



# SERVICIOS [SYSTEMD / SYSTEM-SHUTDOWN]



## OTROS DOCUMENTOS DE LINUX

[SERVICIOS: MONTAR/DESMONTAR UDS](#)[BASH\\_COMANDS](#)[BASH COMANDS - Extended](#)

## INDICE DOCUMENTO

SERVICIO DE INICIO - I	Script .sh	Permisos	SERVICIO DE APAGADO - I	Script .sh	Permisos
	Script SYSTEMD	Habilitar Serv. Reiniciar Serv.		Script SHUTDOWN	Habilitar Serv. Reiniciar Serv.
SERVICIO DE INICIO - II	Servicio Sin Archivo .sh		SERVICIO DE APAGADO - II	Servicio Sin Archivo .sh	
SERVICIO DE INICIO - III	Archivo /etc/fstab				
OPERACIONES CON SERV	Ver Estado de Serv	Listar Serv Habilitados			



## CREAR UN SERVICIO



### SERVICIO SYSTEMD

- 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 SCRIPT .SH



### SERVICIO SYSTEMD - SHUTDOWN

- CREAR SCRIPT
- DARLE PERMISOS EJECUCION
- CREAR ARCHIVO DE SERVICIO SHUTDOWN
- HABILITAR EL SERVICIO
- PROBAR EL SCRIPT SIN APAGAR
- REVISAR LOGS DEL SERVICIO
- RECARGAR Y REINICIAR EL SERVICIO
- SERVICIO SYSTEMD-SHUTDOWN SIN SCRIPT .SH





## SERVICIOS SYSTEMD DE INICIO DE SISTEMA

### SERVICIO SYSTEMD CON SCRIPT .SH



#### CREAR SCRIPT .SH

Creamos un script de bash en usr/bin que contenga las órdenes que queremos que se ejecuten al arrancar el sistema

Ruta `/usr/local/bin/`

Tipo Guardarlo como `.sh`

```
vim /usr/local/bin/file-name.sh
```

```
#!/bin/bash
sudo apt update
sudo apt upgrade
exit
```



#### DARLE PERMISOS EJECUCION

Le damos permisos de ejecución:

```
chmod +x file-name.sh
```

A veces tenemos que cambiar también el propietario, porque lo crea como 'root' y hay que pasarlo al usuario.

El 1º lo añade al usuario y e 2º al grupo, en mi caso: `gerar_kde:gerar_kde`

```
Chown user-name:group-name file-name.sh
```

podemos probarlo para ver que vaya bien

```
./file-name.sh
```



#### SCRIPT DE SERVICIO SYSTEMD

Creamos también un script de servicio en `/etc/systemd/system/` que contenga la orden de ejecución del archivo sh

Ruta `/etc/systemd/system/`

Tipo Guardarlo como `service-name.service`

```
sudo vim /etc/systemd/system/service-name.service
```

```
[Unit]
Description=Montar unidad remota al inicio
After=network-online.target
```



```
Wants=network-online.target
[Service]
Type=simple
User=user-name
ExecStart=/usr/local/bin/file-name.sh
Restart=on-failure
[Install]
WantedBy=default.target
```



## HABILITAR EL SERVICIO

Habilitamos el Servicio para que se inicie al arrancar:

```
sudo systemctl daemon-reexec
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable service-name.service
```



## PROBAR SCRIPT SIN REINICIAR

Probamos si todo funciona correctamente

```
sudo systemctl start service-name.service
```



## REVISAR EL PROCESO

Después de ejecutar el servicio podemos revisar si falla algo como permisos, rutas, o errores de rclone, con journalctl :

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

Podemos crear un Archivo log en el que revisar como ha ido el proceso, o ver las causas si falla y que se sobrescriba cada vez que se ejecute el servicio.



## CREAR ARCHIVO DE LOG

Podemos crear un Archivo log en el que revisar como ha ido el proceso, o ver las causas si falla.

Ruta **/var/log/**

Tipo Guardarlo como **log-name.log**

```
sudo touch /var/log/log-name.log
```

Asegúrate de que el archivo de log se pueda escribir, para lo que cambiamos el dueño y el grupo, porque lo crea como 'root'

El 1º lo añade al usuario y e 2º al grupo, en mi caso:



```
gerar_kde:gerar_kde
```

```
chown user-name:group-name log-name.log
```



## RECARGAR Y REINICIAR SERVICIO

Habilitamos el Servicio para que se inicie al arrancar:

```
sudo systemctl daemon-reexec
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable service-name.service
```



## CAMBIOS POSTERIORES

Si queremos hacer cambios en el servicio, modificaremos el script .sh y sobre el servicio no necesitas modificar nada más, únicamente volver a reiniciarlo

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

## SERVICIO SYSTEMD SIN SCRIPT .SH

El proceso es el mismo, lo único es que no creamos un script de bash en un archivo y luego lo ejecutamos desde un archivo .service, si no que directamente creamos el .service.



## CREAR SCRIPT DE SERVICIO SYSTEMD

El proceso es el mismo, lo único es que no creamos un script de bash en un archivo y luego lo ejecutamos desde un archivo .service, si no que directamente creamos el .service.

