-28-00                   0   100% /media/seguridad/CDROM
sr1
```

- mount | head
    - Este comando lista los archivos actualmente montados y sus opciones (/media se usa para montajes automáticos como una usb y /mnt para montajes manuales, de prueba o temporales)

```
seguridad@seguridad-virtual-machine:~$ mount | head
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=1941760k,nr_inodes=485440,mode=755,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=396140k,mode=755,inode64)
/dev/sda3 on / type ext4 (rw,relatime,errors=remount-ro)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k,inode64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,nsdelegate,memory_recursiveprot)
```

- df -hT | column -t | head
    - Este comando muestra el espacio usado y disponible en cada sistema de archivos junto con el tipo (ext4, tmpfs…)

```
seguridad@seguridad-virtual-machine:~$ df -hT | column -t | head
Filesystem  Type     Size  Used  Avail  Use%  Mounted                on
tmpfs       tmpfs    387M  2,1M  385M   1%    /run
/dev/sda3   ext4     39G   13G   24G    35%   /
tmpfs       tmpfs    1,9G  0     1,9G   0%    /dev/shm
tmpfs       tmpfs    5,0M  4,0K  5,0M   1%    /run/lock
/dev/sda2   vfat     512M  6,1M  506M   2%    /boot/efi
tmpfs       tmpfs    387M  164K  387M   1%    /run/user/1000
/dev/sr0    iso9660  157M  157M  0      100%  /media/seguridad/CDROM
seguridad@seguridad-virtual-machine:~$
```

3.8 /opt y /usr

Evidencia: cuándo usarías /opt vs instalar vía gestor de paquetes.

- sudo tree -L 1 /opt 2>/dev/null || echo "sin /opt"
    - Este comando muestra la estructura del directorio /opt en un solo nivel o imprime “sin /opt” si no existe. Sirve para revisar si hay aplicaciones instaladas fuera del gestor de paquetes
    - Se usa /opt cuando quieres instalar aplicaciones de terceros que no están en los repositorios oficiales y conviene mantener separadas del sistema, mientras que el gestor de paquetes se usa para instalar programas oficiales del sistema operativo, ya que maneja dependencias y actualizaciones de

forma automática y segura.

```
seguridad@seguridad-virtual-machine:~$ sudo tree -L 1 /opt 2>/dev/null || echo "sin /opt"
/opt

0 directories, 0 files
```

- du -sh /usr/bin | awk '{print $1}'
    - Este comando calcula el tamaño total ocupado por usr/bin y lo muestra en formato (MB)

```
seguridad@seguridad-virtual-machine:~$ du -sh /usr/bin | awk '{print $1}'
156M
seguridad@seguridad-virtual-machine:~$
```


3.9 /run, /sys, /srv

Evidencia: MAC de una interfaz y explicación de /run como tmpfs.

- mount | grep " /run "
    - Este comando verifica como está montado el directorio /run que en este caso es un tmpfs (se borra al reiniciar). Se usa para archivos temporales de procesos en ejecución
    - /run es un directorio temporal montado en tmpfs, usado por procesos y servicios del sistema; todo lo que guarda se borra al reiniciar.

```
seguridad@seguridad-virtual-machine:~$ mount | grep " /run "
tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=396140k,mode=755,inode64)
```

- ls -l /sys/class/net
    - Este comando lista las interfaces de red detectadas en el sistema dentro de /sys. Sirve para explorar los parámetros del hardware de red

```
seguridad@seguridad-virtual-machine:~$ ls -l /sys/class/net
total 0
lrwxrwxrwx 1 root root 0 ago 25 10:27 ens33 -> ../../devices/pci0000:00/0000:00:11.0/0000:02:01.
0/net/ens33
lrwxrwxrwx 1 root root 0 ago 25 10:27 lo -> ../../devices/virtual/net/lo
```

- cat /sys/class/net/*/address
    - Este comando muestra las direcciones de MAC de cada interfaz de red
    - MAC es la dirección física única de cada interfaz de red, usada para identificarla a nivel de hardware en la red.

