

Es posible generar un script para descargar archivos de Onedrive y guardar en un disco usb sin que se descarguen a la pc donde se sincroniza los archivos

Sí, es posible. La idea es **evitar que los archivos pasen por la carpeta sincronizada del sistema** y descargarlos **directamente desde la nube hacia el disco USB** usando una herramienta que haga transferencia nube → destino.

La forma más estable y usada para esto es con **rclone**, que permite copiar archivos desde OneDrive directamente a cualquier unidad montada (USB, disco externo, NAS, etc.) sin almacenarlos previamente en el disco interno.

---

## Cómo funciona la solución

El flujo sería:

```
add codereplace  
OneDrive (nube)  
↓  
rclone  
↓  
Disco USB  
add codereplace
```

El archivo nunca se guarda en la carpeta local sincronizada ni en el disco principal.

---

## Opción recomendada: usar rclone (ultra estable)

### 1 Instalar rclone

En Windows:

```
winget install rclone
```

o descargar desde:

<https://rclone.org/downloads/>

---

### 2 Configurar acceso a OneDrive

Ejecutar:

```
rclone config
```

Pasos básicos:

1. n → nueva configuración
2. nombre: onedrive
3. storage: elegir Microsoft OneDrive
4. login en navegador
5. aceptar permisos

Esto solo se hace una vez.

---

### 3 Identificar el disco USB

Ejemplo:

- Windows: E:\
  - Linux: /media/usb
  - Mac: /Volumes/USB
- 

### 4 Script básico (descarga directa al USB)

Ejemplo Windows (.bat):

```
rclone copy onedrive:/CarpetarRemota E:\BackupOneDrive --progress --  
transfers=4 --checkers=8
```

Ejemplo Linux/Mac:

```
rclone copy onedrive:/CarpetarRemota /media/usb/BackupOneDrive --progress
```

---

### ✓ Ventajas de este método

- No usa espacio del disco interno
  - Descarga directa nube → USB
  - Reanuda descargas interrumpidas
  - Verifica integridad
  - Puede automatizarse con tareas programadas
  - Mucho más estable que scripts HTTP simples
-

## Versión avanzada (ideal para backups automáticos)

Solo copia archivos nuevos o modificados:

```
rclone sync onedrive:/CarpetaRemota E:\BackupOneDrive --progress
```

Esto convierte el USB en un espejo del OneDrive.

---

### Importante

- El USB debe estar conectado antes de ejecutar el script.
  - Si el USB se desconecta, rclone detiene la copia sin corromper archivos.
  - No depende del cliente oficial de sincronización.
- 

Si quieras, puedo hacerte una **versión ultra estable tipo servidor**, por ejemplo:

- que detecte automáticamente el USB
- que continúe después de apagones
- que genere log de errores
- que descargue solo archivos nuevos cada día

¿Lo quieres para Windows o Linux?

hacer versión ultra estable tipo servidor y que instale todos requerimientos necesarios y dependencias en python

Sí. Vamos a hacer una **versión tipo servidor**, pensada para:

- Descargar directamente nube → USB (sin usar disco interno)
- Reanudar automáticamente si se corta la conexión
- Detectar si el USB está conectado
- Registrar logs
- Ejecutarse automáticamente (cron / tarea programada)
- Instalar dependencias automáticamente
- Funcionar de forma estable durante meses

La arquitectura más robusta es:

OneDrive → rclone → USB

↑

```
script Python (controlador)
add codereplace
```

Python no descarga los archivos; **controla y supervisa rclone**, que es mucho más estable para transferencias grandes.

---



## 1. Estructura del sistema

El sistema tendrá:

```
onedrive_backup/
└── backup.py          ← script principal
    ├── config.json      ← configuración
    ├── install.py        ← instalador automático
    └── logs/
```

---



## 2. Instalador automático (install.py)

Este script:

- verifica Python
- descarga rclone si no existe
- crea carpetas necesarias
- instala dependencias

```
import os
import subprocess
import sys
import platform
import urllib.request
import zipfile

RCLONE_URL_WIN = "https://downloads.rclone.org/rclone-current-windows-amd64.zip"

def install_python_deps():
    subprocess.run([sys.executable, "-m", "pip", "install",
                   "psutil", "tqdm"], check=True)

def install_rclone():
    if os.path.exists("rclone.exe"):
        print("rclone ya instalado")
        return

    print("Descargando rclone...")
```

```
urllib.request.urlretrieve(RCLONE_URL_WIN, "rclone.zip")

with zipfile.ZipFile("rclone.zip", 'r') as zip_ref:
    zip_ref.extractall(".")

for root, dirs, files in os.walk("."):
    if "rclone.exe" in files:
        os.rename(os.path.join(root, "rclone.exe"), "rclone.exe")

print("rclone instalado")

def create_folders():
    os.makedirs("logs", exist_ok=True)

if __name__ == "__main__":
    install_python_deps()
    install_rclone()
    create_folders()
    print("Instalación completa")
```

---

## 3. Archivo de configuración (config.json)

```
{  
    "remote": "onedrive:/Backup",  
    "usb_path": "E:/OneDriveBackup",  
    "check_interval": 60,  
    "transfers": 4,  
    "checkers": 8  
}
```

---

## 4. Script ULTRA ESTABLE (backup.py)

Características:

- espera hasta que el USB esté conectado
- reintenta indefinidamente
- crea logs
- reinicia si falla
- sincroniza solo cambios

```
import subprocess  
import json  
import time
```

```

import os
from datetime import datetime

CONFIG_FILE = "config.json"
LOG_FILE = "logs/backup.log"

def log(msg):
    line = f"{datetime.now()} - {msg}"
    print(line)
    with open(LOG_FILE, "a", encoding="utf-8") as f:
        f.write(line + "\n")

def load_config():
    with open(CONFIG_FILE) as f:
        return json.load(f)

def usb_connected(path):
    return os.path.exists(path)

def run_rclone(cfg):
    cmd = [
        "rclone",
        "sync",
        cfg["remote"],
        cfg["usb_path"],
        "--progress",
        "--transfers", str(cfg["transfers"]),
        "--checkers", str(cfg["checkers"]),
        "--retries", "999",
        "--low-level-retries", "999",
        "--log-file", LOG_FILE,
        "--log-level", "INFO"
    ]
    log("Iniciando sincronización")
    return subprocess.run(cmd).returncode

def main():
    cfg = load_config()

    while True:
        if not usb_connected(cfg["usb_path"]):
            log("USB no conectado, esperando...")
            time.sleep(cfg["check_interval"])
            continue

        code = run_rclone(cfg)

        if code == 0:
            log("Sincronización completa")
        else:
            log("Error detectado, reintentando en 60s")

        time.sleep(cfg["check_interval"])

if __name__ == "__main__":
    main()

```

---

## 5. Primera configuración (solo una vez)

Ejecutar:

```
rclone config
```

y conectar tu cuenta.

---

## 6. Modo servidor (recomendado)

### Windows (automático al iniciar)

Crear tarea programada:

```
python backup.py
```

Al iniciar el sistema.

---

### Linux (daemon)

```
crontab -e  
@reboot /usr/bin/python3 /ruta/backup.py
```

---

## Ventajas de esta versión

- No usa carpeta sincronizada
  - No usa disco interno
  - Recupera descargas interrumpidas
  - Funciona aunque el USB se desconecte
  - Logs completos
  - Sin límite de tamaño
  - Estable para bibliotecas grandes
- 

Siquieres en el siguiente paso puedo hacer la **versión profesional nivel NAS**, que incluye:

- detección automática del USB por nombre (no letra)
- límite de velocidad para no saturar internet
- múltiples cuentas OneDrive
- verificación hash completa
- modo solo descarga nocturna
- panel web local para ver progreso

Esa ya es versión “servidor permanente”.

detallar los pasos para instalar rclone en windows

A continuación tienes los **pasos completos y seguros** para instalar **rclone en Windows**, pensado para usarlo luego como sistema estable de descarga directa desde OneDrive hacia un disco USB sin pasar por la carpeta sincronizada del PC.

---

## **MÉTODO RECOMENDADO (Manual — más estable para uso tipo servidor)**

Este método evita problemas de permisos o actualizaciones automáticas.

---

### ◆ **Paso 1 — Descargar rclone**

1. Ir al sitio oficial:  
👉 <https://rclone.org/downloads/>
2. Descargar:

`rclone-current-windows-amd64.zip`  
add codereplace

(para Windows 64 bits, que es lo normal hoy).

