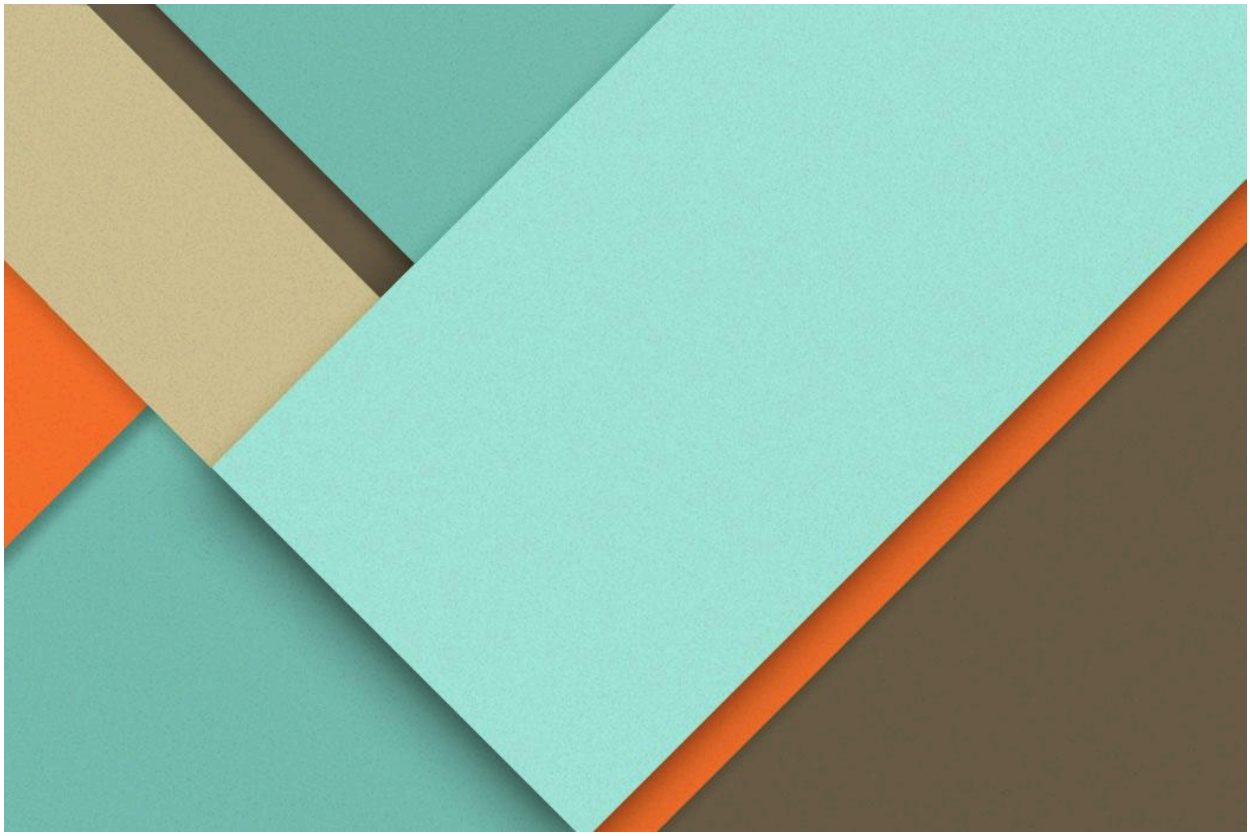


# 4. Seguridad en el arranque del SI. Cifrado. Borrado seguro

24/11/2025

Ramón Romero Montilla

# Actividad 1. Particionamiento de discos. Filesystem Linux vs Filesystem Windows. Lectura de las diapositivas de la Uniovi Hardening del SO general Seguridad en el proceso de arranque.

El proceso de arranque es una secuencia crítica que inicia con la BIOS/UEFI, pasa al gestor de arranque (como GRUB) y finaliza cargando el kernel y los servicios del espacio de usuario mediante systemd. Es fundamental proteger esta fase porque, si un atacante tiene acceso físico, puede interrumpir el inicio para acceder al "Single User Mode" o modo de mantenimiento, lo que le otorgaría privilegios de root sin necesidad de contraseña. Para evitar que modifiquen parámetros del kernel o reseteen claves, es obligatorio configurar una contraseña robusta en el bootloader, bloqueando así la edición de entradas de arranque no autorizadas.

El bastionado del almacenamiento se basa en separar los sistemas de archivos críticos en particiones físicas o lógicas independientes, evitando tener todo el sistema en una sola partición raíz. Esta segregación de directorios clave como /home, /var, /tmp y /var/log/audit es vital para evitar que un desbordamiento de datos (por ejemplo, logs infinitos) llene el disco entero y colapse el sistema operativo por falta de espacio. Además, aislar estas particiones contiene la corrupción de datos en caso de fallo y permite usar sistemas de ficheros optimizados o contextos de seguridad específicos para cada tipo de información.

La mayor ventaja de particionar es poder aplicar opciones de montaje restrictivas (nodev, nosuid, noexec) en directorios donde los usuarios tienen permisos de escritura, como /tmp o /var/tmp. Estas opciones impiden la ejecución de binarios maliciosos y bloquean la creación de dispositivos de caracteres o archivos con permisos heredados (setuid) que facilitan la escalada de privilegios por parte de atacantes.

## Actividad 2. Proteger el acceso de arranque en Windows. La amenaza: los rootkits. Realiza una tabla con los tipos de rootkits que se cargan durante el inicio y las contramedidas (bastionado)

| Tipo de Amenaza / Rootkit | Fase de Carga | Contramedida (Bastionado) |
|---|---|---|
| **Bootkit** (Infección en Firmware/UEFI) | Antes de cargar el Sistema Operativo | **Secure Boot (UEFI):** Impide que se ejecuten gestores de arranque (bootloaders) o sistemas operativos que no estén firmados digitalmente por una entidad de confianza. |
| **Kernel-mode Rootkit** (Drivers maliciosos) | Durante la carga del Kernel y Drivers del Sistema | **Trusted Boot:** Verifica la integridad del kernel de Windows y los archivos de inicio antes de cargarlos.<br><br>**ELAM (Early Launch Anti-Malware):** Carga el driver del antivirus antes que los drivers de terceros para poder inspeccionarlos y bloquearlos si son maliciosos[3]. |

| | | |
|---|---|---|
| **Rootkits ocultos / Persistentes** | Inicio de sesión y ejecución del sistema | **Measured Boot:** Mide cada componente de arranque y envía los datos a un servidor remoto (attestation) para verificar la salud del equipo si el antimalware local ha sido comprometido<br><br>**Anti-Malware:** Escaneo tradicional una vez el sistema está arriba. |

## Actividad 3. Lectura de las diapositivas de la Uniovi Hardening del SO general Seguridad en el proceso de arranque. Explica el proceso de arranque de un SO. ¿Qué es el modo de usuario único (singler use mode)? Desde el punto de vista del bastionado, ¿por qué debemos evitar el acceso a este modo? Anota las fuentes consultadas.

A) Explicación del Proceso de Arranque (Linux)

Basándonos en el diagrama de *ByteByteGo*, el arranque de un sistema operativo Linux sigue una secuencia lógica de **8 pasos** fundamentales desde que pulsas el botón hasta que puedes usar la máquina:

1. **Power On:** Se enciende el equipo.

2. **BIOS/UEFI y POST:** El sistema carga el firmware básico desde la memoria no volátil y ejecuta el *Power-On Self-Test* (POST) para comprobar que el hardware (RAM, CPU) funciona correctamente.

3. **Detección de Dispositivos:** La BIOS/UEFI detecta los discos y periféricos conectados para decidir desde cuál arrancar (Disco Duro, USB, CD).