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

En Linux, los scripts que se ejecutan automáticamente por el sistema son identificados por su nombre y permisos, no por la extensión. Scripts de

**/etc/systemd/system/**

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

**Lo importante es que:**

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

Creamos también un script de servicio en /etc/systemd/system/ que contenga la orden a ejecutar

Ruta **/etc/systemd/system/**

Tipo Guardarlo como **file-name** o como **file-name.sh**



```
sudo vim /etc/systemd/system/file-name
```

```
#!/bin/bash
/etc/systemd/system/file-name
sudo apt update
sudo apt upgrade
exit
```



## DARLE PERMISOS EJECUCION

Le damos permisos de ejecución:

```
chmod +x file-name
```

podemos probarlo para ver que vaya bien

```
./file-name
```



## HABILITAR EL SERVICIO

Habilitamos el Servicio para que se inicie al arrancar:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable file-name
```



## PROBAR SCRIPT SIN REINICIAR

Probamos si todo funciona correctamente

```
sudo systemctl start file-name
```



## RECARGAR Y REINICIAR SERVICIO

Habilitamos el Servicio para que se inicie al arrancar:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable file-name
```

## ARCHIVO /ETC/FSTAB

Para Unidades físicas es mejor hacer el montaje desde este archivo de tal manera que se inicie siempre al arrancar



GIT HUB LINK

GERARDO MTNZ - Dev



REPOSITORY LINK

LINUX PUBLIC REPO



G-Mail



LinkedIn



Linktree



G-Mtnz Web



## CONFIGURAR EL ARCHIVO FSTAB

Ruta **/etc/fstab**

Tipo Sin extensión

**sudo vim /etc/fstab**

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options>          <dump> <pass>
# / was on /dev/sda4 during curtin installation
/dev/disk/by-uuid/ce51d912-6808-46e4-874b-d1660ba3d35c /          ext4    defaults 0 1
# /boot/efi was on /dev/sda1 during curtin installation
/dev/disk/by-uuid/3807-86E0 /boot/efi  vfat    defaults 0 1
/swap.img                none     swap    sw       0 0
#/dev/sda3 /media/DATOS ntfs defaults,noauto 0 0
```

Incluimos esta línea

**UUID=524F567904D39ECB /media/DATOS ntfs-3g defaults,uid=1001,gid=1001,umask=022 0 0**

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options>          <dump> <pass>
# / was on /dev/sda4 during curtin installation
/dev/disk/by-uuid/ce51d912-6808-46e4-874b-d1660ba3d35c /          ext4    defaults 0 1
# /boot/efi was on /dev/sda1 during curtin installation
/dev/disk/by-uuid/3807-86E0 /boot/efi  vfat    defaults 0 1
/swap.img                none     swap    sw       0 0
#/dev/sda3 /media/DATOS ntfs defaults,noauto 0 0
# permitir el montaje sin sudo (Gerar para RClone - Datos)
UUID=524F567904D39ECB /media/DATOS ntfs-3g defaults,uid=1001,gid=1001,umask=022 0 0
```



## APLICAR CAMBIOS DE SYSTEMD (MENSAJE DE ADVERTENCIA)

**sudo systemctl daemon-reexec**

ó

**sudo systemctl daemon-reload**

Esto asegura que systemd tenga la última versión del fstab.



## INTENTAR MONTAR



```
sudo mount -a
```

Esto monta todo lo que haya pendiente



## SERVICIOS SHUTDOWN DE APAGADO DE SISTEMA



### SERVICIO SHUTDOWN

El proceso es exactamente el mismo, pero cambia la ruta del servicio. El mayor cambio se produce a nivel de logs.

Como en los servicios de inicio podemos hacerlo de 2 maneras:

Opción 1	file-name.sh	/usr/bin/
	+	
	file-name.service	/lib/systemd/system-shutdown/
Opción 2	[file-name] o [file-name.sh]	/lib/systemd/system-shutdown/



### PROCESO



CREAR SCRIPT



DARLE PERMISOS DE EJECUCION



CREAR ARCHIVO DE SERVICIO SHUTDOWN



HABILITAR EL SERVICIO



### CAMBIOS A NIVEL DE LOGS

El mayor cambio se produce a nivel de logs.

Durante el apagado, muchos sistemas de archivos ya pueden estar desmontados cuando se ejecutan los scripts en /lib/systemd/system-shutdown.

Por tanto, escribir en /var/log/ no siempre es seguro ni efectivo en ese momento.