---

### ◆ **Paso 2 — Extraer archivos**

1. Clic derecho al archivo ZIP.
2. Elegir **Extraer todo**.
3. Obtendrás una carpeta similar a:

```
rclone-v1.xx.x-windows-amd64
```

Dentro está:

```
rclone.exe
```

---

## ◆ Paso 3 — Mover rclone a una carpeta fija

Para uso estable tipo servidor:

1. Crear carpeta:

```
C:\rclone
```

2. Copiar dentro:

```
rclone.exe
```

Debe quedar:

```
C:\rclone\rclone.exe
```

---

## ◆ Paso 4 — Agregar rclone al PATH (MUY IMPORTANTE)

Esto permite ejecutar `rclone` desde cualquier ventana.

### 1 Abrir variables de entorno

- Presiona Win + R
- Escribe:

```
sysdm.cpl
```

- Pestaña **Opciones avanzadas**
  - Botón **Variables de entorno**
- 

### 2 Editar PATH

En **Variables del sistema**:

1. Buscar Path
2. Editar
3. Nuevo
4. Agregar:

C:\rclone

Aceptar todo.

---

## ◆ **Paso 5 — Verificar instalación**

Abrir **CMD o PowerShell** y escribir:

```
rclone version
```

Debe aparecer algo como:

```
rclone v1.xx.x
```

Si aparece, ya está instalado correctamente.

---

## ◆ **Paso 6 — Configurar acceso a OneDrive (solo una vez)**

Ejecutar:

```
rclone config
```

Luego:

1. n → nueva configuración
  2. nombre: onedrive
  3. storage → elegir Microsoft OneDrive
  4. Enter en opciones por defecto
  5. Se abre navegador
  6. Iniciar sesión
  7. Aceptar permisos
  8. y para guardar
-

## ◆ Paso 7 — Probar conexión

```
rclone lsd onedrive:
```

Debe listar tus carpetas de OneDrive.

---

## ✓ Resultado final

Ahora puedes copiar directamente nube → USB:

```
rclone copy onedrive:/Carpeta E:\Backup
```

Sin usar el disco interno.

---

## ⚠ Recomendación importante (modo servidor)

Crear carpeta:

```
C:\rclone\logs
```

y siempre ejecutar con:

```
--log-file C:\rclone\logs\backup.log
```

Esto permite detectar errores después.

---

Siquieres, en el siguiente paso puedo mostrarte cómo dejar **rclone funcionando como servicio de Windows**, que:

- ✓ inicia solo al encender la PC
- ✓ reinicia si falla
- ✓ continúa descargas automáticamente
- ✓ funciona sin abrir ventanas

(es como un mini servidor de backup).

rclone funcionando como servicio de Windows

A continuación tienes el procedimiento **correcto y estable** para ejecutar **rclone como servicio de Windows**, de modo que funcione como un **servidor de sincronización automática**, sin ventanas abiertas y reiniciándose solo si falla.

Este método es el que se usa en servidores y NAS Windows.

---

## **Objetivo**

Que el sistema funcione así:

```
Windows inicia
  ↓
Servicio rclone arranca automáticamente
  ↓
Sincroniza OneDrive → USB
  ↓
Si falla, se reinicia solo
add codereplace
```

Sin intervención manual.

---

## **Método profesional (usando NSSM)**

Windows no permite ejecutar fácilmente programas normales como servicios, por eso se usa:

 **NSSM** (Non-Sucking Service Manager)

Es la forma más estable.

---

### ◆ **Paso 1 — Descargar NSSM**

1. Ir a:  
<https://nssm.cc/download>
2. Descargar la versión **win64**
3. Extraer el ZIP.

Dentro encontrarás:

win64\nssm.exe

---

## ◆ Paso 2 — Crear estructura recomendada

Ejemplo:

```
C:\rclone\  
└── rclone.exe  
└── backup.bat  
└── logs\
```

---

## ◆ Paso 3 — Crear script de ejecución

Crear archivo:

```
C:\rclone\backup.bat
```

Contenido:

```
@echo off  
  
rclone sync onedrive:/Backup E:\OneDriveBackup ^  
--transfers 4 ^  
--checkers 8 ^  
--retries 999 ^  
--low-level-retries 999 ^  
--log-file C:\rclone\logs\backup.log ^  
--log-level INFO ^  
--stats 30s  
  
timeout /t 10
```

Este script:

- reintenta automáticamente
  - escribe logs
  - no se detiene por errores temporales
- 

## ◆ Paso 4 — Instalar el servicio

Abrir CMD como Administrador.