4. **Boot Loader (GRUB):** Se carga el gestor de arranque (normalmente GRUB en Linux). Este lee su archivo de configuración (/etc/grub2.cfg) para saber qué sistema operativo lanzar.

5. **Carga del Kernel:** El GRUB ejecuta el kernel de Linux y carga las librerías necesarias en memoria.

6. **Systemd (User Space):** El kernel cede el control al primer proceso del espacio de usuario, que suele ser systemd (con PID 1). Este se encarga de gestionar todo el resto del arranque.

7. **Ejecución de Targets:** Systemd lee los archivos .target y ejecuta los scripts de inicio necesarios (como montar sistemas de ficheros, activar red, interfaz gráfica, etc.).

8. **Login:** Finalmente, el sistema presenta la pantalla de inicio de sesión para que el usuario entre.

B) ¿Qué es el Modo de Usuario Único (Single User Mode)?

El "Single User Mode" (o modo de rescate/mantenimiento) es un nivel de ejecución especial del sistema operativo diseñado para tareas administrativas críticas o recuperación ante desastres.

- **Características:** Arranca el sistema con lo mínimo indispensable, montando los discos a veces en modo solo lectura y **sin activar los servicios de red** ni la interfaz gráfica.
- **Propósito:** Se utiliza cuando el sistema detecta un fallo grave al arrancar o cuando el administrador necesita reparar el sistema de ficheros, resetear contraseñas olvidadas o arreglar configuraciones rotas.

C) ¿Por qué debemos evitar el acceso libre a este modo (Bastionado)?

Porque el acceso no restringido al *Single User Mode* es una vulnerabilidad crítica por:

1. **Acceso Root sin contraseña:** Por defecto, muchas configuraciones permiten entrar en este modo simplemente editando una línea en el menú del GRUB durante el arranque. Al entrar, el sistema te otorga **privilegios de superusuario (root)** sin pedirte ninguna contraseña.

2. **Riesgo de Acceso Físico:** Si un atacante tiene acceso físico a tu máquina (o a la consola virtual), puede reiniciar el equipo, entrar en este modo y cambiar la contraseña de root, robar datos o modificar el sistema a su antojo, saltándose todos los controles de acceso habituales.

Para mitigar esto, es obligatorio configurar una contraseña en el gestor de arranque (GRUB) para que nadie pueda modificar los parámetros de inicio sin autorización.

Fuentes:

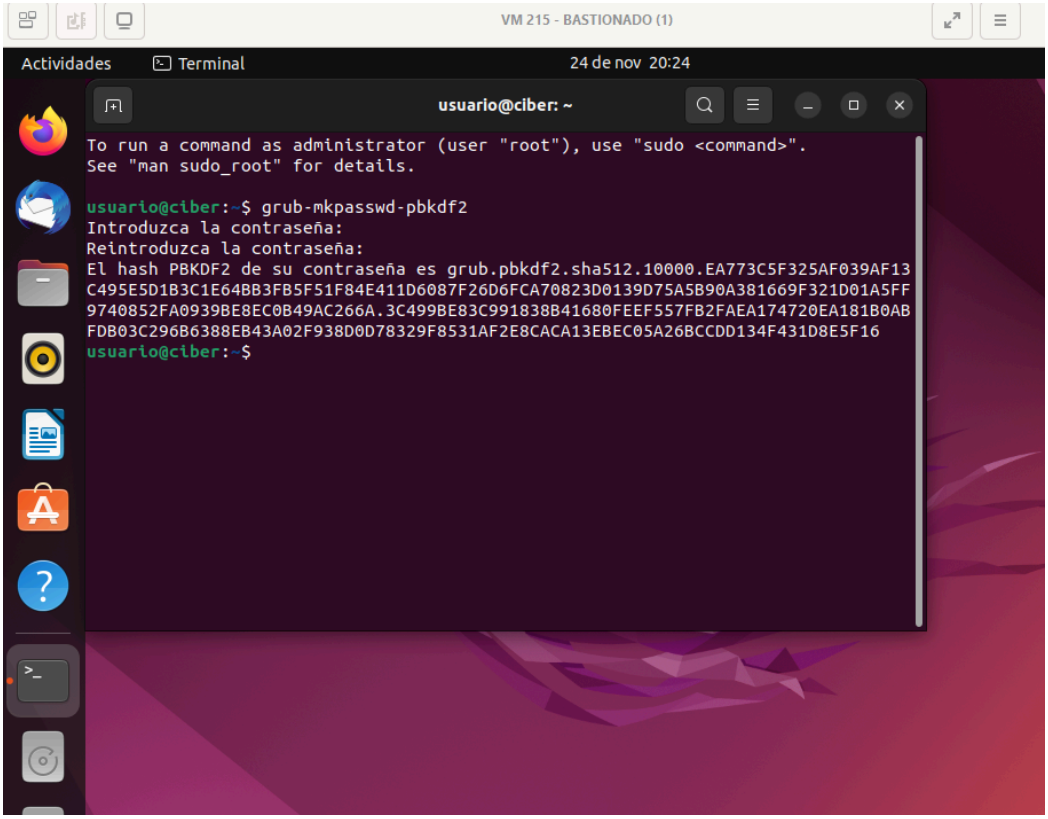
**Red Hat. (s.f.).** *Capítulo 30: Proceso de arranque, inicio y cierre del sistema*. Red Hat Enterprise Linux Documentation

.https://docs.redhat.com/es/documentation/red_hat_enterprise_linux/5/html/installation_guide/ch-boot-init-shutdown

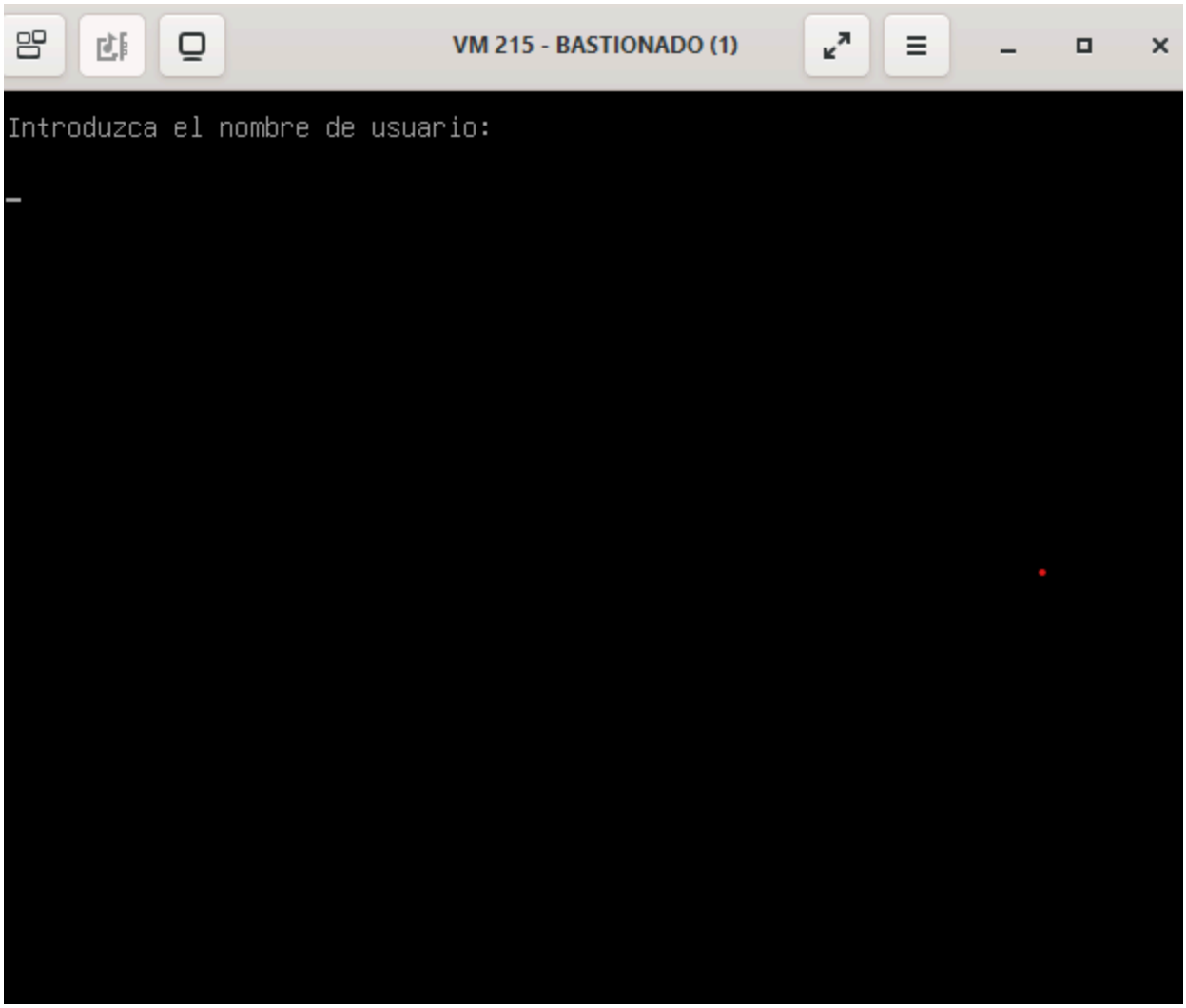
**INCIBE-CERT. (2023, 2 de marzo).** *Bastionado de sistemas: el caso de Linux*. Instituto Nacional de Ciberseguridad. https://www.incibe.es/incibe-cert/blog/bastionado-sistemas-el-caso-linux