Usa un archivo de log en /run/ (que está en tmpfs, en memoria):

Ruta **/run/**

Tipo Guardarlo como **file-name.log**

```
Sudo touch /run/log-name.log
```

Asegúrate de que el archivo de log se pueda escribir





GITHUB LINK

GERARDO MTNZ-DEV



REPOSITORY LINK

LINUX PUBLIC REPO



G-Mail



LinkedIn



Linktree



G-Mtnz Web

```
chown user-name:group-name log-name.log
```



## EL SCRIPT

Script para desmontar una Unidad

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

Tipo Guardado sin extensión, como Script **file-name**

```
#!/bin/bash
#!/lib/systemd/system-shutdown/desmontar_datos
DATOS_MOUNT="/media/DATOS"
LOG_FILE="/run/desmontar_datos.log"
echo "[Shutdown] Intentando desmontar unidad D..." >> "$LOG_FILE"
desmontar_si() {
    MOUNT_PATH="$1"
    if mount | grep -q "$MOUNT_PATH"; then
        umount -l "$MOUNT_PATH" && \
            echo "Desmontado correctamente: $MOUNT_PATH" >> "$LOG_FILE" || \
            echo "Fallo al desmontar: $MOUNT_PATH" >> "$LOG_FILE"
    else
        echo "No estaba montado: $MOUNT_PATH" >> "$LOG_FILE"
    fi
}
desmontar_si "$DATOS_MOUNT"
exit 0
```



## MEJORAS DEL SCRIPT



### LOG SE GUARDE EN DISCO ANTES DEL APAGADO

Para que ese log se guarde en disco antes del apagado para conservarlo tras reiniciar, necesitas mover el archivo de log desde la RAM (/run/) a un disco persistente antes de que el sistema se apague del todo.

Lo más seguro es escribir el log en un archivo dentro de /root/ o / directamente (porque el sistema de archivos raíz permanece disponible hasta el final del apagado).

Añadimos al Script

```
#!/bin/bash
#!/lib/systemd/system-shutdown/desmontar_datos
DATOS_MOUNT="/media/DATOS"
PERSISTENT_LOG="/root/desmontar_datos.log"
echo "[Shutdown] $(date '+%Y-%m-%d %H:%M:%S') - Intentando desmontar unidad D..." >> "$PERSISTENT_LOG"
desmontar_si() {
    MOUNT_PATH="$1"
    if mount | grep -q "$MOUNT_PATH"; then
        umount -l "$MOUNT_PATH" && \
            echo "Desmontado correctamente: $MOUNT_PATH" >> "$LOG_FILE" || \
            echo "Fallo al desmontar: $MOUNT_PATH" >> "$LOG_FILE"
    else

```





GITHUB LINK

GERARDO MTNZ - DEV



REPOSITORY LINK

LINUX PUBLIC REPO



G-Mail



LinkedIn



Linktree



G-Mtnz Web

```

    echo "No estaba montado: $MOUNT_PATH" >> "$LOG_FILE"
fi
}
desmontar_si "$DATOS_MOUNT"
exit 0

```



## LIMITAR TAMANO ARCHIVO LOG

Si te preocupa el crecimiento del archivo, puedes limitarlo manualmente agregando al principio del script:

```

if [ -f "$PERSISTENT_LOG" ] && [ "$(wc -l < "$PERSISTENT_LOG")" -gt 500 ];
then
    tail -n 200 "$PERSISTENT_LOG" > "$PERSISTENT_LOG.tmp" && mv
"$PERSISTENT_LOG.tmp" "$PERSISTENT_LOG"
fi

```



## COPIAR LOG, P.EJ. A CARPETA USUARIO

Para que se copie el log a una ruta como /home/gerar\_kde/.logs/desmontar\_datos.log.

Creamos la carpeta .logs en la carpeta del usuario

Ruta **/home/gerar\_kde/.logs**

```
Sudo mkdir /home/gerar_kde/.logs
```

Asegúrate de que el archivo de log se pueda escribir

```
chown user-name:group-name /home/gerar_kde/.logs
```

Y añadimos al Script

```

#!/bin/bash
#/lib/systemd/system-shutdown/desmontar_datos
DATOS_MOUNT="/media/DATOS"
PERSISTENT_LOG="/root/desmontar_datos.log"
DEST_DIR="/home/gerar_kde/.logs"
DEST_LOG="$DEST_DIR/desmontar_datos.log"

# Limitar tamaño del log en /root
if [ -f "$PERSISTENT_LOG" ] && [ "$(wc -l < "$PERSISTENT_LOG")" -gt 500 ];
then
    tail -n 200 "$PERSISTENT_LOG" > "${PERSISTENT_LOG}.tmp" && mv "$
${PERSISTENT_LOG}.tmp" "$PERSISTENT_LOG"
fi

# Registro inicial
echo "[Shutdown] $(date '+%Y-%m-%d %H:%M:%S') - Intentando desmontar unidad D..." >> "$PERSISTENT_LOG"

# Función para copiar el log si existe el directorio destino
copiar_log_a_home() {
    if [ -d "$DEST_DIR" ]; then
        cp "$PERSISTENT_LOG" "$DEST_LOG"
        echo "Log copiado a $DEST_LOG" >> "$PERSISTENT_LOG"
    else

```



```

        echo "Directorio de destino no encontrado: $DEST_DIR" >>
"$PERSISTENT_LOG"
    fi
}

# Función de desmontaje
desmontar_si() {
    MOUNT_PATH="$1"
    if mount | grep -q "$MOUNT_PATH"; then
        copiar_log_a_home
        umount -l "$MOUNT_PATH" && \
            echo "Desmontado correctamente: $MOUNT_PATH" >> "$PERSISTENT_LOG" || \
            echo "Fallo al desmontar: $MOUNT_PATH" >> "$PERSISTENT_LOG"
    else
        echo "No estaba montado: $MOUNT_PATH" >> "$PERSISTENT_LOG"
    fi
}
desmontar_si "$DATOS_MOUNT"
exit 0

```

El log se escribe en /root/desmontar\_datos.log (como copia confiable). Limitando el tamaño tal que: Si el archivo existe y tiene más de 500 líneas. Conservar solo las últimas 200 líneas.