**Ir a la carpeta NSSM:**

```
cd C:\nssm\win64
```

**Ejecutar:**

```
nssm install RcloneBackup
```

Se abre ventana gráfica.

---

**Configurar:**

**Application tab**

**Path:**

C:\Windows\System32\cmd.exe

**Arguments:**

/c C:\rclone\backup.bat

**Startup directory:**

C:\rclone

---

**Details tab**

**Service name:**

RcloneBackup

**Startup type:**

Automatic

---

**I/O tab (recomendado)**

**Output:**

C:\rclone\logs\service.log

**Error:**

```
C:\rclone\logs\service-error.log
```

---

Presionar **Install service.**

---

## ◆ Paso 5 — Iniciar servicio

En CMD administrador:

```
net start RcloneBackup
```

o desde:

```
services.msc
```

Buscar:

```
RcloneBackup
```

---

## ✓ Verificación

El servicio debe:

- iniciar solo al arrancar Windows
- continuar aunque cierres sesión
- reiniciarse si falla
- escribir logs en:

```
C:\rclone\logs\
```

---

## ✓ Configuración ULTRA ESTABLE recomendada

Agregar al comando rclone:

```
--drive-chunk-size 64M  
--timeout 1h  
--contimeout 60s  
--expect-continue-timeout 10s
```

Esto evita cortes con archivos grandes.

---

## Recomendación importante para USB

Si el USB cambia de letra (muy común):

- asignar letra fija desde:
- Administración de discos

add codereplace

- por ejemplo siempre E::.
- 

Si quieras, en el siguiente paso puedo dejarte la **versión nivel servidor profesional**, usada en backups empresariales:

- detecta automáticamente cuando conectas el USB
- pausa si el USB se desconecta
- solo sincroniza en horarios definidos (ej. noche)
- envía aviso si falla
- múltiples cuentas OneDrive simultáneas

Eso ya queda funcionando 24/7 sin mantenimiento.

hice la prueba con el primer script backup.py y me dio esto: 2026-02-15 10:24:39.815860 -  
USB no conectado, esperando...

Ese mensaje significa simplemente que el script **no encuentra la ruta del USB definida en config.json**, entonces entra en modo espera (lo cual es correcto y es parte del diseño estable).

El sistema está funcionando bien   
Ahora vamos a corregir la detección.

---



## Por qué aparece “USB no conectado”

En tu config.json tienes algo como:

```
"usb_path": "E:/OneDriveBackup"  
add codereplace
```

El script verifica:

```
os.path.exists(path)
```

Si **esa carpeta exacta no existe**, interpreta que el USB no está conectado.

Puede fallar por:

1. ✗ El USB tiene otra letra (F:, G:, etc.)
  2. ✗ No existe la carpeta OneDriveBackup
  3. ✗ La barra está mal escrita (\ vs /)
  4. ✗ Windows no asignó letra fija
- 

## Solución rápida (probar en 1 minuto)

### 1 Verifica letra real del USB

Abrir CMD y escribir:

```
wmic logicaldisk get name
```

Ver qué letra tiene el USB.

---

### 2 Crear la carpeta manualmente

Si el USB es E:

Ir a:

```
E:\
```

Crear carpeta:

```
OneDriveBackup
```

Debe quedar:

```
E:\OneDriveBackup
```

---

### 3 Ajustar config.json correctamente

En Windows es mejor usar doble barra invertida:

```
{  
    "remote": "onedrive:/Backup",  
    "usb_path": "E:\\OneDriveBackup",  
    "check_interval": 60,  
    "transfers": 4,  
    "checkers": 8  
}
```

Guardar.

---

## 4 Prueba manual

En CMD:

```
rclone lsd onedrive:
```

Si lista carpetas, conexión OK.

Luego prueba:

```
rclone sync onedrive:/Backup E:\\OneDriveBackup --progress
```

Si esto funciona, el problema era solo la ruta.