## Actividad 4. Establecer un password de arranque / al bootloader (gestor de arranque). (Univoi. ejercicio L5B1_SINGLEUSER) Configuración del arranque con la ayuda del Vídeo Hardening básico de Linux. Incibe. minuto (4:45 – 07.20) y añade las fuentes consultadas.

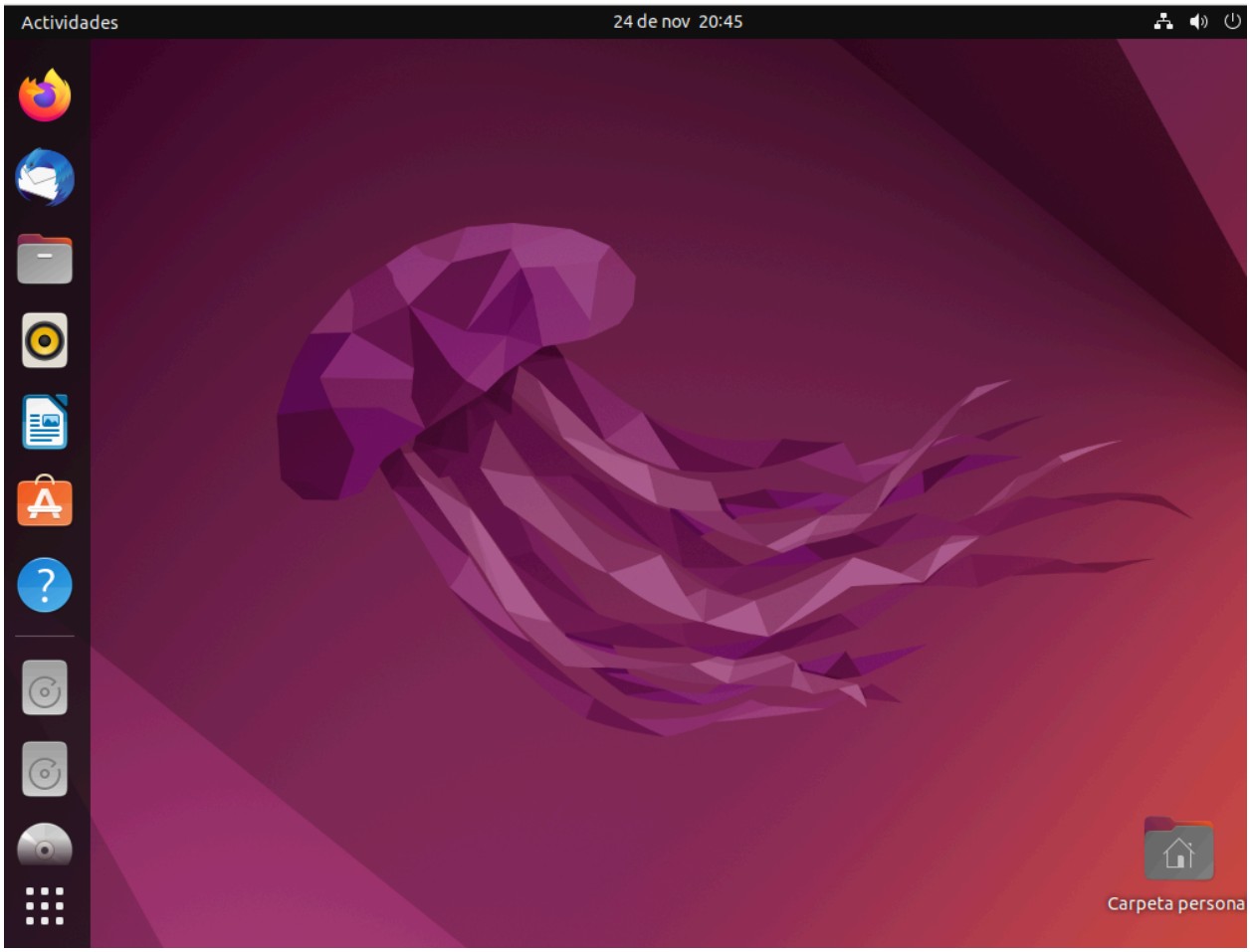
Para realizar el bastionado del arranque lo primero que hice fue abrir una terminal en mi máquina virtual de Ubuntu y ejecutar el comando grub-mkpasswd-pbkdf2 para cifrar mi contraseña de seguridad, tras introducirla dos veces el sistema me generó una cadena hash PBKDF2 muy larga que es la que copié para configurar posteriormente el archivo de usuarios del gestor de arranque tal y como se ve en la primera captura de pantalla con la terminal.



Una vez aplicada la configuración y actualizado el GRUB reinicié el sistema para comprobar la efectividad de la medida y como se puede observar en la segunda imagen con el fondo negro el proceso de arranque se detuvo automáticamente solicitándome un nombre de usuario y contraseña antes de cargar nada, lo que demuestra que he bloqueado el acceso no autorizado y la edición de parámetros de arranque a cualquier persona que no tenga mis credenciales.



Finalmente tras autenticarse correctamente con el usuario root y la clave que generé al principio el gestor de arranque liberó el kernel y el sistema operativo cargó con normalidad hasta llegar al escritorio gráfico de Ubuntu con la medusa que muestro en la tercera imagen, confirmando así que el proceso de protección del arranque se ha completado con éxito y el sistema es completamente funcional.



