



SERVICIOS SYSTEMD / SYSTEM-SHUTDOWN:



OTROS DOCUMENTOS DE LINUX

[SCRIPTS_BASH](#)

[BASH_COMANDS](#)

[BASH COMANDS - Extended](#)



INDICE DOCUMENTO

	Script .sh	Permisos		Script .sh	Permisos
SERVICIO DE ARRANQUE - I	Script SYSTEMD	Habilitar Serv. Reiniciar Serv.	SERVICIO DE APAGADO - I	Script SHUTDOWN	Habilitar Serv. Reiniciar Serv.
SERVICIO DE ARRANQUE - II	Servicio Sin Archivo .sh		SERVICIO DE APAGADO - II	Servicio Sin Archivo .sh	



CONCEPTOS UTILES



CREAR UN SERVICIO DE ARRANQUE



SERVICIO SYSTEMD CON SCRIPT .SH

■ CREAR SCRIPT	▶▶
■ DARLE PERMISOS EJECUCION	▶▶
■ CREAR ARCHIVO DE SERVICIO SYSTEMD	▶▶
■ HABILITAR EL SERVICIO AL INICIO:	▶▶
■ PROBAR EL SCRIPT SIN REINICIAR:	▶▶
■ REVISAR LOGS DEL SERVICIO:	▶▶
■ RECARGAR Y REINICIAR EL SERVICIO	▶▶
■ SERVICIO SYSTEMD SIN .SH	



SERVICIO SYSTEMD SIN .SH

■ CREAR ARCHIVO DE SERVICIO SYSTEMD	▶▶
■ HABILITAR EL SERVICIO AL INICIO:	▶▶
■ PROBAR EL SCRIPT SIN REINICIAR:	▶▶
■ REVISAR LOGS DEL SERVICIO:	▶▶
■ RECARGAR Y REINICIAR EL SERVICIO	▶▶



CREAR UN SERVICIO DE APAGADO



SERVICIO SYSTEM-SHUTDOWN

■ CREAR SCRIPT	▶▶
■ DARLE PERMISOS EJECUCION	▶▶
■ CREAR SERVICIO SYSTEM-SHUTDOWN	▶▶
■ HABILITAR EL SERVICIO AL INICIO:	▶▶
■ PROBAR EL SCRIPT SIN APAGAR:	▶▶
■ REVISAR LOGS DEL SERVICIO:	▶▶
■ RECARGAR Y REINICIAR EL SERVICIO	▶▶



RECARGAR+REINICIAR EL SERVICIO:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl restart montar_drive.service
```



CONCEPTOS



Haz un ejemplo de un script de bash para linux para montar una unidad remota por ejemplo Drive, con rclone

ChatGPT Plus

Claro, aquí tienes un ejemplo de un script de Bash para montar una unidad remota de Google Drive usando rclone en Linux:

Requisitos:

Tener rclone instalado y configurado con una unidad remota (por ejemplo, llamada miDrive).

Tener permisos para montar sistemas de archivos (puede requerir fusermount o mount).

Script: montar_drive.sh





```
#!/bin/bash
```

```
# Nombre de la unidad remota configurada en rclone
```

```
REMOTE_NAME="miDrive"
```

```
# Carpeta local donde se montará la unidad
```

```
MOUNT_POINT="$HOME/DriveRemoto"
```

```
# Crear punto de montaje si no existe
```

```
mkdir -p "$MOUNT_POINT"
```

```
# Montar la unidad remota con rclone
```

```
rclone mount "$REMOTE_NAME": "$MOUNT_POINT" \
```

```
--vfs-cache-mode writes \
```

```
--allow-other \
```

```
--daemon
```

```
# Confirmación
```

```
if mount | grep "$MOUNT_POINT" > /dev/null; then
```

```
    echo "Unidad remota montada exitosamente en $MOUNT_POINT"
```

```
else
```

```
    echo "Error al montar la unidad remota"
```

```
fi
```

Instrucciones para usarlo:

Guarda el script como montar_drive.sh.

Dale permisos de ejecución:

```
chmod +x montar_drive.sh
```

Ejecútalo:

```
./montar_drive.sh
```

Nota: Si --allow-other da error, puede que debas habilitarlo en /etc/fuse.conf descomentando la línea user_allow_other.

¿Te gustaría que te haga también un script para desmontar la unidad?

COMO HACER QUE SIMPRE, AL ARRANCAR EL ORDENADO, SE EJECUTE UN SCRIPT DA BASH PARA MONTAR UNIDADES REMOTAS?

ChatGPT Plus

Para que un script Bash se ejecute automáticamente al arrancar el ordenador y monte unidades remotas (por ejemplo, usando sshfs, cifs, etc.), hay varias formas de lograrlo según el sistema operativo y el entorno de inicio. A continuación te explico cómo hacerlo en Linux (Debian, Ubuntu, etc.) usando métodos confiables:

✓ Método recomendado: Crear un servicio systemd

Crea tu script de montaje, por ejemplo:

```
#!/bin/bash
```





```
# /usr/local/bin/montar_remoto.sh
```

```
# Ejemplo con sshfs:
```

```
sshfs usuario@192.168.1.100:/ruta/remota /mnt/remoto -o  
reconnect,ServerAliveInterval=15,ServerAliveCountMax=3
```

Asegúrate de que /mnt/remoto existe y que el usuario tiene acceso
Dale permisos de ejecución:

```
sudo chmod +x /usr/local/bin/montar_remoto.sh
```

Crea un archivo de servicio systemd:

```
sudo nano /etc/systemd/system/montar-remoto.service
```

Pega este contenido adaptado:

```
[Unit]
```

```
Description=Montar unidad remota al inicio
```

```
After=network-online.target
```

```
Wants=network-online.target
```

```
[Service]
```

```
Type=simple
```

```
User=TU_USUARIO
```

```
ExecStart=/usr/local/bin/montar_remoto.sh
```

```
Restart=on-failure
```

```
[Install]
```

```
WantedBy=default.target
```

⚠ Reemplaza TU_USUARIO con tu nombre de usuario.

Habilita el servicio para que se inicie al arrancar:

```
sudo systemctl daemon-reexec
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable montar-remoto.service
```

Prueba el script sin reiniciar:

```
sudo systemctl start montar-remoto.service
```

¿Quieres montar la unidad como usuario normal o como root?

Tú dijiste:

este script que pongo a continuacion:

```
#!/bin/bash
```

```
# /usr/local/bin/montar_drive.sh
```

```
rclone mount --allow-non-empty gdrive: /home/gerar_kde/Drive &
```





exit

no está funcionando como servicio con este archivo

```
[Unit]
Description=Montar Google Drive al inicio
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=gerar_kde@Lap-Linux
ExecStart=/usr/local/bin/montar_drive.sh
Restart=on-failure
[Install]
WantedBy=default.target
```

el servicio ya se ha habilitado, y el archivo montar_drive.sh tiene permisos de ejecución,

al hacer

```
sudo systemctl start montar_drive.service
```

no monta drive, puedes mejorar a ver donde está el error?

ChatGPT Plus

Gracias por compartir los detalles. Hay varios puntos importantes que podrían estar impidiendo que el servicio funcione correctamente. Vamos a revisar paso a paso:

🔍 1. Problemas posibles en tu script
Tu script actual:

```
#!/bin/bash
rclone mount --allow-non-empty gdrive: /home/gerar_kde/Drive &
exit
```

Problemas:

El & hace que el proceso se ejecute en segundo plano, y luego haces exit, lo que mata el proceso del script junto con el montaje, porque Type=simple espera que el proceso continúe.

--allow-non-empty no es recomendable, puede causar errores si la carpeta ya tiene contenido.

No usas --vfs-cache-mode, que es obligatorio para muchas operaciones con rclone mount.

🔧 Corregido:

```
#!/bin/bash
/usr/bin/rclone mount \
  gdrive: /home/gerar_kde/Drive \
  --vfs-cache-mode writes \
```





`--allow-other`

Asegúrate de que `/home/gerar_kde/Drive` existe y que tienes permiso.



2. Corrección del archivo de servicio `systemd`

Tu archivo actual tiene un problema en la línea de `User=`:

`User=gerar_kde@Lap-Linux` # ✗ Esto es inválido

`User=` debe contener solo el nombre de usuario del sistema, no el hostname. Entonces debe ser:

`User=gerar_kde`



Archivo corregido completo:

```
[Unit]
Description=Montar Google Drive al inicio
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=gerar_kde
ExecStart=/usr/local/bin/montar_drive.sh
Restart=on-failure
Environment=PATH=/usr/bin:/bin
[Install]
WantedBy=default.target
```



Pasos finales:

Revisa la ruta de `rclone`:

`which rclone`

Asegúrate de que es `/usr/bin/rclone` o cambia la ruta en el script si es diferente.

Revisar logs del servicio:

Después de hacer

`sudo systemctl start montar_drive.service`

revisa con:

`journalctl -u montar_drive.service --no-pager`

Ahí verás si falla algo como permisos, rutas, o errores de `rclone`.

Recarga y reinicia el servicio:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl restart montar_drive.service
```

¿Quieres que te ayude a hacer una versión más robusta que espere red y monte sólo si la carpeta





está vacía?

Tú dijiste:

no quiero que saques un script tambien para un servicio que monte una particion del disco duro como: "Datos-D" como ntfs, su UUID es 524F567904D39ECB

ChatGPT Plus

Perfecto. Vamos a crear un script y un servicio systemd para montar automáticamente una partición NTFS (UUID 524F567904D39ECB) en una carpeta como /mnt/Datos-D.



1. Crear el punto de montaje

Primero, asegúrate de que el directorio donde se montará exista:

```
sudo mkdir -p /mnt/Datos-D
sudo chown gerar_kde:gerar_kde /mnt/Datos-D
```



2. Crear el script de montaje

Archivo: /usr/local/bin/montar_datos_d.sh

```
#!/bin/bash
mount -t ntfs3 -o rw,uid=1000,gid=1000,umask=022 UUID=524F567904D39ECB
/mnt/Datos-D
```

- ♦ Explicación de las opciones:

ntfs3: controlador moderno para NTFS (más rápido que ntfs-3g en kernels nuevos).

uid=1000,gid=1000: asegura que el usuario padrão (gerar_kde) tenga acceso.

umask=022: da permisos de lectura para todos y escritura sólo para el dueño.

Dale permisos de ejecución:

```
sudo chmod +x /usr/local/bin/montar_datos_d.sh
```



3. Crear servicio systemd

Archivo: /etc/systemd/system/montar-datos-d.service

```
[Unit]
Description=Montar partición Datos-D (NTFS)
After=local-fs.target
```

```
[Service]
Type=oneshot
ExecStart=/usr/local/bin/montar_datos_d.sh
RemainAfterExit=true
```

```
[Install]
WantedBy=multi-user.target
```





- ♦ Type=oneshot porque solo monta una vez.
- ♦ RemainAfterExit=true para que systemd lo considere "activo" después de ejecutar.

✓ 4. Habilitar y probar el servicio

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar-datos-d.service
sudo systemctl start montar-datos-d.service
```

Verifica si está montado:

`mount | grep Datos-D`

¿Quieres que añada una verificación en el script para evitar que se monte si ya está montado?

Tú dijiste:

Que hace esta parte?

```
sudo chown gerar_kde:gerar_kde /mnt/Datos-D
```

ChatGPT Plus

Ese comando:

```
sudo chown gerar_kde:gerar_kde /mnt/Datos-D
```

significa:

chown: cambia el propietario de un archivo o directorio.

gerar_kde:gerar_kde: asigna como propietario al usuario gerar_kde y también al grupo gerar_kde.
/mnt/Datos-D: es el directorio del punto de montaje al que se le aplican esos cambios.

¿Por qué es útil?

Cuando montas una partición NTFS con ciertas opciones (como uid, gid), el usuario que aparece como dueño del sistema de archivos montado debe coincidir con el dueño del directorio donde se monta (en este caso /mnt/Datos-D), si quieres poder leer y escribir sin usar sudo.

Así te aseguras de que el usuario gerar_kde tenga acceso completo a esa carpeta una vez montada.

¿Quieres que el punto de montaje esté en otro sitio, como en tu carpeta personal?

Tú dijiste:

No

ChatGPT Plus

Perfecto, entonces dejamos el punto de montaje en /mnt/Datos-D como estaba.

Si ya has creado el script y el servicio como te indiqué, deberías poder montar la partición automáticamente al iniciar el sistema. Si necesitas que revise los permisos después de montarla o quieres añadir validaciones (como evitar doble montaje), puedo ayudarte a mejorar el script también.

¿Te gustaría añadir esa verificación para que no intente montarla si ya está montada?



Tú dijiste:

Si y también verificación de permisos

ChatGPT Plus

Genial. Vamos a mejorar el script para:

Evitar montar si ya está montada

Verificar permisos de escritura del punto de montaje

Informar claramente en caso de error

✓ Script mejorado: /usr/local/bin/montar_datos_d.sh

```
#!/bin/bash
```

```
MOUNT_POINT="/mnt/Datos-D"
UUID="524F567904D39ECB"
FS_TYPE="ntfs3"
```

```
# Comprobamos si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    echo "La partición ya está montada en $MOUNT_POINT."
    exit 0
fi
```

```
# Comprobamos si el punto de montaje existe y tiene permisos
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: El directorio $MOUNT_POINT no existe."
    exit 1
fi
```

```
if [ ! -w "$MOUNT_POINT" ]; then
    echo "Error: No tienes permisos de escritura en $MOUNT_POINT."
    exit 2
fi
```

```
# Intentamos montar
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID "$MOUNT_POINT"
STATUS=$?
```

```
if [ $STATUS -eq 0 ]; then
    echo "Partición montada correctamente en $MOUNT_POINT."
else
    echo "Error al montar la partición (código $STATUS)."
```

```
exit $STATUS
fi
```

✓ Asegúrate de:





El script tiene permisos de ejecución:

```
sudo chmod +x /usr/local/bin/montar_datos_d.sh
```

Tu usuario gerar_kde tiene UID 1000 (casi siempre es así para el primer usuario). Puedes verificarlo con:

```
id -u gerar_kde
```

El servicio systemd sigue apuntando al mismo script, no necesitas cambiarlo si ya lo creaste.

¿Quieres que el script registre todo en un archivo de log también?

Tú dijiste:

cambia el script con punto de montaje /media/DATOS

ChatGPT Plus

Perfecto, aquí tienes el script actualizado con el nuevo punto de montaje /media/DATOS:

✓ Script actualizado: /usr/local/bin/montar_datos_d.sh
#!/bin/bash