```
seguridad@seguridad-virtual-machine:~$ cat /sys/class/net/*/address
00:0c:29:37:97:f3
00:00:00:00:00:00
seguridad@seguridad-virtual-machine:~$ s
```

3.10 /tmp (sticky) y /var, /home

Evidencia: interpretación de permisos de /tmp (bit t).

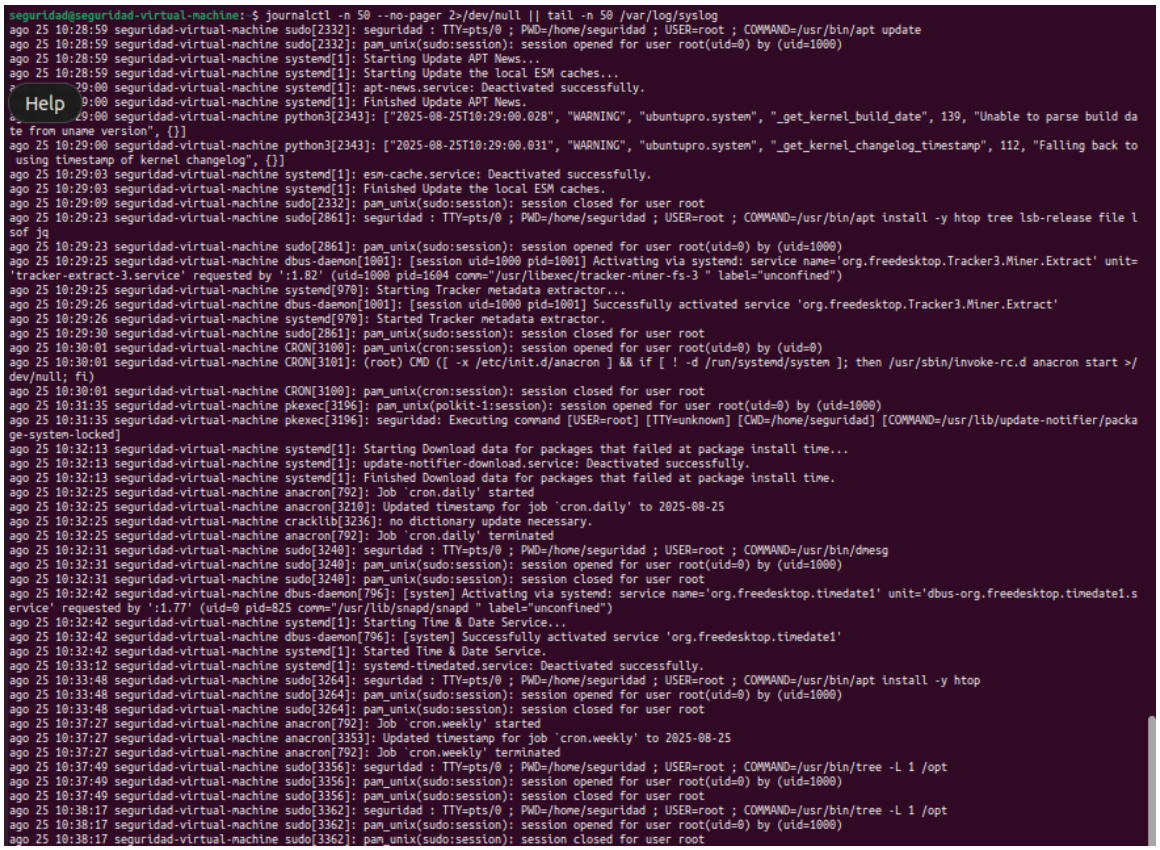
- ls -ld /tmp
    - Este comando muestra los permisos del directorio /tmp el cual contiene una t, la cual hace referencia a que todos pueden escribir, pero solo el dueño puede borrar sus archivos

```
seguridad@seguridad-virtual-machine:~$ ls -ld /tmp
drwxrwxrwt 20 root root 4096 ago 25 10:37 /tmp
```

- echo $UMASK || umask
    - Este comando muestra el valor de umask actual, que define los permisos por defecto con los que se crean nuevos archivos y directorios. Umask no esta definida por el momento

```
seguridad@seguridad-virtual-machine:~$ echo $UMASK || umask
```

- journalctl -n 50 --no-pager 2>/dev/null || tail -n 50 /var/log/syslog
    - Este comando muestra las ultimas 50 entradas del log del sistema. Sirve para revisar eventos que sucedieron recientemente.



- echo "hola" > ~/demo.txt && ls -l ~/demo.txt
    - Este comando crea un directorio personal y lista sus permisos. Esto sirve para confirmar como se aplican los permisos por defecto en archivos de usuario