## Actividad 6. Lectura del punto 4.3. Encriptar las particiones con LUKS del documento Bastionado de Sistemas Operativos Linux. Parte I. Realiza una tabla con las ventajas y desventajas del cifrado completo de disco vs cifrado basado en archivos.

| Característica | Cifrado Completo de Disco (FDE) | Cifrado Basado en Archivos |
|---|---|---|
| **Nivel de Cifrado** | Opera a nivel de disco (bloques) por debajo del sistema de archivos. | Implementado mediante bibliotecas y depende del sistema de archivos. |
| **Alcance** | Lo cifra todo sin diferenciar si la información es sensible o no. | Se aplica a contextos específicos definidos por el desarrollador. |
| **Acceso/Uso** | Transparente para el usuario. Si el sistema está encendido, el disco está desbloqueado (Desventaja). | Debe ser descifrado cada vez que se necesite usar el archivo. |
| **Compatibilidad** | Independiente del sistema de archivos. | Podría no funcionar en otros ordenadores si no se implementa correctamente. |

| | | |
|---|---|---|
| **Seguridad Principal** | Ideal para proteger los datos si el equipo (portátil o disco externo) es robado mientras está apagado. | Protege archivos individuales, pero requiere gestión constante de descifrado. |

## Actividad 7. Lectura del punto 4.3. Encriptar las particiones con LUKS del documento Bastionado de Sistemas Operativos Linux. Parte I. Con la ayuda del ChatGPT u otras fuentes (manuales, videos) compara la información de los apuntes ¿Cómo funciona LUKS? y hazte tu propio resumen de dicho funcionamiento y qué comandos deberíamos de usar para realizar un backups de los encabezados y la forma de recuperarlo en caso de corrupción de la cabecera LUKS. Anota las fuentes consultadas.

**A) ¿Cómo funciona LUKS? (Resumen técnico)**

LUKS (Linux Unified Key Setup) funciona mediante una estructura de **cabecera (header)** situada al principio del disco:

1. **Cabecera y Metadatos:** El disco no empieza directamente con los datos cifrados, sino con una cabecera que contiene la configuración (algoritmo, sal, UUID) y metadatos en formato JSON (en LUKS2 hay dos copias para redundancia).
2. **Master Key (Llave Maestra):** Los datos de la partición se cifran con una única clave maestra. Esta clave **no se guarda en texto plano**.
3. **Keyslots (Ranuras de Clave):** La Master Key se cifra con tu contraseña (frase de paso) y se almacena en una de las ranuras o *keyslots*.
4. **Desbloqueo:** Al iniciar, introduces tu contraseña. LUKS prueba a descifrar los *keyslots*. Si acierta, libera la Master Key en la memoria RAM y el módulo dm-crypt del kernel la usa para leer y escribir los datos en tiempo real.

**B) Comandos de Backup y Restauración**

Si la cabecera se daña (sectores defectuosos, error de escritura), **pierdes el acceso a todos los datos** para siempre, porque la Master Key se vuelve ilegible. Por eso el backup es crítico.

**Para realizar el Backup (Copia de seguridad):** Guarda los metadatos y las claves en un archivo fuera del disco cifrado.

*sudo cryptsetup luksHeaderBackup /dev/sdXn --header-backup-file /ruta/segura/header_backup.img*

**Para Restaurar la Cabecera (Recuperación):** Si el disco no reconoce tu contraseña por corrupción de la cabecera, usa este comando para restaurarla desde la copia:

*sudo cryptsetup luksHeaderRestore /dev/sdXn --header-backup-file /ruta/segura/header_backup.img*

# Actividad 8. Cifrado de un Disco o partición e integración en el arranque (resuelta, Paraninfo). En una máquina virtual con sistema operativo Linux, añade un disco nuevo. Configura el disco con una única partición y que esté cifrado por completo. Una vez creada la tabla de arranque, configura la encriptación con la herramienta Cryptsetup, y formatea la partición. Realiza las comprobaciones de funcionamiento y añade la partición encriptada en el inicio del sistema.

El objetivo de esta práctica ha sido crear un volumen de almacenamiento seguro, cifrado mediante el estándar LUKS, y configurar el sistema para que solicite la contraseña de descifrado automáticamente durante el inicio del sistema operativo.

**1. Creación de la Partición.** En primer lugar, se identificó el nuevo disco añadido al sistema (/dev/sda). Utilizando la herramienta fdisk, se creó una nueva partición primaria (/dev/sda1) asignándole el total del espacio disponible (15 GiB). Se guardaron los cambios en la tabla de particiones con la orden w.

```
usuario@ciber:~$ sudo fdisk /dev/sda
[sudo] contraseña para usuario:

Bienvenido a fdisk (util-linux 2.37.2).
Los cambios solo permanecerán en la memoria, hasta que decida escribirlos.
Tenga cuidado antes de utilizar la orden de escritura.

El dispositivo no contiene una tabla de particiones reconocida.
Se ha creado una nueva etiqueta de disco DOS con el identificador de disco 0xc75
a39a2.

Orden (m para obtener ayuda): n
Tipo de partición
   p   primaria (0 primary, 0 extended, 4 free)
   e   extendida (contenedor para particiones lógicas)
Seleccionar (valor predeterminado p): p
Número de partición (1-4, valor predeterminado 1):
Primer sector (2048-31457279, valor predeterminado 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-31457279, valor predetermina
do 31457279):

Crea una nueva partición 1 de tipo 'Linux' y de tamaño 15 GiB.

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
```

**2. Cifrado del Volumen.** Se procedió a cifrar la nueva partición utilizando el comando cryptsetup luksFormat /dev/sda1. Tras confirmar la operación escribiendo "YES" en mayúsculas (para evitar borrados accidentales), se estableció una frase de paso robusta que servirá como clave para encriptar la *Master Key* del volumen.

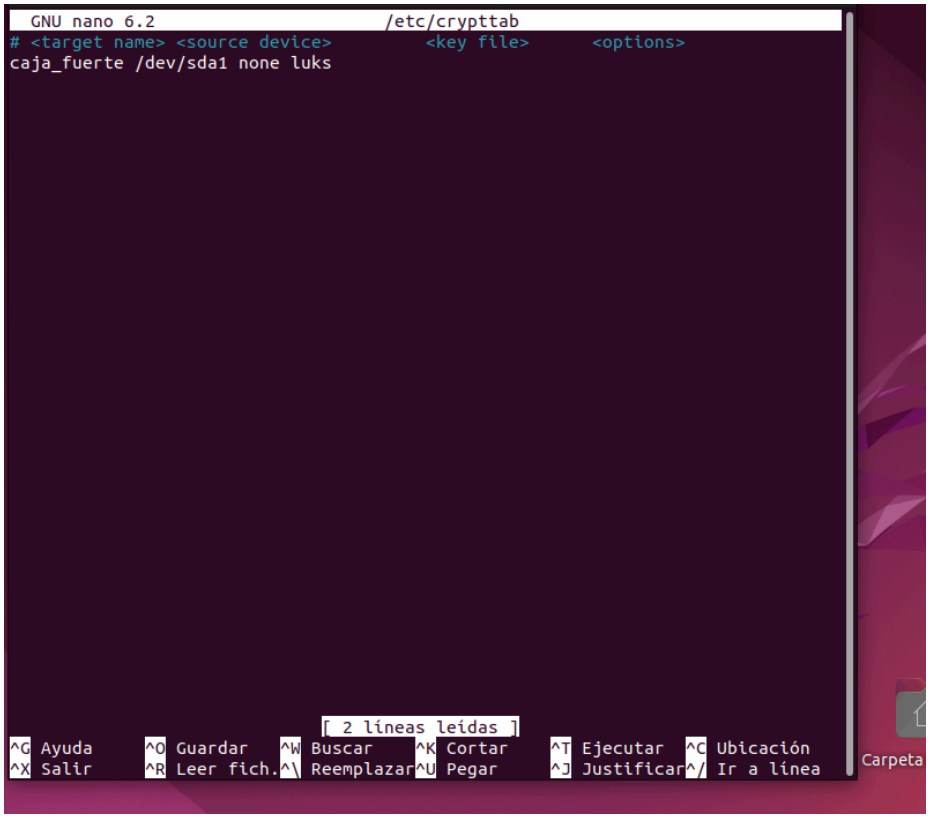
```
usuario@ciber:~$ sudo cryptsetup luksFormat /dev/sda1

WARNING!
========
Sobrescribirá los datos en /dev/sda1 de forma irrevocable.

Are you sure? (Type 'yes' in capital letters): yes
Operation aborted.

usuario@ciber:~$ sudo cryptsetup luksFormat /dev/sda1

WARNING!
========
Sobrescribirá los datos en /dev/sda1 de forma irrevocable.

Are you sure? (Type 'yes' in capital letters): YES
Introduzca una contraseña para /dev/sda1:
Verificar frase de paso:
```

**3. Apertura y Mapeo del Dispositivo.** Para poder trabajar con el volumen cifrado, fue necesario "abrirlo" mediante el comando cryptsetup open /dev/sda1 caja_fuerte. Esto creó un dispositivo virtual mapeado en /dev/mapper/caja_fuerte, tal y como se verificó listando el directorio, lo que permite al kernel interactuar con los datos en texto plano mientras se escriben cifrados en el disco físico.