```
MOUNT_POINT="/media/DATOS"
```

```
UUID="524F567904D39ECB"
```

```
FS_TYPE="ntfs3"
```

```
# Comprobamos si ya está montado
```

```
if mountpoint -q "$MOUNT_POINT"; then
```

```
    echo "La partición ya está montada en $MOUNT_POINT."
```

```
    exit 0
```

```
fi
```

```
# Comprobamos si el punto de montaje existe y tiene permisos
```

```
if [ ! -d "$MOUNT_POINT" ]; then
```

```
    echo "Error: El directorio $MOUNT_POINT no existe."
```

```
    exit 1
```

```
fi
```

```
if [ ! -w "$MOUNT_POINT" ]; then
```

```
    echo "Error: No tienes permisos de escritura en $MOUNT_POINT."
```

```
    exit 2
```

```
fi
```

```
# Intentamos montar
```

```
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID "$MOUNT_POINT"
```

```
STATUS=$?
```

```
if [ $STATUS -eq 0 ]; then
```

```
    echo "Partición montada correctamente en $MOUNT_POINT."
```

```
else
```

```
    echo "Error al montar la partición (código $STATUS)."
```

```
    exit $STATUS
```

```
fi
```





🔧 Pasos adicionales

Asegúrate de que el directorio existe y es accesible:

```
sudo mkdir -p /media/DATOS
sudo chown gerar_kde:gerar_kde /media/DATOS
Verifica permisos de ejecución del script:
```

```
sudo chmod +x /usr/local/bin/montar_datos_d.sh
```

Si ya tienes el servicio montar-datos-d.service, no necesitas modificar nada más.

¿Te gustaría que este script también registre mensajes en un archivo de log (por ejemplo, en /var/log/montar_datos.log)?

Tú dijiste:

si

ChatGPT Plus

Perfecto. A continuación te dejo el script actualizado que incluye registro en un archivo de log: /var/log/montar_datos.log.



Script final con logging: /usr/local/bin/montar_datos_d.sh

#!/bin/bash

```
MOUNT_POINT="/media/DATOS"
UUID="524F567904D39ECB"
FS_TYPE="ntfs3"
LOG_FILE="/var/log/montar_datos.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

```
log() {
    echo "[${TIMESTAMP}] $1" >> "$LOG_FILE"
}
```

```
# Comprobamos si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    log "La partición ya está montada en $MOUNT_POINT."
    exit 0
fi
```

```
# Comprobamos si el punto de montaje existe y tiene permisos
if [ ! -d "$MOUNT_POINT" ]; then
    log "Error: El directorio $MOUNT_POINT no existe."
    exit 1
fi
```

```
if [ ! -w "$MOUNT_POINT" ]; then
    log "Error: No tienes permisos de escritura en $MOUNT_POINT."
    exit 2
fi
```





<https://>



G-Mail



LinkedIn



Linktree



MI Web

```
# Intentamos montar
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID "$MOUNT_POINT"
STATUS=$?

if [ $STATUS -eq 0 ]; then
    log "Partición montada correctamente en $MOUNT_POINT."
else
    log "Error al montar la partición (código $STATUS)."
    exit $STATUS
fi
```



Asegúrate de que:

El archivo de log se pueda escribir:

```
sudo touch /var/log/montar_datos.log
sudo chown gerar_kde:gerar_kde /var/log/montar_datos.log
```

El script es ejecutable:

```
sudo chmod +x /usr/local/bin/montar_datos_d.sh
```

El servicio montar-datos-d.service sigue funcionando sin cambios.

¿Quieres que se eliminen entradas antiguas del log automáticamente cuando supere cierto tamaño?

Tú dijiste:

que se mantenga solo el ultimo log

ChatGPT Plus

Perfecto. Para que solo se conserve el último log, puedes modificar el script para sobrescribir el archivo cada vez que se ejecuta, en lugar de agregar contenido.



Script final con solo el último log: /usr/local/bin/montar_datos_d.sh

```
#!/bin/bash
```

```
MOUNT_POINT="/media/DATOS"
UUID="524F567904D39ECB"
FS_TYPE="ntfs3"
LOG_FILE="/var/log/montar_datos.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

```
# Sobrescribimos el archivo de log en cada ejecución
exec > "$LOG_FILE" 2>&1
echo "[$TIMESTAMP] Inicio del script de montaje"
```

```
# Comprobamos si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    echo "La partición ya está montada en $MOUNT_POINT."
    exit 0
fi
```

```
# Comprobamos si el punto de montaje existe y tiene permisos
```





```
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: El directorio $MOUNT_POINT no existe."
    exit 1
fi

if [ ! -w "$MOUNT_POINT" ]; then
    echo "Error: No tienes permisos de escritura en $MOUNT_POINT."
    exit 2
fi

# Intentamos montar
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID "$MOUNT_POINT"
STATUS=$?

if [ $STATUS -eq 0 ]; then
    echo "Partición montada correctamente en $MOUNT_POINT."
else
    echo "Error al montar la partición (código $STATUS)."
    exit $STATUS
fi
```

🔧 Lo que cambia:

La línea `exec > "$LOG_FILE" 2>&1` redirige toda la salida estándar y de error al log, sobrescribiéndolo.

Cada vez que el script se ejecuta, el archivo `/var/log/montar_datos.log` se reemplaza completamente por la salida actual.

¿Quieres que te prepare algo similar para el servicio que monta Google Drive con `rclone` también?

Tú dijiste:

como puedo saber el propietario de un archivo o directorio.

ChatGPT Plus

Puedes saber el propietario de un archivo o directorio usando el comando:

`ls -l /ruta/del/archivo_o_directorio`

Por ejemplo:

`ls -l /media/DATOS`

La salida será algo así:

```
drwxr-xr-x 2 gerar_kde gerar_kde 4096 may 11 10:00 /media/DATOS
```

La tercera columna es el usuario propietario (en este caso, `gerar_kde`).

La cuarta columna es el grupo propietario (también `gerar_kde` aquí).

Otra opción más directa:

`stat /ruta/del/archivo`

Ejemplo:



`stat /media/DATOS`

Te mostrará algo como:

```
File: /media/DATOS
Size: 4096    Blocks: 8      IO Block: 4096  directory
Device: 802h/2050d Inode: 131073    Links: 2
Access: 2025-05-11 ...
Modify: ...
Change: ...
Birth: -
Uid: ( 1000/ gerar_kde)  Gid: ( 1000/ gerar_kde)
Ahí ves directamente el UID y GID con los nombres asociados.
```

¿Te gustaría una forma de cambiar el propietario si no es el correcto?

Tú dijiste:

PUEDO HACER STAT A UN ARCHIVO, AL LOG POR EJEMPLO?

ChatGPT Plus

Sí, absolutamente. Puedes usar stat con cualquier archivo, incluido un archivo de log como /var/log/montar_datos.log.



Ejemplo:

`stat /var/log/montar_datos.log`

La salida te mostrará detalles como:

```
File: /var/log/montar_datos.log
Size: 134    Blocks: 8      IO Block: 4096  regular file
Device: 802h/2050d Inode: 123456    Links: 1
Access: 2025-05-11 11:05:20.000000000 +0200
Modify: 2025-05-11 11:04:59.000000000 +0200
Change: 2025-05-11 11:04:59.000000000 +0200
Birth: -
Uid: ( 1000/ gerar_kde)  Gid: ( 1000/ gerar_kde)
Ahí puedes verificar:
```

Propietario (Uid)

Grupo (Gid)

Tiempos de acceso, modificación y cambio

Tamaño y tipo de archivo



Si no tienes permisos

Si no eres propietario o no tienes acceso de lectura al archivo, verás un error como:

```
stat: cannot stat '/var/log/montar_datos.log': Permission denied
```





En ese caso, puedes hacer:

```
sudo stat /var/log/montar_datos.log
```

¿Quieres que este log sea visible también desde tu carpeta de usuario para mayor comodidad?

Tú dijiste:

modifica este script:

```
#!/bin/bash
#/usr/local/bin/mount_drive.sh
/usr/bin/rclone mount \
  gdrive: /home/gerar_kde/Drive \
  --vfs-cache-mode writes \
  --allow-other
```

ChatGPT Plus

Claro. A continuación te dejo una versión mejorada del script mount_drive.sh, con:

Verificación del punto de montaje

Registro en log

Control de errores

Ejecución en segundo plano si lo usas como servicio systemd

✓ Script mejorado: /usr/local/bin/mount_drive.sh

```
#!/bin/bash
# /usr/local/bin/mount_drive.sh

MOUNT_POINT="/home/gerar_kde/Drive"
LOG_FILE="/var/log/mount_drive.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

Redirige salida estándar y de errores al log (sobrescribe)

```
exec > "$LOG_FILE" 2>&1
echo "[$TIMESTAMP] Inicio del montaje de Google Drive"
```

Verifica si ya está montado

```
if mountpoint -q "$MOUNT_POINT"; then
    echo "Google Drive ya está montado en $MOUNT_POINT"
    exit 0
fi
```

Verifica que el punto de montaje exista

```
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: el directorio $MOUNT_POINT no existe."
    exit 1
fi
```





```
# Ejecuta rclone mount
/usr/bin/rclone mount \
  gdrive: "$MOUNT_POINT" \
  --vfs-cache-mode writes \
  --allow-other \
  --daemon

STATUS=$?

if [ $STATUS -eq 0 ]; then
  echo "Montaje exitoso de Google Drive en $MOUNT_POINT"
else
  echo "Error al montar Google Drive (código $STATUS)"
  exit $STATUS
fi
```

🔧 Asegúrate de:
Crear el archivo de log y dar permisos adecuados:

```
sudo touch /var/log/mount_drive.log
sudo chown gerar_kde:gerar_kde /var/log/mount_drive.log
```

Hacer el script ejecutable:

```
sudo chmod +x /usr/local/bin/mount_drive.sh
```

Este script es ideal para usarse desde un servicio systemd como el que mencionaste antes.
¿Quieres que también revise si rclone está instalado y dé un mensaje claro si falta?

Claro. A continuación te doy una versión mejorada y robusta del script para montar OneDrive con rclone, incluyendo:

Comprobaciones de existencia del punto de montaje

Verificación de si ya está montado

Ejecución como demonio (--daemon) para uso con systemd

Registro de salida y errores en un log (solo se guarda el último)

Verificación de que rclone esté instalado

✅ Script mejorado: /usr/local/bin/montar_onedrive.sh

```
#!/bin/bash
# /usr/local/bin/montar_onedrive.sh

MOUNT_POINT="/home/gerar_kde/One_Drive"
LOG_FILE="/var/log/mount_onedrive.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

Redirige toda la salida al archivo de log (sobrescribe)





```
exec > "$LOG_FILE" 2>&1

echo "[$TIMESTAMP] Iniciando montaje de OneDrive..."

# Verifica que rclone esté instalado
if ! command -v rclone >/dev/null 2>&1; then
    echo "Error: rclone no está instalado."
    exit 1
fi

# Verifica que el directorio de montaje existe
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: el directorio $MOUNT_POINT no existe."
    exit 2
fi

# Verifica si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    echo "OneDrive ya está montado en $MOUNT_POINT."
    exit 0
fi

# Ejecuta rclone mount
/usr/bin/rclone mount \
    onedrive: "$MOUNT_POINT" \
    --allow-other \
    --vfs-cache-mode writes \
    --daemon

STATUS=$?

if [ $STATUS -eq 0 ]; then
    echo "Montaje de OneDrive exitoso en $MOUNT_POINT."
else
    echo "Error al montar OneDrive (código $STATUS)."
    exit $STATUS
fi
```



Notas importantes:

He cambiado `--allow-non-empty` por `--allow-other`, que es más seguro en general. Si necesitas específicamente `--allow-non-empty`, puedes volver a ponerlo, pero ten en cuenta que permite montar sobre carpetas que ya tienen contenido, lo cual puede causar problemas.

`--daemon` permite que se use correctamente con `systemd` (el proceso no se queda colgado en primer plano).

El log se sobrescribe en cada ejecución: `/var/log/mount_onedrive.log`.



Configuración final

Asegúrate de que el script tenga permisos de ejecución:





```
sudo chmod +x /usr/local/bin/montar_onedrive.sh
```

Crea el archivo de log y da permisos:

```
sudo touch /var/log/mount_onedrive.log
```

```
sudo chown gerar_kde:gerar_kde /var/log/mount_onedrive.log
```

¿Quieres que te prepare también el archivo .service de systemd para este script?

Muy buena pregunta. Vamos a desglosarlo bien:

✚ ¿Qué hacen estas dos opciones?

--allow-other

Permite que otros usuarios además del que montó el sistema de archivos accedan a él.

Requiere que fusermount esté configurado con permisos para permitir eso (generalmente configurado en /etc/fuse.conf, debe tener la línea: user_allow_other sin comentar).

Útil si el contenido montado necesita ser accesible por más de un usuario (como cuando systemd ejecuta el script en segundo plano y el escritorio espera poder acceder al contenido montado).

--allow-non-empty

Permite montar el sistema de archivos sobre un directorio que ya contiene archivos.

⚠ Peligroso: Oculta el contenido existente del directorio (hasta que se desmonte).

Solo debes usarlo si sabes con certeza que el punto de montaje ya tiene contenido y eso no te importa (por ejemplo, scripts que se ejecutan sin garantizar un directorio vacío).

? ¿Necesitas alguno?

Depende de tu caso:

Situación

El directorio de montaje está vacío o controlado por el script

¿Necesitas --allow-non-empty?

✗ No

Necesitas --allow-other?

✓ Probablemente sí

Quieres montar sobre un directorio que podría tener contenido

¿Necesitas --allow-non-empty?

✓ Sí

Necesitas --allow-other?

✓ Si otros usuarios necesitan acceder

Solo accede tu usuario al montaje y el script se ejecuta como tú

¿Necesitas --allow-non-empty?

✗ No

Necesitas --allow-other?

✗ No necesariamente





El montaje falla y tu escritorio/otro usuario no puede leerlo

¿Necesitas --allow-non-empty?

☒ No

Necesitas --allow-other?

☒ Sí

¿Escribirá en el log si falla por estas flags?

Sí, si usas `exec > log` como tienes en el script, cualquier error del montaje —incluyendo uno provocado por `--allow-other` o `--allow-non-empty`— será registrado, por ejemplo:

Fatal error: mountpoint is not empty: /home/gerar_kde/Drive

Fatal error: failed to mount FUSE fs: exit status 1

o

Fatal error: cannot mount with allow-other: fusermount: option allow_other only allowed if 'user_allow_other' is set in /etc/fuse.conf

☒ Recomendaciones

Evita `--allow-non-empty` a menos que sepas lo que haces.

Activa `--allow-other` solo si necesitas que otros usuarios accedan al montaje.

Revisa el archivo `/etc/fuse.conf` para permitir `--allow-other`:

`sudo nano /etc/fuse.conf`

Asegúrate de que la línea:

`user_allow_other`

no esté comentada (sin `#` al inicio). Guarda y cierra.

¿Quieres que te ayude a detectar automáticamente en el script si el punto de montaje no está vacío y mostrar un aviso?

EL OMONTAJE DE dRIVE Y DE oNEdRIVE ME HA DADO EL SIGUIENTE ERROR:

gerar_kde@Lap-Linux:~/Code\$ cat /var/log/montar_drive.log

[2025-05-11 22:01:52] Inicio del montaje de Google Drive

2025/05/11 22:01:54 ERROR : Daemon timed out. Failed to terminate daemon pid 5554: os: process already finished

2025/05/11 22:01:54 CRITICAL: Fatal error: daemon exited with error code 1

Error al montar Google Drive (código 1)

QUE PUEDE SER?

ChatGPT Plus

El error que estás viendo al intentar montar Google Drive con Rclone indica que el daemon (el proceso en segundo plano) no pudo iniciarse correctamente o se cerró inesperadamente. Aquí está el análisis del mensaje de error:





ERROR : Daemon timed out. Failed to terminate daemon pid 5554: os: process already finished
CRITICAL: Fatal error: daemon exited with error code 1
Esto sugiere que:

El proceso del daemon de Rclone (probablemente rclone mount) se inició pero terminó inesperadamente.

Rclone intentó terminar el proceso (quizás porque detectó un error), pero ya había terminado por su cuenta.

El código de error 1 generalmente indica un fallo genérico en Rclone.

Posibles causas:

Fallo en la autenticación con Google Drive

Si el token expiró o hay un error en la configuración remota (rclone config), el montaje fallará.

Permisos insuficientes

Si estás ejecutando el script con un usuario que no tiene los permisos necesarios para montar FUSE o crear directorios en el destino.

FUSE no está correctamente instalado

Rclone usa FUSE para montar sistemas de archivos. Si fuse o fusermount no están disponibles o bien configurados, puede fallar.

Conflicto con un montaje previo

Puede que el punto de montaje ya esté ocupado o haya quedado "colgado" de un intento anterior.

Soluciones recomendadas:

Verifica tu configuración remota:

rclone config show

Asegúrate de que el remoto de Google Drive esté bien configurado.

Prueba montar manualmente con rclone mount para ver más detalles:

rclone mount gdrive: ~/mnt/gdrive --vfs-cache-mode writes --log-file=rclone.log -vv

Esto generará un log más detallado (rclone.log).

Revisa si fusermount está instalado:

which fusermount

Si no está, instala FUSE:





```
sudo apt install fuse
```

Asegúrate de que el punto de montaje no esté ya ocupado:

```
mount | grep ~/mnt/gdrive
```

O desmonta si hace falta:

```
fusermount -u ~/mnt/gdrive
```

Ejecuta como superusuario solo si es estrictamente necesario, pero generalmente Rclone funciona bien como usuario normal si los permisos están bien configurados.

¿Podrías mostrarme el comando exacto que usas para montar Google Drive o el contenido del script que lo hace? Así te doy una ayuda más específica.

¡Exactamente! Lo has pensado muy bien. Sí, un montaje previo que no se haya desmontado correctamente puede causar conflictos cuando se vuelve a intentar montar el mismo punto en el siguiente arranque o sesión.

Por eso, crear un script que desmonte Google Drive y OneDrive al apagar o reiniciar el sistema es una buena práctica, sobre todo si los estás montando con rclone mount, que depende de FUSE.

¿Por qué puede fallar el montaje si no se desmonta?
fusermount monta el sistema en modo usuario.

Si no se desmonta correctamente (por ejemplo, si apagas bruscamente), el punto de montaje puede quedar en un estado inconsistente.

Al reiniciar, si intentas montar de nuevo en el mismo directorio, puede dar error porque "ya está ocupado" o "no está vacío", aunque no lo parezca.

¿Cómo automatizar el desmontaje al apagar?
Opción 1: Usar un script en /lib/systemd/system-shutdown/
Crea el script:

```
sudo nano /lib/systemd/system-shutdown/desmontar_drives
```

Contenido del script:

```
#!/bin/bash
```

```
fusermount -u /ruta/a/tu/punto/de/montaje/drive
```

```
fusermount -u /ruta/a/tu/punto/de/montaje/onedrive
```

Reemplaza las rutas por tus directorios reales, como ~/mnt/drive y ~/mnt/onedrive.

Hazlo ejecutable:

```
sudo chmod +x /lib/systemd/system-shutdown/desmontar_drives
```

Este script se ejecutará al apagar o reiniciar el sistema y desmontará ambos drives.

Opción 2: Usar systemd con unidades de desmontaje personalizadas

Si prefieres algo más limpio o quieres gestionar esto por servicio, también puedes crear unidades systemd que desmonten Drive al apagar, pero la opción anterior es más rápida para lo que buscas.





¿Quieres que te genere un script completo para tu sistema, incluyendo detección de si el punto está montado antes de desmontar?

Tú dijiste:
si

ChatGPT Plus

Perfecto. Aquí tienes un script completo para desmontar Google Drive y OneDrive al apagar o reiniciar tu sistema. Este script incluye verificación para asegurarse de que el punto de montaje esté montado antes de intentar desmontarlo.

✓ Pasos para instalar el script
Crea el archivo de apagado del sistema:

```
sudo nano /lib/systemd/system-shutdown/desmontar_drives
```

Pega este contenido:

```
#!/bin/bash

# Rutas de montaje - cambia si son diferentes
DRIVE_MOUNT="/home/gerar_kde/mnt/drive"
ONEDRIVE_MOUNT="/home/gerar_kde/mnt/onedrive"

log="/var/log/desmontar_drives.log"
echo "`date`" Iniciando desmontaje de unidades" >> "$log"

desmontar_si_montado() {
    local punto="$1"
    if mountpoint -q "$punto"; then
        echo "`date`" Desmontando $punto" >> "$log"
        fusermount -u "$punto"
        if [ $? -eq 0 ]; then
            echo "`date`" $punto desmontado correctamente." >> "$log"
        else
            echo "`date`" Error al desmontar $punto" >> "$log"
        fi
    else
        echo "`date`" $punto no está montado. Nada que desmontar." >> "$log"
    fi
}

desmontar_si_montado "$DRIVE_MOUNT"
desmontar_si_montado "$ONEDRIVE_MOUNT"

echo "`date`" Desmontaje completado" >> "$log"
```

Hazlo ejecutable:

```
sudo chmod +x /lib/systemd/system-shutdown/desmontar_drives
```





Verificación

Después de reiniciar, puedes revisar el log para confirmar que se ejecutó correctamente:

```
cat /var/log/desmontar_drives.log
```

¿Te gustaría que también te prepare los scripts de montaje para Google Drive y OneDrive para que funcionen al inicio del sistema con systemd?

Tú dijiste:

el siguiente script:

```
"
```

```
#!/bin/bash
```

```
# Rutas de montaje - cambia si son diferentes
```

```
DRIVE_MOUNT="/home/gerar_kde/Drive"
```

```
ONEDRIVE_MOUNT="/home/gerar_kde/One_Drive"
```

```
log="/var/log/desmontar_drives.log"
```

```
echo "[date] Iniciando desmontaje de unidades" >> "$log"
```

```
desmontar_si_montado() {
```

```
local punto="$1"
```

```
if mountpoint -q "$punto"; then
```

```
echo "[date] Desmontando $punto" >> "$log"
```

```
fusermount -u "$punto"
```

```
if [ $? -eq 0 ]; then
```

```
echo "[date] $punto desmontado correctamente." >> "$log"
```

```
else
```

```
echo "[date] Error al desmontar $punto" >> "$log"
```

```
fi
```

```
else
```

```
echo "[date] $punto no está montado. Nada que desmontar." >> "$log"
```

```
fi
```

```
}
```

```
desmontar_si_montado "$DRIVE_MOUNT"
```

```
desmontar_si_montado "$ONEDRIVE_MOUNT"
```

```
echo "[date] Desmontaje completado" >> "$log"
```

```
"
```

me saca el siguiente error:

```
"
```

```
cat /var/log/desmontar_drives.log
```

```
[lun 12 may 2025 01:53:45 CEST] Iniciando desmontaje de unidades
```

```
[lun 12 may 2025 01:53:45 CEST] /home/gerar_kde/Drive no está montado. Nada que desmontar.
```

```
[lun 12 may 2025 01:53:45 CEST] /home/gerar_kde/One_Drive no está montado. Nada que desmontar.
```

```
[lun 12 may 2025 01:53:45 CEST] Desmontaje completado
```

```
[lun 12 may 2025 02:30:54 CEST] Iniciando desmontaje de unidades
```





```
[lun 12 may 2025 02:30:54 CEST] /home/gerar_kde/Drive no está montado. Nada que desmontar.
[lun 12 may 2025 02:30:54 CEST] Desmontando /home/gerar_kde/One_Drive
[lun 12 may 2025 02:30:54 CEST] /home/gerar_kde/One_Drive desmontado correctamente.
[lun 12 may 2025 02:30:54 CEST] Desmontaje completado
"
```

porque deusmonta drive y no onedrive?

ChatGPT Plus

Gracias por compartir el log. El comportamiento del script parece correcto: intenta desmontar ambos puntos de montaje, pero solo desmonta OneDrive porque es el único que está montado en ese momento.

En el primer intento:

```
/home/gerar_kde/Drive no está montado. Nada que desmontar.
/home/gerar_kde/One_Drive no está montado. Nada que desmontar.
En el segundo intento:
```

```
/home/gerar_kde/Drive no está montado. Nada que desmontar.
/home/gerar_kde/One_Drive desmontado correctamente.
```

Esto sugiere que Google Drive (/home/gerar_kde/Drive) no estaba montado cuando se ejecutó el script, mientras que OneDrive sí lo estaba.

✅ Posibles causas por las que Google Drive no aparece montado al apagar:
No llegó a montarse correctamente durante la sesión (quizás por el error anterior de Rclone).

Se desmontó automáticamente por algún otro script, fallo, o cierre de sesión antes del apagado.

El directorio estaba montado pero el proceso de FUSE se cayó. En ese caso, mountpoint -q no lo detecta como montado.

🔍 Para investigar:

Durante una sesión donde sabes que Drive está montado, ejecuta:

```
mount | grep /home/gerar_kde/Drive
o
```

```
findmnt /home/gerar_kde/Drive
```

Si devuelve algo, entonces está montado correctamente y el script debería poder desmontarlo.

Si no devuelve nada, probablemente el montaje nunca ocurrió correctamente, o se cayó el proceso que lo mantiene.

🔧 Alternativa más robusta para detectar montajes FUSE (como los de Rclone):
Cambia esta línea:

```
if mountpoint -q "$punto"; then
```





Por esta otra más robusta:

```
if grep -qs "$punto" /proc/mounts; then
```

Esto detecta montajes incluso si mountpoint falla en reconocer algunos sistemas como montados (común con rclone mount y FUSE).

¿Quieres que te envíe el script ya corregido con esta mejora?

Tú dijiste:

devuelve:

"

```
mount | grep /home/gerar_kde/Drive
```

```
gdrive: on /home/gerar_kde/Drive type fuse.rclone
```

```
(rw,nosuid,nodev,relatime,user_id=1001,group_id=1001)"
```

ChatGPT Plus

Perfecto, gracias por compartir la salida.

Esto confirma que Google Drive está efectivamente montado usando rclone mount con FUSE, pero el comando mountpoint -q no lo detecta correctamente, lo cual es una limitación conocida con montajes tipo fuse.rclone.



Solución: Usar /proc/mounts en lugar de mountpoint

Aquí tienes el script corregido que detecta correctamente montajes de tipo FUSE (como los de Rclone):



Script actualizado (/lib/systemd/system-shutdown/desmontar_drives):

```
#!/bin/bash
```

```
# Rutas de montaje - cambia si son diferentes
```

```
DRIVE_MOUNT="/home/gerar_kde/Drive"
```

```
ONEDRIVE_MOUNT="/home/gerar_kde/One_Drive"
```

```
log="/var/log/desmontar_drives.log"
```

```
echo "`date`" Iniciando desmontaje de unidades" >> "$log"
```

```
desmontar_si_montado() {
```

```
    local punto="$1"
```

```
    if grep -qs "$punto" /proc/mounts; then
```

```
        echo "`date`" Desmontando $punto" >> "$log"
```

```
        fusermount -u "$punto"
```

```
        if [ $? -eq 0 ]; then
```

```
            echo "`date`" $punto desmontado correctamente." >> "$log"
```

```
        else
```

```
            echo "`date`" Error al desmontar $punto" >> "$log"
```

```
        fi
```

```
    else
```

```
        echo "`date`" $punto no está montado. Nada que desmontar." >> "$log"
```





```

    fi
}

desmontar_si_montado "$DRIVE_MOUNT"
desmontar_si_montado "$ONEDRIVE_MOUNT"

echo "[`date`] Desmontaje completado" >> "$log"

```

🔧 Actualiza permisos si no lo has hecho aún:

```
sudo chmod +x /lib/systemd/system-shutdown/desmontar_drives
```

Con este cambio, el script detectará correctamente si Drive está montado y lo desmontará como debe ser.

¿Te gustaría también automatizar el montaje de Drive y OneDrive al inicio del sistema con systemd para completar el flujo?

Ruta: `/usr/local/bin/file-name.sh`

```
#!/bin/bash
#/usr/local/bin/mount_drive.sh

rclone mount --allow-non-empty gdrive: /home/gerar_kde/Drive &
exit

```

&	exit	& hace que el proceso se ejecute en segundo plano, y luego haces exit, lo que mata el proceso del script junto con el montaje, porque Type=simple espera que el proceso continúe.
--allow-non-empty		no es recomendable, puede causar errores si la carpeta ya tiene contenido.
--vfs-cache-mode		es obligatorio para muchas operaciones con rclone mount.





CORREGIDO:

```
#!/bin/bash
#/usr/local/bin/mount_drive.sh

/usr/bin/rclone mount \
gdrive: /home/gerar_kde/Drive \
--vfs-cache-mode writes \
--allow-other
```



VERSION MEJORADA DEL SCRIPT

- Verificación del punto de montaje
- Registro en log
- Control de errores
- Ejecución en segundo plano si lo usas como servicio systemd

```
#!/bin/bash
# /usr/local/bin/mount_drive.sh

MOUNT_POINT="/home/gerar_kde/Drive"
LOG_FILE="/var/log/mount_drive.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