Luego, se asegura de que /home/gerar\_kde/.logs exista antes de copiar, si existe lo copia y si no existe, el mensaje correspondiente se agrega al log en /root/.



## RENOMBRAR POR FECHA LA COPIA DEL LOG

Para que también el log en /home/gerar\_kde/.logs/ se renombre por fecha cada vez, por ejemplo: desmontar\_datos\_2025-05-13.log

añadimos al Script

```

#!/bin/bash
# /lib/systemd/system-shutdown/desmontar_datos

DATOS_MOUNT="/media/DATOS"
PERSISTENT_LOG="/root/desmontar_datos.log"
DEST_DIR="/home/gerar_kde/.logs"
DATE_SUFFIX="$(date '+%Y-%m-%d')"
DEST_LOG="$DEST_DIR/desmontar_datos_${DATE_SUFFIX}.log"

# Limitar tamaño del log en /root
if [ -f "$PERSISTENT_LOG" ] && [ "$(wc -l < "$PERSISTENT_LOG")" -gt 500 ]; then
    tail -n 200 "$PERSISTENT_LOG" > "${PERSISTENT_LOG}.tmp" && mv "${PERSISTENT_LOG}.tmp" "$PERSISTENT_LOG"
fi

# Registro inicial
echo "[Shutdown] $(date '+%Y-%m-%d %H:%M:%S') - Intentando desmontar unidad D..." >> "$PERSISTENT_LOG"

# Función para copiar el log a /home/gerar_kde/.logs con nombre por fecha
copiar_log_a_home() {
    if [ -d "$DEST_DIR" ]; then
        cp "$PERSISTENT_LOG" "$DEST_LOG"
        echo "Log copiado a $DEST_LOG" >> "$PERSISTENT_LOG"
    else
        echo "Directorio de destino no encontrado: $DEST_DIR" >>
"$PERSISTENT_LOG"
    fi
}

# Función de desmontaje
desmontar_si() {
    MOUNT_PATH="$1"
    if mount | grep -q "$MOUNT_PATH"; then

```



GITHUB LINK

GERARDO MTNZ - DEV



REPOSITORY LINK

LINUX PUBLIC REPO



G-Mail



LinkedIn



Linktree



G-Mtnz Web

```

copiar_log_a_home
umount -l "$MOUNT_PATH" && \
    echo "Desmontado correctamente: $MOUNT_PATH" >> "$PERSISTENT_LOG" || \
    echo "Fallo al desmontar: $MOUNT_PATH" >> "$PERSISTENT_LOG"
else
    echo "No estaba montado: $MOUNT_PATH" >> "$PERSISTENT_LOG"
fi
}
desmontar_si "$DATOS_MOUNT"
exit 0

```

El log principal Permanece en /var/log/montar\_drive.log como archivo principal, limitado a cierto número de líneas.

Se copia a /home/gerar\_kde/.logs/ con un nombre por fecha, para mantener un historial diario.



## NUEVO SCRIPT

Script para montar Google Drive con rClone

Ruta **/usr/local/bin/**

Tipo **.sh montar\_drive.sh**

Log **/var/log/montar\_drive.log**

```

#!/bin/bash
# /usr/local/bin/montar_drive.sh

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

# Limitar tamaño del log principal

```

if [ -f "$LOG_FILE" ] && [ "$(wc -l < "$LOG_FILE")" -gt 500 ]; then
    tail -n 200 "$LOG_FILE" > "${LOG_FILE}.tmp" && mv "${LOG_FILE}.tmp"
"$LOG_FILE"
fi

```

# Redirigir salida estándar y de errores al log principal

```

exec > "$LOG_FILE" 2>&1

```

# Verifica que el punto de montaje exista

```

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

```

# Verifica si hay conectividad antes de montar

```

if ! ping -c 1 -W 2 www.googleapis.com &> /dev/null; then
    echo "No hay conexión a internet. Reintentará más tarde..."
    exit 2
fi

```

# Ejecuta rclone en modo foreground (sin --daemon, lo controla systemd)

```

exec /usr/bin/rclone mount \
    gdrive: "$MOUNT_POINT" \
    --vfs-cache-mode writes \
    --allow-other \
    --dir-cache-time 1h \
    --poll-interval 1m \
    --timeout 1m \

```



```
--umask 002 \
--log-level INFO \
--log-file "$LOG_FILE"
```



## MEJORAS DEL SCRIPT



### LIMITAR LOG PRINCIPAL Y COPIARLO CON UN NOMBRE POR FECHA

El log principal Permanece en /var/log/montar\_drive.log como archivo principal, limitado a cierto número de líneas.

Además Se copia a /home/gerar\_kde/.logs/ con un nombre por fecha, para mantener un historial diario.

Cambiamos esta parte que es la que manejaba la salida log

```
LOG_FILE="/var/log/montar_drive.log"
exec > "$LOG_FILE" 2>&1
```

Lo cambiamos y Ponemos este bloque al inicio del script, justo antes de que empiece a escribir logs o realizar acciones importantes.