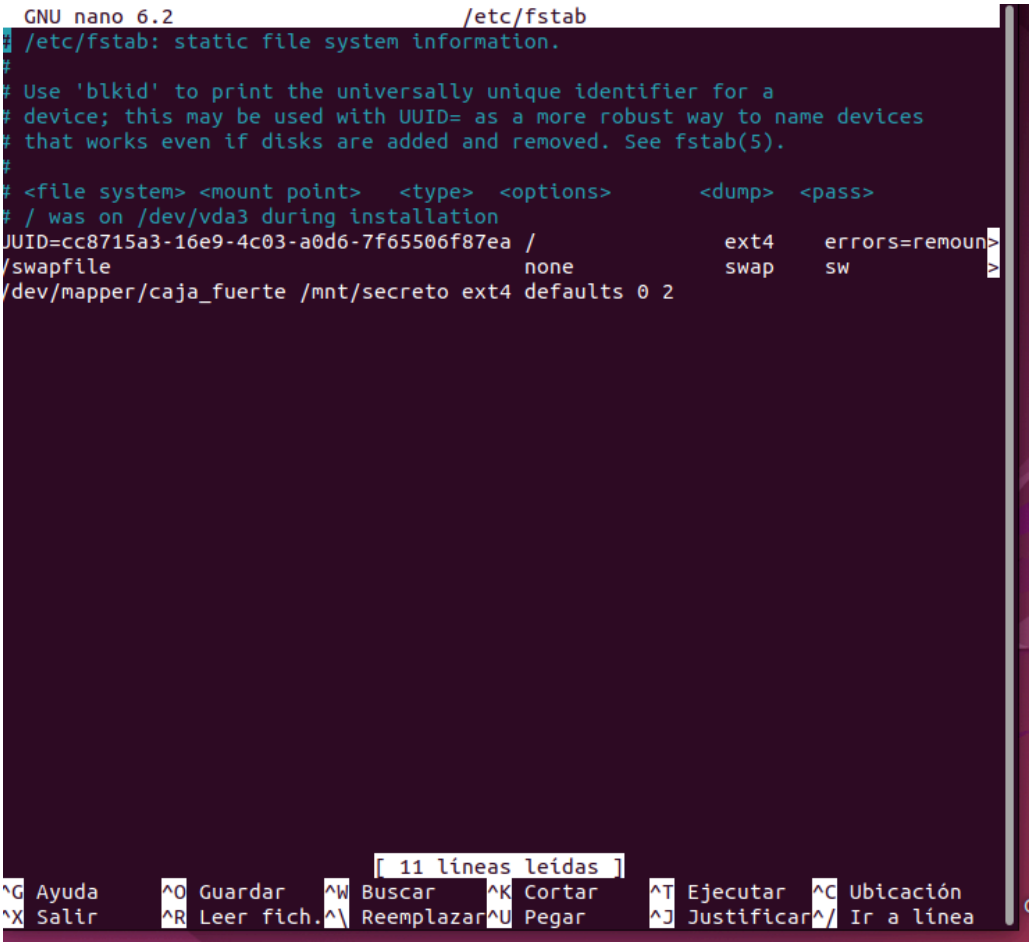
```
usuario@ciber:~$ sudo cryptsetup open /dev/sda1 caja_fuerte
Introduzca una contraseña para /dev/sda1:
usuario@ciber:~$ ls -l /dev/mapper/caja_fuerte
lrwxrwxrwx 1 root root 7 nov 25 19:56 /dev/mapper/caja_fuerte -> ../dm-0
```

**4. Configuración de la Persistencia.** Para evitar tener que montar el disco manualmente tras cada reinicio, se editaron dos archivos de configuración del sistema:

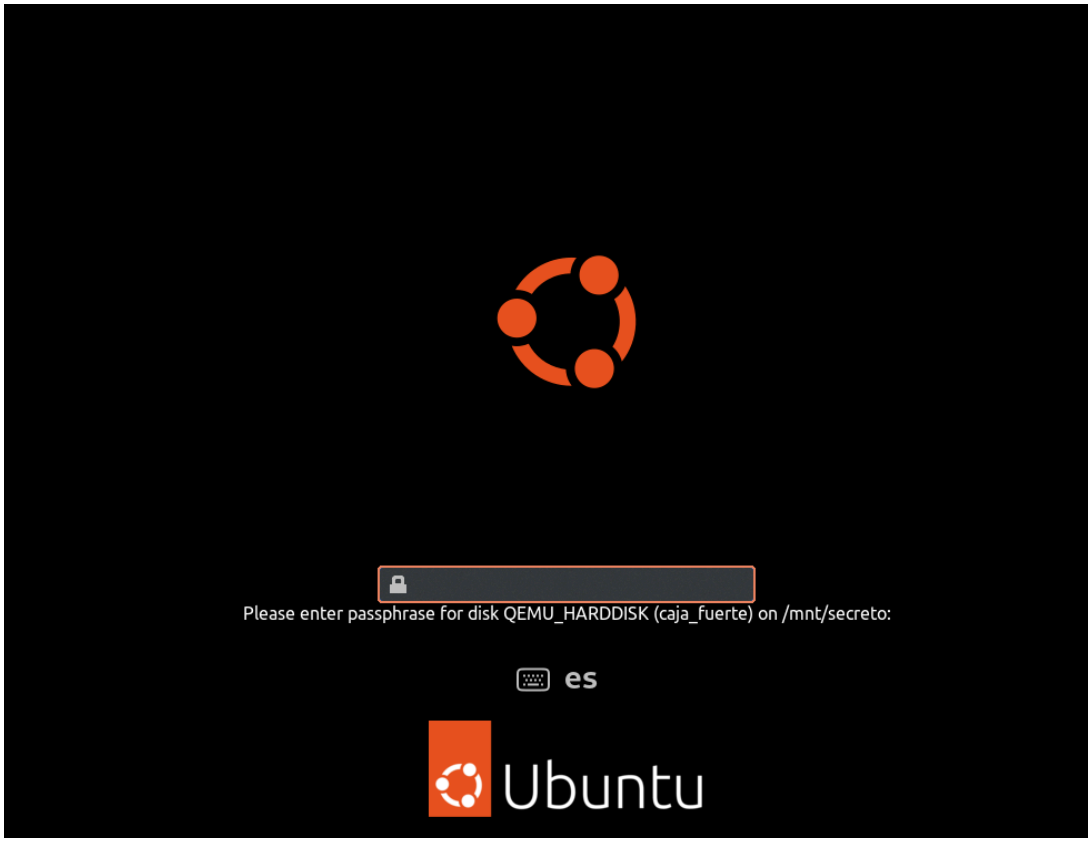
- **/etc/crypttab:** Se añadió la línea caja_fuerte /dev/sda1 none luks para indicar al sistema que debe intentar desbloquear esa partición durante el arranque solicitando la contraseña por consola.