Redirige salida estándar y de errores al log (sobrescribe)

```
exec > "$LOG_FILE" 2>&1
echo "[${TIMESTAMP}] Inicio del montaje de Google Drive"
```

Verifica si ya está montado

```
if mountpoint -q "$MOUNT_POINT"; then
    echo "Google Drive ya está montado en $MOUNT_POINT"
    exit 0
fi
```

Verifica que el punto de montaje exista

```
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: el directorio $MOUNT_POINT no existe."
    exit 1
fi
```

Ejecuta rclone mount

```
/usr/bin/rclone mount \
gdrive: "$MOUNT_POINT" \
--vfs-cache-mode writes \
--allow-other \
--daemon
```

```
STATUS=$?
```

```
if [ $STATUS -eq 0 ]; then
```





```
    echo "Montaje exitoso de Google Drive en $MOUNT_POINT"
else
    echo "Error al montar Google Drive (código $STATUS)"
    exit $STATUS
fi
```



ASEGURATE DE:

- Crear el archivo de log y dar permisos adecuados:

```
sudo touch /var/log/montar_drive.log
sudo chown gerar_kde:gerar_kde /var/log/montar_drive.log
```

- Hacer el script ejecutable:

```
sudo chmod +x /usr/local/bin/mount_drive.sh
```

■ CREAR ARCHIVO DE SERVICIO SYSTEMD

Ruta: `/etc/systemd/system/file-name.service`

```
[Unit]
Description=Montar Google Drive al inicio
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=gerar_kde
ExecStart=/usr/local/bin/montar_drive.sh
Restart=on-failure

[Install]
WantedBy=default.target
```

■ HABILITAR EL SERVICIO AL INICIO:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar_drive.service
```

■ PRUEBA EL SCRIPT SIN REINICIAR:

```
sudo systemctl start montar_drive.service
```





REVISAR LOGS DEL SERVICIO:

```
journalctl -u montar_drive.service --no-pager
```

Ahí verás si falla algo como permisos, rutas, o errores de rclone.

RECARGA Y REINICIA EL SERVICIO

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar_drive.service
```

MONTAJE UNIDAD REMOTA MS ONE-DRIVE

CREAR SCRIPT DE MONTAJE

Ruta: /usr/local/bin/montar_onedrive.sh

```
#!/bin/bash
# /usr/local/bin/montar_onedrive.sh

/usr/bin/rclone mount \
onedrive: /home/gerar_kde/One_Drive \
--allow-non-empty
```

--allow-non-empty

no es recomendable, puede causar errores si la carpeta ya tiene contenido, pero One Drive parece que lo use así

DARLE PERMISOS DE EJECUCION:

```
sudo chmod +x /usr/local/bin/montar_onedrive.sh
```

CREAR ARCHIVO DE SERVICIO SYSTEMD

Ruta: /etc/systemd/system/montar_onedrive.service

```
[Unit]
Description=Montar MS One Drive al inicio
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
```





```
User=gerar_kde
ExecStart=/usr/local/bin/montar_onedrive.sh
Restart=on-failure

[Install]
WantedBy=default.target
```

HABILITAR EL SERVICIO AL INICIO:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar_onedrive.service
```

PRUEBA EL SCRIPT SIN REINICIAR:

```
sudo systemctl start montar_onedrive.service
```

REVISAR LOGS DEL SERVICIO:

```
journalctl -u montar_onedrive.service --no-pager
```

Ahí verás si falla algo como permisos, rutas, o errores de rclone.

RECARGA Y REINICIA EL SERVICIO

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar_onedrive.service
```

MONTAJE UNIDAD DE PARTICIÓN /DATOS

CREAR DIRECTORIO DE MONTAJE

```
sudo mkdir -p /media/DATOS
sudo chown gerar_kde:gerar_kde /media/DATOS
```

chown

controlador moderno para NTFS (más rápido que ntfs-3g en kernels nuevos).

gerar_kde:gerar_kde:

asigna como propietario al usuario gerar_kde y también al grupo gerar_kde





CREAR SCRIPT DE MONTAJE DE UNIDAD



SCRIPT MEJORADO PARA:

- Evitar montar si ya está montada
- Verificar permisos de escritura del punto de montaje
- Informar claramente en caso de error

Ruta: `/usr/local/bin/montar_datos.sh`

```
#!/bin/bash
#/usr/local/bin/montar_datos.sh

MOUNT_POINT="/media/DATOS"
UUID="524F567904D39ECB"
FS_TYPE="ntfs3"

# Comprobamos si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    echo "La partición ya está montada en $MOUNT_POINT."
    exit 0
fi

# Comprobamos si el punto de montaje existe y tiene permisos
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: El directorio $MOUNT_POINT no existe."
    exit 1
fi

if [ ! -w "$MOUNT_POINT" ]; then
    echo "Error: No tienes permisos de escritura en $MOUNT_POINT."
    exit 2
fi

# Intentamos montar
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID
"$MOUNT_POINT"
STATUS=$?

if [ $STATUS -eq 0 ]; then
    echo "Partición montada correctamente en $MOUNT_POINT."
else
    echo "Error al montar la partición (código $STATUS)."
    exit $STATUS
fi
```

ntfs3	controlador moderno para NTFS (más rápido que ntfs-3g en kernels nuevos).
uid=1000,gid=1000	asegura que el usuario (gerar_kde) tenga acceso. Tu usuario gerar_kde tiene UID 1000 (casi siempre es así para el primer usuario). Puedes verificarlo con: <code>id -u gerar_kde</code>
id -u gerar_kde	
umask=022	da permisos de lectura para todos y escritura sólo para el dueño.





■ DARLE PERMISOS DE EJECUCION:

```
sudo chmod +x /usr/local/bin/montar_datos.sh
```

■ CREAR ARCHIVO DE SERVICIO SYSTEMD

Ruta: /etc/systemd/system/montar_datos.service

```
[Unit]
Description=Montar partición Datos (NTFS)
After=local-fs.target

[Service]
Type=oneshot
ExecStart=/usr/local/bin/montar_datos.sh
RemainAfterExit=true

[Install]
WantedBy=multi-user.target
```

Type=oneshot

porque solo monta una vez.

RemainAfterExit=true

para que systemd lo considere "activo" después de ejecutar.

■ HABILITAR EL SERVICIO AL INICIO Y PROBAR :

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable montar_datos.service
sudo systemctl start montar_datos.service
```

■ PRUEBA EL SCRIPT SIN REINICIAR:

```
sudo systemctl start montar_drive.service
```

■ VERIFICA SI ESTA MONTADO:

```
mount | grep Datos-D
```



SCRIPT MEJORADO DE MONTAJE DE UNIDAD

- incluye registro en un archivo de log: /var/log/montar_datos.log.





<https://>



G-Mail



LinkedIn



Linktree



MI Web

```
#!/bin/bash
#/usr/local/bin/montar_datos.sh

MOUNT_POINT="/media/DATOS"
UUID="524F567904D39ECB"
FS_TYPE="ntfs3"
LOG_FILE="/var/log/montar_datos.log"
TIMESTAMP="$(date '+%Y-%m-%d %H:%M:%S')"
```

```
log() {
    echo "[$TIMESTAMP] $1" >> "$LOG_FILE"
}

# Comprobamos si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    log "La partición ya está montada en $MOUNT_POINT."
    exit 0
fi