```
LOG_FILE="/var/log/montar_drive.log"
DATE_SUFFIX="$(date '+%Y-%m-%d')"
DEST_DIR="/home/gerar_kde/.logs"
DEST_LOG="$DEST_DIR/montar_drive_${DATE_SUFFIX}.log"

# Limitar tamaño del log principal
if [ -f "$LOG_FILE" ] && [ "$(wc -l < "$LOG_FILE")" -gt 500 ]; then
    tail -n 200 "$LOG_FILE" > "${LOG_FILE}.tmp" && mv "${LOG_FILE}.tmp"
"$LOG_FILE"
fi

# Este exec reemplazará el script por rclone solo si llega hasta aquí
exec /usr/bin/rclone mount \
    gdrive: "$MOUNT_POINT" \
    --vfs-cache-mode writes \
    --allow-other \
    --dir-cache-time 1h \
    --poll-interval 1m \
    --timeout 1m \
    --umask 002 \
    --log-level INFO \
    --log-file "$LOG_FILE"

# Si exec falla, se ejecutará esta parte (lo cual no debería pasar normalmente)
# Copia el log al directorio del usuario (solo si no fue reemplazado por exec)
if [ -d "$DEST_DIR" ]; then
    cp "$LOG_FILE" "$DEST_LOG"
```



```

    echo "[Info] Log copiado a $DEST_LOG" >> "$LOG_FILE"
else
    echo "[Warning] No se pudo copiar el log a $DEST_DIR (no existe)" >>
"$LOG_FILE"
fi
exit 0

```

```

#!/bin/bash
#/usr/local/bin/montar_drive.sh

LOG_FILE="/var/log/montar_drive.log"
DATE_SUFFIX="$(date '+%Y-%m-%d')"
DEST_DIR="/home/gerar_kde/.logs"
DEST_LOG="$DEST_DIR/montar_drive_${DATE_SUFFIX}.log"

# Limitar tamaño del log principal
if [ -f "$LOG_FILE" ] && [ "$(wc -l < "$LOG_FILE")" -gt 500 ]; then
    tail -n 200 "$LOG_FILE" > "${LOG_FILE}.tmp" && mv "${LOG_FILE}.tmp"
"$LOG_FILE"
fi

# Registro inicial
echo "[${TIMESTAMP}] Inicio del montaje de Google Drive" >> "$LOG_FILE"

# Verifica si ya está montado
if mountpoint -q "$MOUNT_POINT"; then
    echo "Google Drive ya está montado en $MOUNT_POINT" >> "$LOG_FILE"
    exit 0
fi

# Verifica que el punto de montaje exista
if [ ! -d "$MOUNT_POINT" ]; then
    echo "Error: el directorio $MOUNT_POINT no existe." >> "$LOG_FILE"
    exit 1
fi

# Verifica si hay conectividad antes de montar
if ! ping -c 1 -W 2 www.googleapis.com &> /dev/null; then
    echo "No hay conexión a internet. Reintentará más tarde..." >> "$LOG_FILE"
    exit 2
fi

# Montar con rclone
echo "[${date '+%Y-%m-%d %H:%M:%S'}] Ejecutando rclone mount..." >>
"$LOG_FILE"

# Este exec reemplazará el script por rclone solo si llega hasta aquí
exec /usr/bin/rclone mount \
    gdrive: "$MOUNT_POINT" \
    --vfs-cache-mode writes \

```





```

--allow-other \
--dir-cache-time 1h \
--poll-interval 1m \
--timeout 1m \
--umask 002 \
--log-level INFO \
--log-file "$LOG_FILE"
# Si exec falla, se ejecutará esta parte (lo cual no debería pasar normalmente)
# Copia el log al directorio del usuario (solo si no fue reemplazado por exec)
if [ -d "$DEST_DIR" ]; then
    cp "$LOG_FILE" "$DEST_LOG"
    echo "[Info] Log copiado a $DEST_LOG" >> "$LOG_FILE"
else
    echo "[Warning] No se pudo copiar el log a $DEST_DIR (no existe)" >>
"$LOG_FILE"
fi
exit 0

```



## SERVICIOS AL APAGAR O REINICIAR USANDO SYSTEMD



### EJEMPLO CON SINCRONIZACIÓN FREE-FILE-SYNC

Perfecto. Para ejecutar tu script al **apagar** o **reiniciar** el sistema, lo más confiable es usar un **servicio systemd**. A continuación te muestro cómo hacerlo paso a paso.



### ASEGÚRATE Q SCRIPT FUNCIONA SOLO Y TENGA PERMISOS

Supongamos que tu script se llama:

```
/home/gerar_kde/scripts/sincronizar.sh
```

Verifica que sea ejecutable:

```
chmod +x /home/gerar_kde/scripts/sincronizar.sh
```



### CREA UN SERVICIO SYSTEMD

Abre una terminal y ejecuta:

```
sudo vim /etc/systemd/system/ffs-sinc-apagado.service
```

Pega esto dentro:

```
[Unit]
```





Description=Sincronización FreeFileSync antes de apagar o reiniciar

DefaultDependencies=no

Before=shutdown.target reboot.target halt.target

[Service]

Type=oneshot

ExecStart=/home/gerar\_kde/scripts/sincronizar.sh

RemainAfterExit=true

[Install]

WantedBy=halt.target reboot.target shutdown.target



## RECARGA SYSTEMD Y HABILITA EL SERVICIO

```
sudo systemctl daemon-reexec
```

```
sudo systemctl enable ffs-sinc-apagado.service
```



### 4. (OPCIONAL) VERIFICA SI ESTÁ HABILITADO

```
systemctl status ffs-sinc-apagado.service
```



### CONSEJOS ADICIONALES

- Si tu script necesita acceso gráfico (como interfaz de FreeFileSync), **no funcionará** bien al apagar, porque el entorno gráfico ya se está cerrando. Asegúrate de que no dependa de DISPLAY.
- Si quieres evitar problemas, asegúrate de que el `.ffs_batch` tenga marcada la opción:  

```
<ProgressDialog Minimized="true" AutoClose="true"/>
```
- Si falla, redirige la salida en el script a un log:  

```
/usr/bin/FreeFileSync /ruta/al/archivo.ffmpeg_batch >> /home/gerar_kde/log-ffmpeg.log 2>&1
```

¡Gracias por compartir esa información! Veo que estás usando correctamente el directorio:

```
perl
```

```
CopiarEditar
```

```
/usr/lib/systemd/system-shutdown/
```

Este es un mecanismo especial de systemd para ejecutar scripts **justo al apagar o reiniciar**, y **antes del desmontaje del sistema de archivos**. Es ideal para tu caso.



#### 1. Todos los scripts aquí deben:

- Ser **ejecutables**





- Tener **shebang** (`#!/bin/bash`) al principio
- Ser propietarios de root o ejecutables por root
- **No pueden mostrar GUI** (nada de ventanas)

## 2. systemd ejecuta estos scripts con esta sintaxis:

```
bash
CopiarEditar
/script.sh halt
/script.sh poweroff
/script.sh reboot
```

Es decir, se pasa un argumento con el tipo de apagado. Tu script **debe aceptar ese argumento**, aunque lo ignores.



Asegúrate de que tu script tenga este formato:

```
bash
CopiarEditar
#!/bin/bash

# Aceptar argumento del tipo de apagado (reboot, halt, poweroff)
MODE="$1"

# Ruta completa al ejecutable de FreeFileSync
/usr/bin/FreeFileSync /home/gerar_kde/scripts/Syncro_Iconos.ffs_batch >>
/var/log/ffs_iconos.log 2>&1
```



### 1. Dale permisos de ejecución por si acaso:

```
bash
CopiarEditar
sudo chmod +x /usr/lib/systemd/system-shutdown/Syncro_Iconos_D.sh
```

### 2. Prueba manualmente (simulando un reinicio):

```
bash
CopiarEditar
sudo /usr/lib/systemd/system-shutdown/Syncro_Iconos_D.sh reboot
```

### 3. Revisa el log generado:

```
bash
CopiarEditar
cat /var/log/ffs_iconos.log
```





- Si FreeFileSync no hace nada en ese momento, puede deberse a que el entorno gráfico ya no está disponible. Asegúrate de que el `.ffs_batch` no dependa de GUI.
- Usa FreeFileSync solo para sincronización silenciosa (minimizado y autocierre) en este contexto.
- Si falla, puedes usar `rsync` en su lugar como solución más robusta en entorno de apagado.

Tu script `Syncro_Iconos_D.sh` está **casi perfecto**, pero te haré unas observaciones y recomendaciones menores para asegurar que funcione **fiablemente al apagar o reiniciar**.

✓ Validaciones positivas de tu script:

- ✓ Tiene el shebang correcto: `#!/bin/bash`
- ✓ Acepta el argumento `$1` como requiere `systemd`
- ✓ Llama a FreeFileSync con ruta absoluta
- ✓ Redirige el log a un archivo (muy útil)
- ✓ El archivo parece estar en un directorio correcto: `/usr/lib/systemd/system-shutdown/`
- ✓ Tiene permisos de ejecución
- 
- 🛠 Sugerencias para mejorar confiabilidad:

1. **Verifica que el log directory exista**

Si el directorio `/home/gerar_kde/.logs/Syncro_Logs/` **no existe al momento del apagado**, el script podría fallar silenciosamente. Para asegurarte:

```
bash
CopiarEditar
mkdir -p /home/gerar_kde/.logs/Syncro_Logs/
```

O agrega al script:

```
bash
CopiarEditar
mkdir -p /home/gerar_kde/.logs/Syncro_Logs
```

2. **Captura posibles errores (opcional)**

Para facilitar el diagnóstico:

```
bash
```



```
CopiarEditar
echo "[$(date)] Ejecutando sincronización de perfiles ($MODE)" >>
/home/gerar_kde/.logs/Syncro_Logs/sync_perfiles.log
```

Quedando así:

```
bash
CopiarEditar
#!/bin/bash
MODE="$1"

mkdir -p /home/gerar_kde/.logs/Syncro_Logs
echo "[$(date)] Ejecutando sincronización de perfiles ($MODE)" >>
/home/gerar_kde/.logs/Syncro_Logs/sync_perfiles.log

/usr/local/bin/FreeFileSync
/home/gerar_kde/Laptop/Syncro/Perfiles_D.ffmpeg_batch >>
/home/gerar_kde/.logs/Syncro_Logs/sync_perfiles.log 2>&1
```

### 3. Verifica que FreeFileSync funcione sin entorno gráfico

Al apagarse el sistema, no habrá entorno gráfico. Para evitar problemas, asegúrate de que:

- El archivo .ffmpeg\_batch tiene esta línea en la sección <Batch>:

```
xml
CopiarEditar
<ProgressDialog Minimized="true" AutoClose="true"/>
```

- No incluye acciones de post-sincronización que abran ventanas.
- 
- Recomendación: probar sin apagar

Ejecuta este comando manual para probar que todo corre bien *sin depender del apagado real*:

```
bash
CopiarEditar
sudo /usr/lib/systemd/systemd-shutdown/Syncro_Iconos_D.sh reboot
```

Después revisa:

```
bash
CopiarEditar
cat /home/gerar_kde/.logs/Syncro_Logs/sync_perfiles.log
```



## OPERACIONES CON SERVICIOS



### INDICE



LISTAR SERVICIOS HABILITADOS



VER EL ESTADO DE UN SERVICIO



## LISTAR SERVICIOS HABILITADOS AL ARRANQUE

```
systemctl list-unit-files --type=service | grep enabled
```

```

gerar_kde@Lap-Linux:~/$ systemctl list-unit-files --type=service | grep enabled
accounts-daemon.service          enabled      enabled
alsa-utils.service              masked      enabled
anacron.service                 enabled      enabled
apparmor.service               enabled      enabled
apport.service                  enabled      enabled
avahi-daemon.service            enabled      enabled
bluetooth.service              enabled      enabled
brlty.service                   disabled     enabled
chrome-remote-desktop.service   masked      enabled
chrome-remote-desktop@.service  disabled     enabled
cloud-config.service            enabled      enabled
cloud-final.service             enabled      enabled
cloud-init-local.service        enabled      enabled
cloud-init.service              enabled      enabled
console-setup.service           enabled      enabled
cron.service                    enabled      enabled
cryptdisks-early.service        masked      enabled
cryptdisks.service              masked      enabled
cups-browsed.service            enabled      enabled
cups.service                    enabled      enabled
dmesg.service                   enabled      enabled
drkonqi-coredump-processor@.service disabled     enabled
e2scrub_reap.service            enabled      enabled
gdomap.service                  disabled     enabled
getty@.service                  enabled      enabled
gnome-remote-desktop.service    enabled      enabled
gpu-manager.service             enabled      enabled
grub-common.service             enabled      enabled
grub-initrd-fallback.service    enabled      enabled
hwclock.service                 masked      enabled
kerneloops.service             enabled      enabled
keyboard-setup.service          enabled      enabled
m_drive_mount.service           enabled      enabled
m_one_mount.service             enabled      enabled
ModemManager.service           enabled      enabled
mount_datos.service            enabled      enabled
netplan-ovs-cleanup.service     enabled      enabled
networkd-dispatcher.service     enabled      enabled
NetworkManager-dispatcher.service enabled      enabled
NetworkManager-wait-online.service enabled      enabled
NetworkManager.service         enabled      enabled
nftables.service               disabled     enabled
nmbd.service                    enabled      enabled
openvpn-client@.service         disabled     enabled
openvpn-server@.service         disabled     enabled
openvpn.service                enabled      enabled
openvpn@.service               disabled     enabled
power-profiles-daemon.service    enabled      enabled
rsync.service                   disabled     enabled
rsyslog.service                 enabled      enabled

```



rtkit-daemon.service	disabled	enabled
samba-ad-dc.service	enabled	enabled
saned.service	masked	enabled
saned@.service	indirect	enabled
sddm.service	disabled	enabled
secureboot-db.service	enabled	enabled
serial-getty@.service	disabled	enabled
setvtrgb.service	enabled	enabled
smartmontools.service	enabled	enabled
smbd.service	enabled	enabled
snap.canonical-livepatch.canonical-livepatchd.service	enabled	enabled
snap.mesa-2404.component-monitor.service	disabled	enabled
snapd.apparmor.service	enabled	enabled
snapd.autoimport.service	enabled	enabled
snapd.core-fixup.service	enabled	enabled
snapd.recovery-chooser-trigger.service	enabled	enabled
snapd.seeded.service	enabled	enabled
snapd.service	enabled	enabled
snapd.system-shutdown.service	enabled	enabled
speech-dispatcherd.service	disabled	enabled
spice-vdagentd.service	indirect	enabled
ssl-cert.service	enabled	enabled
sssd-autofs.service	indirect	enabled
sssd-nss.service	indirect	enabled
sssd-pac.service	indirect	enabled
sssd-pam.service	indirect	enabled
sssd-ssh.service	indirect	enabled
sssd-sudo.service	indirect	enabled
sssd.service	enabled	enabled
sudo.service	masked	enabled
switcheroo-control.service	enabled	enabled
sysstat.service	enabled	enabled
systemd-confext.service	disabled	enabled
systemd-fsck-root.service	enabled-runtime	enabled
systemd-network-generator.service	disabled	enabled
systemd-networkd-wait-online.service	disabled	enabled
systemd-networkd-wait-online@.service	disabled	enabled
systemd-networkd.service	disabled	enabled
systemd-oemd.service	enabled	enabled
systemd-pcrlock-file-system.service	disabled	enabled
systemd-pcrlock-firmware-code.service	disabled	enabled
systemd-pcrlock-firmware-config.service	disabled	enabled
systemd-pcrlock-machine-id.service	disabled	enabled
systemd-pcrlock-make-policy.service	disabled	enabled
systemd-pcrlock-secureboot-authority.service	disabled	enabled
systemd-pcrlock-secureboot-policy.service	disabled	enabled
systemd-pstore.service	enabled	enabled
systemd-remount-fs.service	enabled-runtime	enabled
systemd-resolved.service	enabled	enabled
systemd-sysextd.service	disabled	enabled
systemd-sysupdate-reboot.service	indirect	enabled
systemd-sysupdate.service	indirect	enabled
systemd-timesyncd.service	enabled	enabled
thermald.service	enabled	enabled
ua-reboot-cmds.service	enabled	enabled
ubuntu-advantage.service	enabled	enabled
udisks2.service	enabled	enabled
ufw.service	enabled	enabled
unattended-upgrades.service	enabled	enabled
upower.service	disabled	enabled
uuidd.service	indirect	enabled
vboxautostart.service.service	enabled	enabled
vboxballoonctrl.service.service	enabled	enabled
vboxdrv.service	enabled	enabled
vboxweb.service.service	enabled	enabled
wpa_supplicant-nl80211@.service	disabled	enabled
wpa_supplicant-wired@.service	disabled	enabled
wpa_supplicant.service	enabled	enabled
wpa_supplicant@.service	disabled	enabled