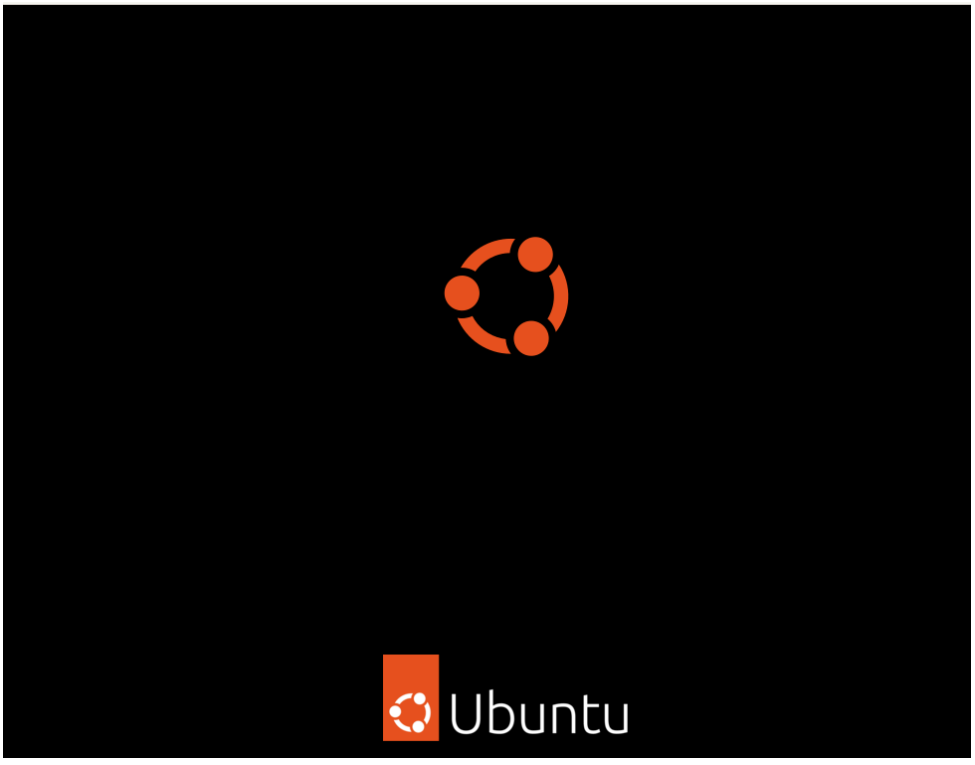


- **/etc/fstab:** Se configuró el punto de montaje automático añadiendo la línea /dev/mapper/caja_fuerte /mnt/secreto ext4 defaults 0 2, asegurando que, una vez descifrado, el volumen se monte en el directorio /mnt/secreto.




**5. Verificación y Resultados.** Finalmente, se reinició la máquina virtual para comprobar el bastionado. Durante el proceso de carga (Splash Screen de Ubuntu), el sistema detuvo el arranque y solicitó la frase de paso para el disco "caja_fuerte". Tras introducir la contraseña correcta, el sistema continuó la carga y permitió el inicio de sesión del usuario, confirmando que el cifrado y el montaje automático funcionan correctamente.

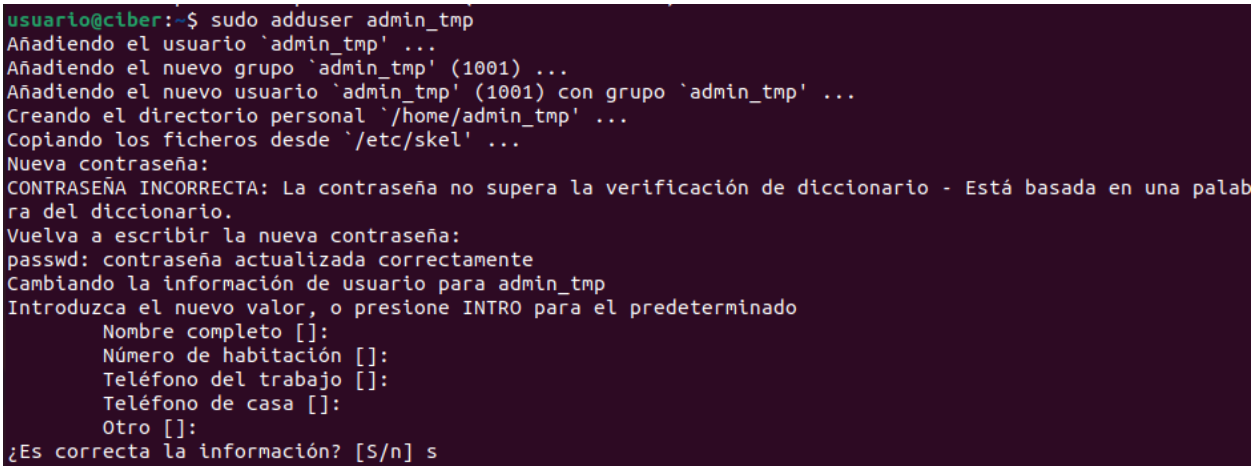

Please enter passphrase for disk QEMU_HARDDISK (caja_fuerte) on /mnt/secreto:
es
Ubuntu


Ubuntu

## Actividad 9. Encriptación de directorios con eCryptfs. Además, como hemos visto para una seguridad plena, LUKS permite hacer un backup de sus encabezados en un archivo, que almacenaremos en un medio externo al disco cifrado por LUKS (sería interesante que lo practicases) a) Encriptación de directorios home de uno de tus usuarios. b) Opcional. Encriptación de otros directorios o un pendrive. c) Encriptación de la partición swap con eCryptfs.

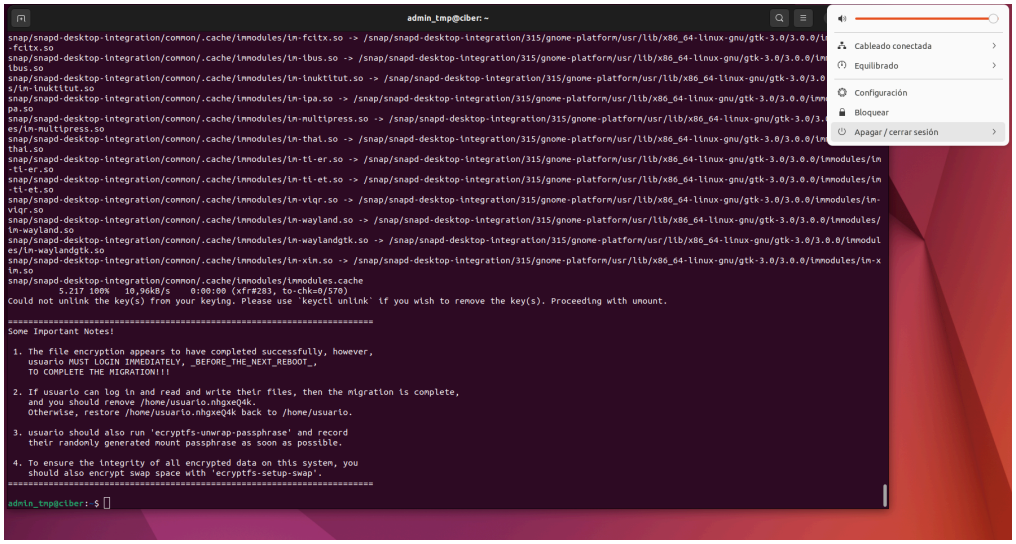
El objetivo de esta práctica ha sido proteger la confidencialidad de los datos del usuario y la memoria temporal del sistema utilizando eCryptfs.

**1. Preparación del Entorno.** Dado que no es posible cifrar el directorio de un usuario mientras este tiene la sesión iniciada, se creó un usuario administrador temporal denominado admin_tmp mediante el comando sudo adduser admin_tmp. Esto permitió realizar las operaciones de mantenimiento desde una sesión externa segura.