```
seguridad@seguridad-virtual-machine:~$ echo "hola" > ~/demo.txt && ls -l ~/demo.txt
-rw-rw-r-- 1 seguridad seguridad 5 ago 25 10:40 /home/seguridad/demo.txt
seguridad@seguridad-virtual-machine:~$ s
```

- Reflexión: ¿Qué directorios no persisten tras reiniciar y por qué?
    - Los directorios que no persisten al reiniciar son /tmp, /run y algunas partes de /sys ya que se montan en memoria como tmpfs, es decir que su contenido se borra al apagar el sistema ya que solo guardan datos temporales necesarios para procesos en ejecución.

Actividad 4 — Rutas, comodines y utilidades de texto (25 min)

Objetivo: practicar rutas absolutas/relativas, globbing y comparación de texto.
*Evidencias: expansión con * y manejo de espacios; salida de diff comentada.*

- Pwd
    - Este comando muestra la ruta en el directorio actual

```
seguridad@seguridad-virtual-machine:~$ pwd
/home/seguridad
```

- mkdir -p ~/lab/fotos/2025
    - Este comando crea un directorio en la ruta indicada, con la opción p se indica que, si las carpetas intermedias no existen, también se creen

```
seguridad@seguridad-virtual-machine:~$ mkdir -p ~/lab/fotos/2025
```

- cd ~/lab/fotos/2025 && touch "foto 1.png" foto2.png notas.txt
    - cd cambia al directorio recién creado, && indica que lo siguiente solo se ejecuta si el primer comando fue exitoso y touch crea archivos vacíos con los nombres indicados, foto1, foto2 y notas

```
seguridad@seguridad-virtual-machine:~$ cd ~/lab/fotos/2025 && touch "foto 1.png" foto2.png notas
.txt
```

- ls -l ~/lab/fotos/2025
    - Este comando lista el directorio anterior y sus detalles (permisos, dueño, grupo, tamaño y fecha de modificación)

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls -l ~/lab/fotos/2025
total 0
-rw-rw-r-- 1 seguridad seguridad 0 ago 25 10:40 'foto 1.png'
-rw-rw-r-- 1 seguridad seguridad 0 ago 25 10:40  foto2.png
-rw-rw-r-- 1 seguridad seguridad 0 ago 25 10:40  notas.txt
```

- ls ~/lab/fotos/2025/foto*.png
    - Este comando lista los archivos que comiencen con foto y terminen en .png (uso de *)

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls ~/lab/fotos/2025/foto*.png
'/home/seguridad/lab/fotos/2025/foto 1.png'   /home/seguridad/lab/fotos/2025/foto2.png
```

- ls ~/lab/fotos/2025/foto\ 1.png
    - Este comando listo específicamente el archivo foto1.png. El \ permite escribir nombres de archivos con espacios

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls ~/lab/fotos/2025/foto\ 1.png
'/home/seguridad/lab/fotos/2025/foto 1.png'
```

- head -n 3 notas.txt
    - Este comando muestra las 3 primeras líneas del archivo notas.txt

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ head -n 3 notas.txt
```

- echo "linea A" >> notas.txt
    - Este comando escribe en el archivo notas.txt el texto “línea A”. Si el archivo no existe lo crea

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ echo "linea A" >> notas.txt
```

- echo "linea B" > notas_nueva.txt
    - Este comando escribe en el archivo notas_nueva.txt el texto “línea B”. Si el archivo no existe lo crea

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ echo "linea B" > notas_nueva.txt
```

- diff -u notas.txt notas_nueva.txt.
    - Este comando compara los dos archivos línea por línea y muestra sus diferencias
    - La salida de diff indica que notas.txt contiene “línea A” mientras que notas_nueva.txt contiene “línea B”, mostrando así la diferencia entre ambos archivos.

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ diff -u notas.txt notas_nueva.txt
--- notas.txt   2025-08-25 10:41:21.071945796 -0500
+++ notas_nueva.txt     2025-08-25 10:41:26.766576184 -0500
@@ -1 +1 @@
-linea A
+linea B
```

Actividad 5 — Permisos, ownership, umask y setuid (20 min)

Objetivo: comprender permisos y sus implicaciones de seguridad.
Evidencias: capturas antes/después; explicación de riesgos del setuid

- mkdir -p ~/lab/comp
    - Este comando crea el directorio comp dentro de lab

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ mkdir -p ~/lab/comp
```

- touch ~/lab/comp/doc.txt
    - Este comando crea un archivo vacio doc.txt dentro del directorio comp

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ touch ~/lab/comp/doc.txt
```