---

## 🔧 Versión mejorada del detector USB (más inteligente)

Si quieres que detecte solo la letra del USB (sin depender de carpeta), cambia esta función en `backup.py`:

```
def usb_connected(path):  
    drive = os.path.splitdrive(path)[0]  
    return os.path.exists(drive + "\\\\")
```

Esto verifica solo que exista `E:\\` aunque no exista la subcarpeta.

---



## Recomendación profesional

Asignar letra fija al USB:

1. Win + X
2. Administración de discos
3. Clic derecho en USB
4. Cambiar letra y rutas
5. Asignar por ejemplo E:

Así nunca cambia.

---

Siquieres, puedo ahora:

- convertir el script en versión que detecta automáticamente cualquier USB por nombre
- hacer que cree la carpeta automáticamente si no existe
- agregar mensaje claro de qué letra está detectando
- hacer versión más robusta que no dependa de letra

Dime cuál prefieres y lo dejamos nivel servidor real.

si

Perfecto. Vamos a convertir tu **backup.py** en una **versión realmente tipo servidor**, mucho más robusta que la anterior.

Esta versión hará automáticamente:

- Detectar el USB por **nombre del volumen** (no por letra)
- Funcionar aunque Windows cambie la letra del USB
- Crear la carpeta destino si no existe
- Mostrar qué USB detectó
- Esperar si el USB se desconecta
- Reintentar indefinidamente
- Mantener logs claros

Esto elimina el problema más común en Windows: el cambio de letras de unidad.

---

## PASO 1 — Poner nombre fijo al USB (MUY IMPORTANTE)

1. Conectar el USB
2. Abrir **Este equipo**
3. Clic derecho sobre el USB
4. **Cambiar nombre**
5. Ejemplo:

```
BACKUP_USB  
add codereplace
```

(usar sin espacios para evitar problemas)

---

## PASO 2 — Nuevo config.json

Ahora NO usamos letra:

```
{  
    "remote": "onedrive:/Backup",  
    "usb_label": "BACKUP_USB",  
    "usb_folder": "OneDriveBackup",  
    "check_interval": 60,  
    "transfers": 4,  
    "checkers": 8  
}
```

---

## PASO 3 — backup.py versión ULTRA ESTABLE

Reemplaza completamente el archivo:

```
import subprocess  
import json  
import time  
import os  
import psutil  
from datetime import datetime  
  
CONFIG_FILE = "config.json"  
LOG_FILE = "logs/backup.log"
```

```

def log(msg):
    line = f"{datetime.now()} - {msg}"
    print(line)
    with open(LOG_FILE, "a", encoding="utf-8") as f:
        f.write(line + "\n")

def load_config():
    with open(CONFIG_FILE) as f:
        return json.load(f)

def find_usb_by_label(label):
    for p in psutil.disk_partitions():
        if label.lower() in p.mountpoint.lower():
            return p.mountpoint
    return None

def run_rclone(source, dest, cfg):
    cmd = [
        "rclone",
        "sync",
        source,
        dest,
        "--transfers", str(cfg["transfers"]),
        "--checkers", str(cfg["checkers"]),
        "--retries", "999",
        "--low-level-retries", "999",
        "--log-file", LOG_FILE,
        "--log-level", "INFO",
        "--stats", "30s"
    ]
    log(f"Iniciando sincronización hacia {dest}")
    return subprocess.run(cmd).returncode

def main():
    cfg = load_config()

    while True:
        usb_path = find_usb_by_label(cfg["usb_label"])

        if not usb_path:
            log("USB no detectado, esperando...")
            time.sleep(cfg["check_interval"])
            continue

        destination = os.path.join(
            usb_path,
            cfg["usb_folder"]
        )

        os.makedirs(destination, exist_ok=True)

        log(f"USB detectado en {usb_path}")

        code = run_rclone(cfg["remote"], destination, cfg)

```

```
if code == 0:  
    log("Sincronización completa")  
else:  
    log("Error detectado, reintentando")  
  
time.sleep(cfg["check_interval"])  
  
if __name__ == "__main__":  
    main()
```

---

## Qué cambia ahora

Antes:

USB debe ser E:

Ahora:

Cualquier letra → detecta BACKUP\_USB automáticamente

Ejemplo:

F:\OneDriveBackup  
G:\OneDriveBackup

Funciona igual.