```
x11-common.service
gerar_kde@Lap-Linux:~/$
```

masked

enabled

## VER EL ESTADO DE UN SERVICIO:

```
systemctl status service-name.service
```

```
gerar_kde@Lap-Linux:~/Code$ systemctl status m_drive_mount.service
systemctl status m_one_mount.service
systemctl status mount_datos.service

• m_drive_mount.service - Montar GOOGLE DRIVE en el arranque
Loaded: loaded (/etc/systemd/system/m_drive_mount.service; enabled; preset: enabled)
Active: active (running) since Sun 2025-05-18 20:47:47 CEST; 29min ago
Main PID: 2407 (m_drive_mount.s)
Tasks: 14 (limit: 9126)
Memory: 36.7M (peak: 39.8M)
CPU: 247ms
CGroup: /system.slice/m_drive_mount.service
└─2407 /bin/bash /usr/local/bin/m_drive_mount.sh
   └─2412 rclone mount --vfs-cache-mode=full --allow-non-empty gdrive: /home/gerar_kd

may 18 20:47:47 Lap-Linux systemd[1]: Started m_drive_mount.service - Montar GOOGLE DRIVE en el a>

• m_one_mount.service - Montar MS ONE-DRIVE en el arranque
Loaded: loaded (/etc/systemd/system/m_one_mount.service; enabled; preset: enabled)
Active: active (running) since Sun 2025-05-18 20:47:47 CEST; 29min ago
Main PID: 2408 (m_one_mount.sh)
Tasks: 14 (limit: 9126)
Memory: 41.8M (peak: 45.3M)
CPU: 224ms
CGroup: /system.slice/m_one_mount.service
└─2408 /bin/bash /usr/local/bin/m_one_mount.sh
   └─2413 rclone mount --vfs-cache-mode=full --allow-non-empty onedrive: /home/gerar_kd

may 18 20:47:47 Lap-Linux systemd[1]: Started m_one_mount.service - Montar MS ONE-DRIVE en el a>

• mount_datos.service - Montar partición DATOS en el arranque
Loaded: loaded (/etc/systemd/system/mount_datos.service; enabled; preset: enabled)
Active: active (exited) since Sun 2025-05-18 20:47:40 CEST; 29min ago
Main PID: 1091 (code=exited, status=0/SUCCESS)
CPU: 65ms

may 18 20:47:39 Lap-Linux systemd[1]: Starting mount_datos.service - Montar partición DATOS en >
may 18 20:47:40 Lap-Linux systemd[1]: Finished mount_datos.service - Montar partición DATOS en >

gerar_kde@Lap-Linux:~/Code$
```