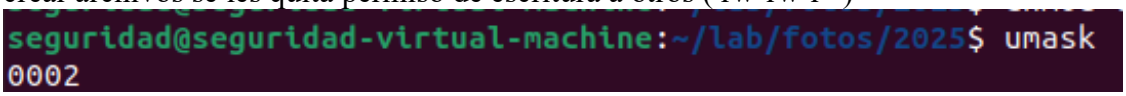
- ls -l ~/lab/comp/doc.txt
    - Este comando lista el archivo doc.txt, mostrando permisos, propietario, grupo, tamaño y fecha de modificación

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls -l ~/lab/comp/doc.txt
-rw-rw-r-- 1 seguridad seguridad 0 ago 25 10:42 /home/seguridad/lab/comp/doc.txt
```

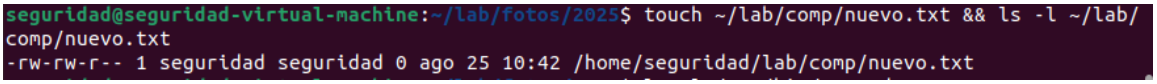
- chmod 640 ~/lab/comp/doc.txt
    - Este comando cambia los permisos de doc.txt a:
        - 6(rw-) -> dueño puede leer y escribir
        - 4(r--) -> grupo puede leer
        - 0 (---) -> otros no tienen permisos

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ chmod 640 ~/lab/comp/doc.txt
```

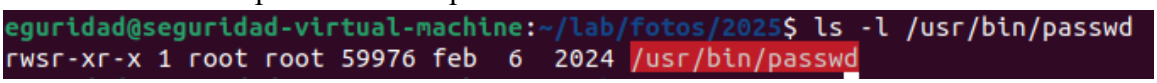
- umask
  - Este comando muestra la máscara de permisos por defecto que se aplica cuando se crean nuevos archivos o directorios. 0002 hace referencia que al crear archivos se les quita permiso de escritura a otros (rw-rw-r--)



- touch ~/lab/comp/nuevo.txt && ls -l ~/lab/comp/nuevo.txt
  - Este comando crea un archivo nuevo.txt y lo lista para ver que permisos heredó según umask



- ls -l /usr/bin/passwd
  - Este comando listo passwd el cual permite cambiar contraseñas.



- Pregunta: ¿qué implica la s en permisos y por qué es sensible? Evidencias:
  - La s indica que el archivo se ejecuta con los permisos del dueño del archivo en lugar de los del usuario que lo ejecuta
  - Es sensible ya que, si se aplica a un programa inseguro o malicioso, podría usarse para tomar privilegios y obtener control total del sistema

Actividad 6 — Montaje loopback en /mnt (15 min)

Objetivo: demostrar montaje de imagen y diferenciar /mnt vs /media.
Evidencias: salida de df -hT mostrando el montaje; listado de /mnt/lab con el archivo creado.

- dd if=/dev/zero of=~/lab.img bs=1M count=64 mkfs. ext4 ~/lab.img
  - dd copia datos a bajo nivel
  - if=/dev/zero usa como entrada un flujo de ceros
  - of= ~/lab.img crea un archivo de salida de 64MB
  - bs =1M count=64 define el tamaño: 1MB por bloque x 6 bloques = 64MB

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ dd if=/dev/zero of=~/lab.img bs=1M count=6
4
64+0 records in
64+0 records out
67108864 bytes (67 MB, 64 MiB) copied, 0,0714287 s, 940 MB/s
```

- mkfs. ext4 ~/lab.img
  - Este comando formatea el archivo lab.img con el sistema de archivos ext4, convirtiéndolo en una imagen lista para ser montada

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ mkfs.ext4 ~/lab.img
mke2fs 1.46.5 (30-Dec-2021)
Discarding device blocks: done
Creating filesystem with 16384 4k blocks and 16384 inodes

Allocating group tables: done
Writing inode tables: done
Creating journal (1024 blocks): done
Writing superblocks and filesystem accounting information: done
```

- sudo mkdir -p /mnt/lab
  - Este comando crea el directorio /mnt/lab, como si fuera un dispositivo real

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ sudo mkdir -p /mnt/lab
```

- sudo mount -o loop ~/lab.img /mnt/lab
  - Este comando monta el archivo lab.img y la opción -o loop permite tratar a un archivo como un disco

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ sudo mount -o loop ~/lab.img /mnt/lab
```