**2. Migración y Cifrado del Home.** Desde la sesión del usuario temporal, se ejecutó la herramienta de migración con sudo ecryptfs-migrate-home -u usuario. El sistema procedió a sincronizar los datos hacia un nuevo directorio cifrado. Aunque se mostraron advertencias sobre archivos abiertos (lsof), el proceso concluyó exitosamente mostrando las instrucciones de seguridad críticas (login inmediato y recuperación de frase de paso).

```
admin_tmp@ciber:~$ sudo ecryptfs-migrate-home -u usuario
[sudo] contraseña para admin_tmp:
INFO:  Checking disk space, this may take a few moments.  Please be patient.
INFO:  Checking for open files in /home/usuario
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.portal file system /run/user/1000/doc
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1001/gvfs
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.portal file system /run/user/1001/doc
      Output information may be incomplete.
INFO:  The following files are in use:

lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.portal file system /run/user/1000/doc
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1001/gvfs
      Output information may be incomplete.
lsof: WARNING: can't stat() fuse.portal file system /run/user/1001/doc
      Output information may be incomplete.
```



**3. Recuperación de la Frase de Paso.** Tras iniciar sesión nuevamente con el usuario propietario (usuario) para verificar que el descifrado en tiempo real funcionaba, se ejecutó el comando ecryptfs-unwrap-passphrase. Se obtuvo la **Mount Passphrase**, la cual se ha guardado de forma segura para poder recuperar los datos en caso de emergencia o cambio de contraseña.

```
usuario@ciber:~$ ecryptfs-unwrap-passphrase
Passphrase:
fae9b28b89395907e16a050e6ed35163
usuario@ciber:~$
```

**4. Verificación del Montaje Cifrado.** Se comprobó el estado del sistema de ficheros con el comando mount | grep ecryptfs. Como se observa en la evidencia, el directorio /home/usuario aparece montado con el tipo ecryptfs, confirmando que los datos se escriben cifrados en el disco y se descifran al vuelo para el usuario.

```
usuario@ciber:~$ mount | grep ecryptfs
/home/.ecryptfs/usuario/.Private on /home/usuario type ecryptfs (rw,nosuid,nodev
,relatime,ecryptfs_fnek_sig=8a1a5efae5a0b49f,ecryptfs_sig=6796ff36c40c00e9,ecryp
tfs_cipher=aes,ecryptfs_key_bytes=16,ecryptfs_unlink_sigs)
usuario@ciber:~$ ls /mnt/secreto
lost+found  top_secret.txt
```

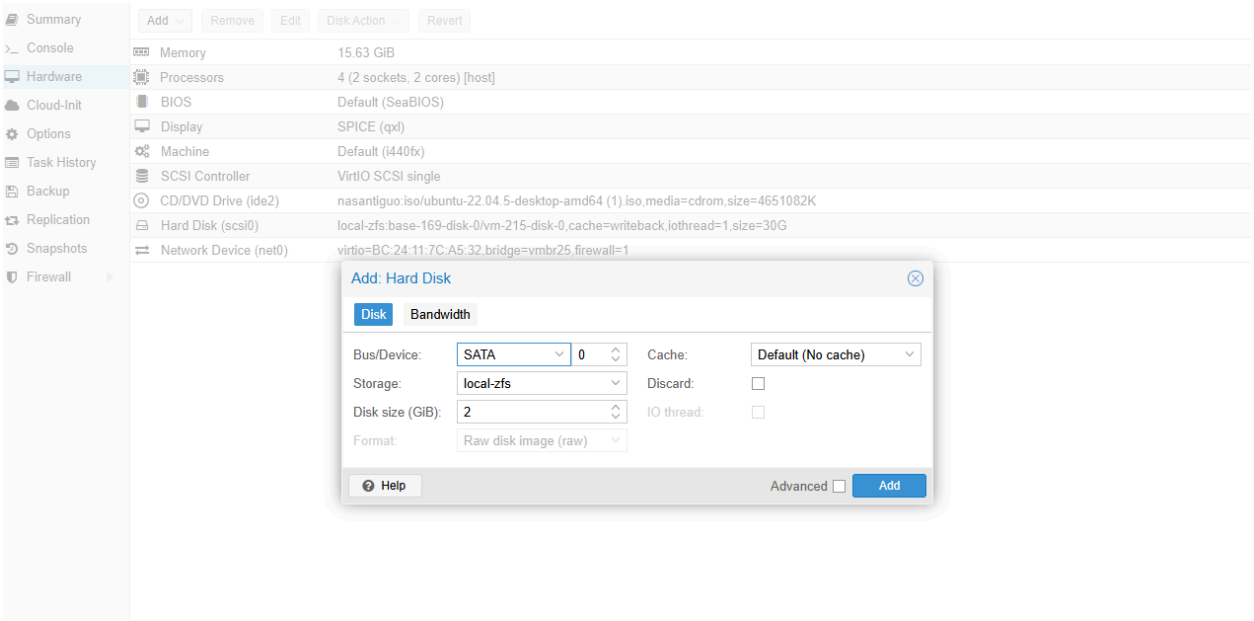
**5. Cifrado de la Memoria de Intercambio (Swap).** Finalmente, se aseguró la memoria virtual. La salida del comando swapon -s muestra que la partición de intercambio activa es /dev/dm-0 (un dispositivo mapeado por el kernel) en lugar de la partición física directa (/dev/sdaX). Esto indica que la Swap está cifrada, evitando que fragmentos de memoria RAM (que podrían contener contraseñas) queden legibles en el disco.

```
usuario@ciber:~$ swapon -s
Filename                                Type            Size            Used    P
riority
/dev/dm-0                               partition       758708          0       -
2
usuario@ciber:~$
```

# Actividad 10.Borrado seguro de un Disco sin rotura física (resulta, Paraninfo). Realiza un borrado seguro del disco que se ha añadido en la práctica anterior o añade un nuevo disco a una de tus máquinas virtuales.

El objetivo de esta actividad ha sido realizar un borrado seguro de un dispositivo de almacenamiento para garantizar que la información contenida sea irrecuperable, simulando un escenario de reutilización o desecho de hardware.

**1. Preparación del Entorno.** Para no comprometer el sistema operativo principal, se añadió un nuevo disco duro virtual de **2 GiB** a la máquina virtual a través de la interfaz de hardware de Proxmox. Esto nos permite simular un disco externo o secundario que necesita ser higienizado.