# Comprobamos si el punto de montaje existe y tiene permisos
if [ ! -d "$MOUNT_POINT" ]; then
    log "Error: El directorio $MOUNT_POINT no existe."
    exit 1
fi

if [ ! -w "$MOUNT_POINT" ]; then
    log "Error: No tienes permisos de escritura en $MOUNT_POINT."
    exit 2
fi

# Intentamos montar
mount -t $FS_TYPE -o rw,uid=1000,gid=1000,umask=022 UUID=$UUID "$MOUNT_POINT"
STATUS=$?

if [ $STATUS -eq 0 ]; then
    log "Partición montada correctamente en $MOUNT_POINT."
else
    log "Error al montar la partición (código $STATUS)."
    exit $STATUS
fi
```

Asegúrate de que El archivo de log se pueda escribir:

```
sudo touch /var/log/montar_datos.log
sudo chown gerar_kde:gerar_kde /var/log/montar_datos.log
```

El script es ejecutable:

```
sudo chmod +x /usr/local/bin/montar_datos.sh
```



ERROR

```
cat /var/log/montar_onedrive.log
[2025-05-11 22:00:17] Iniciando montaje de OneDrive...
2025/05/11 22:00:19 ERROR : Daemon timed out. Failed to terminate daemon pid
3239: os: process already finished
2025/05/11 22:00:19 CRITICAL: Fatal error: daemon exited with error code 1
Error al montar OneDrive (código 1).
```

ERROR AL INTENTAR MONTAR GOOGLE DRIVE CON RCLONE

El error que estás viendo al intentar montar Google Drive con Rclone indica que el daemon (el proceso en segundo plano) no pudo iniciarse correctamente o se cerró inesperadamente. Aquí está el análisis del mensaje de error:

ERROR : Daemon timed out. Failed to terminate daemon pid 5554: os: process already finished
CRITICAL: Fatal error: daemon exited with error code 1

Esto sugiere que:

- El proceso del daemon de Rclone (probablemente rclone mount) se inició pero terminó inesperadamente.
- Rclone intentó terminar el proceso (quizás porque detectó un error), pero ya había terminado por su cuenta.
- El código de error 1 generalmente indica un fallo genérico en Rclone.

Posibles causas:

- Fallo en la autenticación con Google Drive
- Si el token expiró o hay un error en la configuración remota (rclone config), el montaje fallará.
- Permisos insuficientes
- Si estás ejecutando el script con un usuario que no tiene los permisos necesarios para montar FUSE o crear directorios en el destino.
- FUSE no está correctamente instalado
- Rclone usa FUSE para montar sistemas de archivos. Si fuse o fusemount no están disponibles o bien configurados, puede fallar.
- **Conflicto con un montaje previo**
- **Puede que el punto de montaje ya esté ocupado o haya quedado "colgado" de un intento anterior.**

MONTAR UNIDADES CON SYSTEMD

Montar las unidades directamente sólo con un script de servicio, no con script de montaje + script de servicio

REQUISITOS PREVIOS

ASEGURATE DE TENER:

- Rclone funcionando y configurado (con remotos como gdrive: y onedrive:).
- Directorios de montaje vacíos y existentes (ej: ~/mnt/drive, ~/mnt/onedrive).



- FUSE instalado.

CREAR LOS SERVICIOS SYSTEMD

```
cat /var/log/desmontar_drives.log
```



Google Drive

Ruta: /etc/systemd/system/rcclone-gdrive.service

```
[Unit]
Description=Montar Google Drive con Rclone
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=gerar_kde
ExecStart=/usr/bin/rclone mount gdrive: /home/gerar_kde/Drive \
  --vfs-cache-mode writes \
  --dir-cache-time 1h \
  --poll-interval 1m \
  --log-level INFO \
  --log-file /var/log/rclone-gdrive.log
ExecStop=/bin/fusermount -u /home/gerar_kde/Drive
Restart=on-failure

[Install]
WantedBy=default.target
```



MS ONE DRIVE

Ruta: /etc/systemd/system/rcclone-onedrive.service

```
[Unit]
Description=Montar OneDrive con Rclone
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=gerar_kde
ExecStart=/usr/bin/rclone mount onedrive: /home/gerar_kde/One_Drive \
  --vfs-cache-mode writes \
  --dir-cache-time 1h \
  --poll-interval 1m \
  --log-level INFO \
  --log-file /var/log/rclone-onedrive.log
```



```
ExecStop=/bin/fusermount -u /home/gerar_kde/One_Drive
Restart=on-failure

[Install]
WantedBy=default.target
```

■ RECARGAR SYSTEMD Y HABILITAR LOS SERVICIOS

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload

sudo systemctl enable rclone-gdrive.service
sudo systemctl enable rclone-onedrive.service
```



OPCIONAL: PARA MONTARLOS AHORA SIN REINICIAR:

```
sudo systemctl start rclone-gdrive.service
sudo systemctl start rclone-onedrive.service
```

■ LOGS Y MONITOREO

Puedes ver si funcionó con:

```
systemctl status rclone-gdrive.service
cat /var/log/rclone-gdrive.log
```



DESMONTAR DRIVE Y ONE DRIVE AL APAGAR/REINICIAR



CREAR SCRIPT DE DESMONTAJE DE UNIDAD

Script completo que desmonta las unidades de Google Drive y OneDrive solo si están montadas, y que se ejecuta automáticamente al apagar o reiniciar el sistema:

Ruta **/lib/systemd/system-shutdown/**

```
#!/bin/bash

# Rutas a tus puntos de montaje
DRIVE_MOUNT="/home/gerar_kde/Drive"
ONEDRIVE_MOUNT="/home/gerar_kde/One_Drive"

echo "[Shutdown] Intentando desmontar Google Drive y OneDrive..." >>
/var/log/desmontar_drives.log

# Función para desmontar si está montado
desmontar_si_montado() {
```



```
MOUNT_PATH="$1"
if mount | grep -q "$MOUNT_PATH"; then
    fusermount -u "$MOUNT_PATH" && echo "Desmontado: $MOUNT_PATH"
>> /var/log/desmontar_drives.log || echo "Fallo al desmontar:
$MOUNT_PATH" >> /var/log/desmontar_drives.log
else
    echo "No montado: $MOUNT_PATH" >> /var/log/desmontar_drives.log
fi
}

desmontar_si_montado "$DRIVE_MOUNT"
desmontar_si_montado "$ONEDRIVE_MOUNT"

exit 0
```

DAR PERMISOS DE EJECUCION

```
sudo chmod +x /lib/systemd/system-shutdown/desmontar_drives
```

PRUEBA EL SCRIPT SIN APAGAR:

```
sudo /lib/systemd/system-shutdown/desmontar_drives
```

REVISAR EL LOG:

```
cat /var/log/desmontar_drives.log
```

En este caso, no es necesario que el script tenga la extensión .sh.

En Linux, los scripts que se ejecutan automáticamente por el sistema (como los de /lib/systemd/system-shutdown/) son identificados por su nombre y permisos, no por la extensión.

Lo importante es que:

- El archivo sea ejecutable (chmod +x).
- Tenga una shebang válida (#!/bin/bash al principio).
- Esté en el directorio correcto (como /lib/systemd/system-shutdown/).

Aunque puede ser **desmontar_drives.sh**, Y También funcionará perfectamente





<https://>



G-Mail



LinkedIn



Linktree



MI Web



TÍTULO-Y [11 / 1⁵] IMG 0⁴⁵ x 0³⁰