- df -hT | grep "/mnt/lab"
  - Este comando muestra la información de sistemas de archivos montados. El grep filtra la salida para que solo se muestre /mnt/lab

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ df -hT | grep "/mnt/lab"
/dev/loop8     ext4      56M   24K   52M   1% /mnt/lab
```

- sudo sh -c 'echo "hola-fhs" > /mnt/lab/ejemplo.txt'
  - Este comando crea dentro del montaje un archivo ejemplo.txt con el texto “hola-fhs”

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ sudo sh -c 'echo "hola-fhs" > /mnt/lab/eje
mplo.txt'
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls -l /mnt/lab
total 20
-rw-r--r-- 1 root root     9 ago 25 10:43 ejemplo.txt
drwx------ 2 root root 16384 ago 25 10:43 lost+found
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ sudo umount /mnt/lab
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ S
```

- ls -l /mnt/lab

- Este comando lista los archivos en el punto de montaje

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ ls -l /mnt/lab
total 20
-rw-r--r-- 1 root root     9 ago 25 10:43 ejemplo.txt
drwx------ 2 root root 16384 ago 25 10:43 lost+found
```

- sudo umount /mnt/lab
  - Este comando desmonta el sistema de archivos /mnt/lab, liberando el archivo lab.img y cierra el montaje

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ sudo umount /mnt/lab
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ S
```

- Reflexión: ventajas de loop devices para prácticas y riesgos.
  - Los dispositivos loop permiten usar archivos como si fueran discos, lo que facilita hacer pruebas sin necesidad de hardware extra. El riesgo está en montarlos con permisos inseguros o no desmontarlos, ya que podría exponer datos o dejar el sistema vulnerable.

Actividad 7 (opcional) — Script de inventario del sistema (20 min)

Objetivo: generar un reporte rápido del sistema y añadirlo al PATH.
RESULTADO:

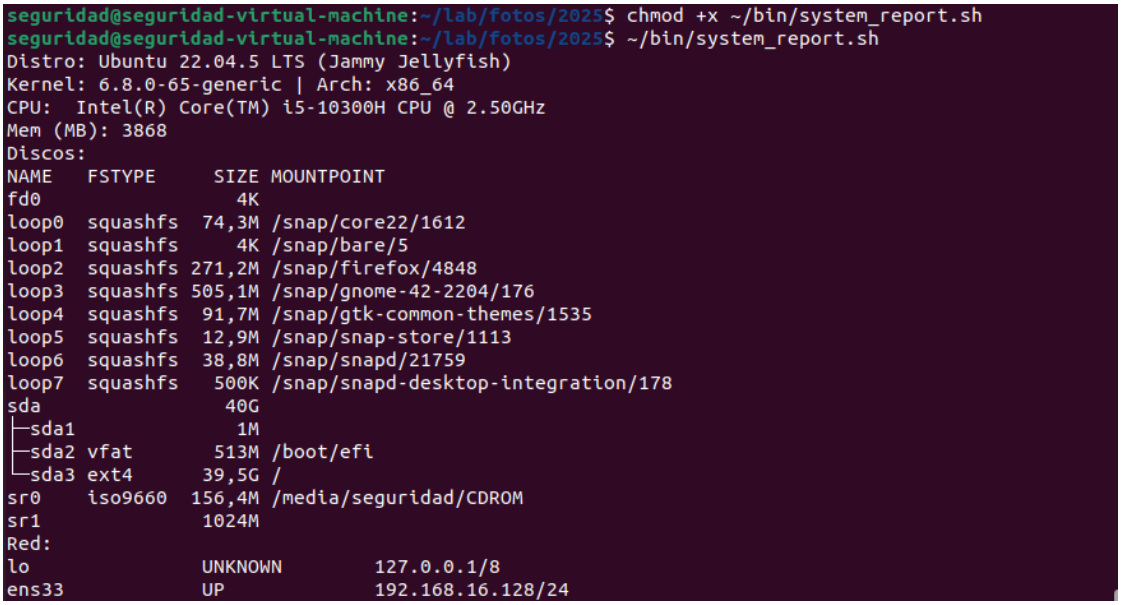
- Estos comandos crean el directorio /bin dentro del home usuario y abre el editor para crear el archivo system_report.sh

```
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ mkdir -p ~/bin
seguridad@seguridad-virtual-machine:~/lab/fotos/2025$ nano ~/bin/system_report.sh
```