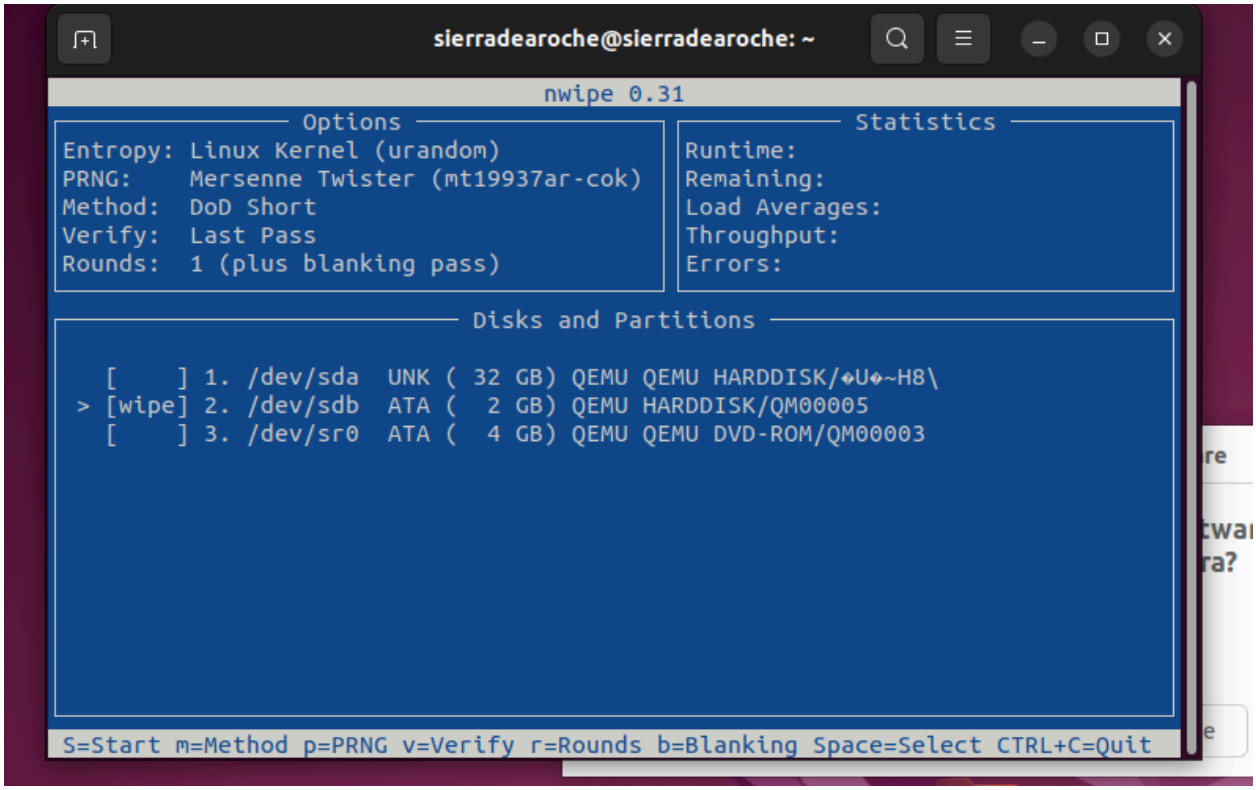

**2. Selección de Herramienta e Instalación.** Se optó por utilizar **nwipe**, una herramienta de *wiping* que implementa el mismo motor que DBAN pero ejecutable directamente desde la terminal. Se procedió a su instalación mediante el comando sudo apt install nwipe.

```
sierradearoche@sierradearoche:~$ sudo apt install && sudo apt install nwipe
[sudo] contraseña para sierradearoche:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
```

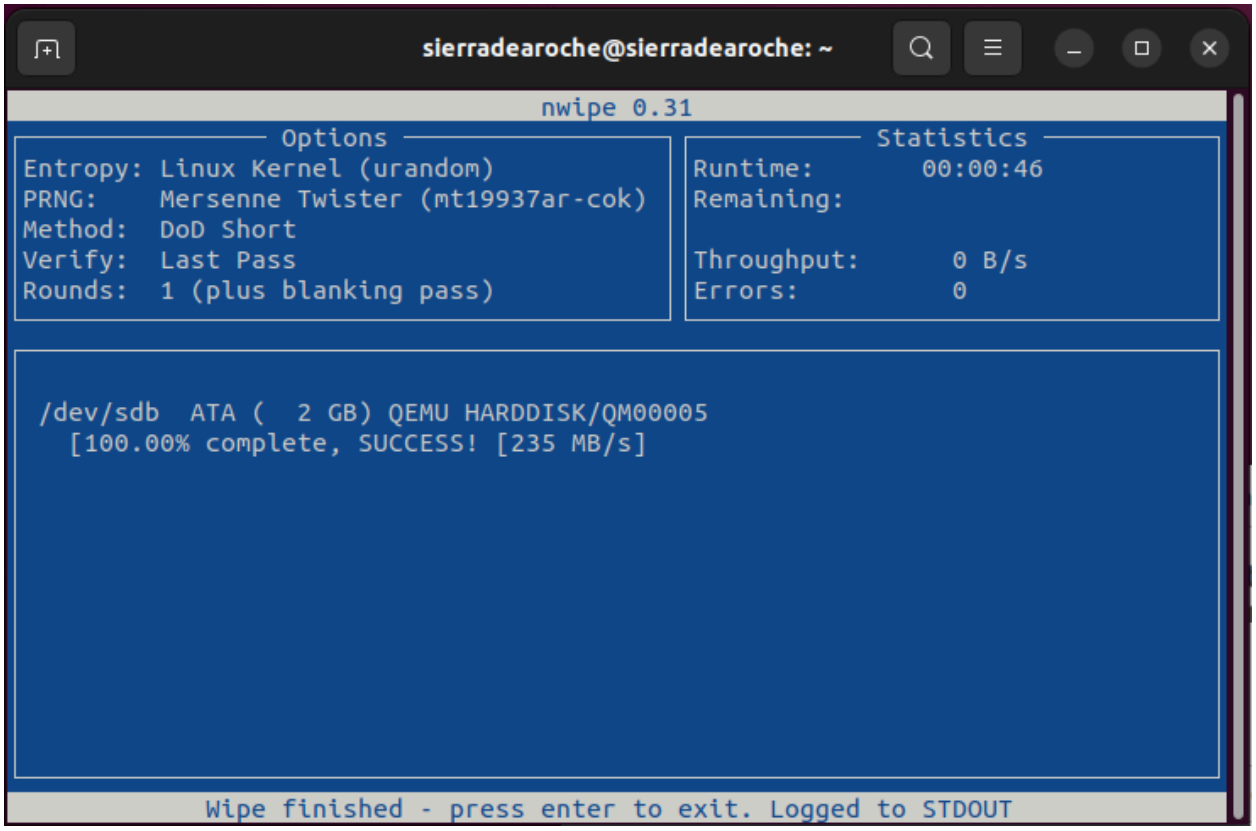
**3. Identificación del Dispositivo.** Antes de iniciar el borrado, se ejecutó el comando lsblk para identificar inequívocamente el disco objetivo. Se verificó que el dispositivo /dev/sdb tenía el tamaño de 2G, confirmando que era el disco seguro para borrar y evitando afectar al disco del sistema (/dev/sda).

```
sierradearoche@sierradearoche:~$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
loop0    7:0    0     4K  1 loop /snap/bare/5
loop1    7:1    0  74,3M  1 loop /snap/core22/1612
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    30G  0 disk
├─sda1   8:1    0     1M  0 part
├─sda2   8:2    0   513M  0 part /boot/efi
└─sda3   8:3    0  29,5G  0 part /
sdb      8:16   0     2G  0 disk
sr0     11:0    1   4,4G  0 rom  /media/sierradearoche/Ubuntu 22.04.5 LTS amd64
```

**4. Ejecución del Borrado Seguro.** Se lanzó nwipe y se configuró el método de borrado **DoD Short** (Estándar del Departamento de Defensa de EE. UU.), que realiza 3 pasadas de sobreescritura. Como se observa en las capturas, se seleccionó el disco /dev/sdb y se inició el proceso, mostrando la velocidad de escritura y el progreso de las rondas.




**5. Verificación de resultados.** El proceso finalizó mostrando el mensaje **"Success"** y **"Wipe finished"**. Esto indica que cada sector del disco ha sido sobrescrito con patrones aleatorios y ceros. Aunque el dispositivo sigue apareciendo en lsblk (porque sigue conectado físicamente a la máquina), su tabla de particiones y sistema de ficheros han sido totalmente destruidos, haciendo la recuperación de datos imposible por métodos software estándar.


sierradearoche@sierradearoche: ~
nwipe 0.31
Options
Entropy: Linux Kernel (urandom)
PRNG: Mersenne Twister (mt19937ar-cok)
Method: DoD Short
Verify: Last Pass
Rounds: 1 (plus blanking pass)
Statistics
Runtime: 00:00:46
Remaining:
Throughput: 0 B/s
Errors: 0
/dev/sdb ATA ( 2 GB) QEMU HARDDISK/QM00005
[100.00% complete, SUCCESS! [235 MB/s]
Wipe finished - press enter to exit. Logged to STDOUT