- Dentro del archivo se escribe el script que contiene:
  - Distribución (/etc/os-release)
  - Kernel y arquitectura (uname -r, uname -m)
  - CPU (extraída de /proc/cpuinfo)
  - Memoria RAM en MB (free -m)
  - Discos (con lsblk)
  - Interfaces de red (con ip addr)

```
 GNU nano 6.2                  /home/seguridad/bin/system_report.sh
#!/usr/bin/env bash
echo "Distro: $(. /etc/os-release; echo $NAME $VERSION)"
echo "Kernel: $(uname -r) | Arch: $(uname -m)"
echo "CPU: $(grep -m1 'model name' /proc/cpuinfo | cut -d: -f2-)"
echo "Mem (MB): $(free -m | awk '/Mem:/ {print $2}')"
echo "Discos:"
lsblk -o NAME,FSTYPE,SIZE,MOUNTPOINT
echo "Red:"
ip -4 -br addr 2>/dev/null
```

- chmod +x ~/bin/system_report.sh
    - Se le da permiso de ejecución al script.
- system_report.sh
    - Se ejecuta el script creado.
    - El resultado muestra:
        - Nombre y versión de la distribución
        - Version del Kernel y arquitectura
        - Modelo de la CPU
        - Cantidad de memoria RAM
        - Discos y sus puntos de montaje
        - Interfaces de red configuradas



Preguntas de cierre

1) Explica la diferencia entre /bin, /usr/bin y /sbin, y por qué existen.

- /bin = contiene los comandos básicos y esenciales del sistema que se necesitan para arrancar y reparar este
- /usr/sbin = guarda la mayoría de los programas de usuario, utilidades adicionales y comandos no críticos para el arranque del sistema
- /sbin = incluye administración del sistema (mount, shutdown, ifconfig), normalmente usados por el administrador
- Existen porque la separación garantiza que el sistema pueda arrancar y repararse incluso si /usr no está disponible

2) ¿Qué directorios del FHS suelen estar en tmpfs y por qué? Riesgos/beneficios.

- Suelen estar directorios como /run, /tmp …
- Estos directorios suelen estar montados en tmpfs para que el acceso sea más rápido y porque contienen datos temporales
- Beneficios = mayor rendimiento y los datos se borran al reiniciar, evitando la acumulación

- Riesgos = Todo se pierde al reiniciar y si se llena puede afectar procesos que dependen de /run, /tmp …

3) ¿Qué aprendiste sobre permisos y su relación con la seguridad del sistema? Ejemplos.

- Los permisos controlan quien puede leer, escribir o ejecutar un archivo o directorio
- Ejemplo
    - Chmod 640 archivo.txt -> solo el dueño lee/escribe, el grupo solo lee y otros no tienen permisos
- El setuid permite a un usuario ejecutar el programa con privilegios de root (útil pero riesgoso si un atacante la explota)

4) ¿Por qué “todo es un archivo” facilita la observabilidad (/proc, /sys)?

- Porque en Linux el kernel expone la información del sistema como si fueran archivos de texto, por lo que, en lugar de usar programas especiales, basta con leer archivos en /proc o /sys para saber que sucede

- /proc/cpunfo -> información del procesador
- /proc/uptime -> tiempo encedido
- /sys/class/net/*/address -> MAC de las tarjetas de red

Conclusiones

- Se comprobó que Linux se organiza bajo una jerarquía clara de directorios (FHS) y que el principio de "todo es un archivo" facilita entender cómo se gestionan tanto los archivos reales como los dispositivos y procesos del sistema.

- Al manipular rutas absolutas y relativas, así como comodines y utilidades de texto, se reforzó la importancia de la terminal como herramienta principal de administración y automatización.

- El trabajo con permisos, ownership, umask y setuid evidenció el rol que tienen en la seguridad, mostrando cómo pequeños detalles en la configuración pueden marcar la diferencia entre un sistema seguro y uno vulnerable.

- La exploración de `/media` y `/mnt` permitió diferenciar los montajes automáticos frente a los manuales, y el uso de imágenes montadas en loopback resultó práctico para pruebas sin afectar discos reales.

- Finalmente, se entendió que directorios como `/proc`, `/run` y `/tmp` son temporales o virtuales y que no persisten tras reiniciar, lo cual ayuda a comprender la dinámica del sistema y sus recursos en tiempo real.

- Conocer la estructura interna de Linux no solo ayuda a usarlo mejor, sino que también es clave para administrar servidores, aplicar buenas prácticas de seguridad y resolver problemas de forma más eficiente.