---

## Prueba esperada en consola

Cuando conectes el USB verás:

USB detectado en F:\  
Iniciando sincronización...

Si lo desconectas:

USB no detectado, esperando...

---

## Siguiente mejora (nivel servidor real)

En el siguiente paso podemos añadir:

- pausa automática si se desconecta el USB durante copia
- detección instantánea (sin esperar 60s)
- límite de velocidad internet
- solo sincronizar de noche
- múltiples cuentas OneDrive
- modo NAS 24/7 sin supervisión

Esto ya queda como sistema profesional de backup continuo.

¿Quieres que lo dejemos en ese nivel?

sigue mostrando: USB no detectado, esperando...

Eso ocurre porque en Windows **el método anterior no puede leer correctamente el nombre (label) del USB.**

`psutil.disk_partitions()` devuelve la letra (E:\, F:\) pero **no el nombre del volumen**, por eso nunca coincide y el script cree que el USB no está conectado.

Esto es normal en Windows y lo vamos a corregir de forma definitiva.

---



## Problema real

La función actual:

```
if label.lower() in p.mountpoint.lower():
    add codereplace
```

no funciona porque:

```
mountpoint = E:\
label real = BACKUP_USB
```

Windows no incluye el label en esa información.

---

## Solución correcta (modo servidor real en Windows)

Vamos a leer el **nombre real del volumen** usando la API de Windows.

Esto es exactamente lo que hacen los programas profesionales de backup.

---

## ◆ PASO 1 — Instalar dependencia necesaria

En CMD:

```
pip install pywin32
```

---

## ◆ PASO 2 — Reemplazar función de detección USB

En `backup.py`, elimina esta función:

```
def find_usb_by_label(label):
```

y reemplázala por esta versión:

```
import win32api

def find_usb_by_label(label):
    drives = win32api.GetLogicalDriveStrings()
    drives = drives.split('\000')[:-1]

    for drive in drives:
        try:
            volume_name = win32api.GetVolumeInformation(drive)[0]
            if volume_name.lower() == label.lower():
                return drive
        except:
            pass

    return None
```

---

## ◆ PASO 3 — Verificar nombre EXACTO del USB

Muy importante:

1. Abrir **Este equipo**
2. Ver el nombre exacto del USB

Ejemplo:

BACKUP\_USB

Si aparece:

```
BACKUP_USB (E:)
```

el nombre correcto es solo:

```
BACKUP_USB
```

Debe coincidir exactamente con:

```
"usb_label": "BACKUP_USB"
```

---

## ◆ PASO 4 — Prueba rápida

Antes de ejecutar el script completo, prueba esto:

```
import win32api  
  
print(win32api.GetVolumeInformation("E:\\\\") [0])
```

Debe imprimir el nombre del USB.

---



## Resultado esperado ahora

Cuando ejecutes el script:

```
USB detectado en E:\\\nIniciando sincronización...
```

Si lo desconectas:

```
USB no detectado, esperando...
```

---

## ⚠ Nota importante (para estabilidad máxima)

Windows a veces tarda 3–5 segundos en montar el USB después de conectarlo.  
El script lo detectará en el siguiente ciclo automáticamente.

---

Siquieres, en el siguiente paso te dejo la **versión nivel servidor definitivo**, que ya usan setups 24/7:

- detecta conexión/desconexión instantánea (sin esperar 60s)
- pausa rclone si el USB se desconecta en mitad de copia
- continúa exactamente donde quedó
- protege contra corrupción si se retira el USB

Esto ya queda más robusto que el cliente oficial de sincronización de OneDrive.

ahora si lo reconoce, pero no hace nada: 2026-02-15 10:42:06.707384 - USB detectado en E:\ 2026-02-15 10:42:06.707384 - Iniciando sincronización hacia E:\OneDriveBackup

Perfecto 👍 ahora el USB ya se detecta correctamente.

Si muestra:

```
USB detectado en E:\  
Iniciando sincronización hacia E:\OneDriveBackup  
add codereplace
```

y luego **no parece hacer nada**, hay 4 causas posibles. Vamos directo al diagnóstico correcto.

---

## 1 Lo más probable: no hay cambios que sincronizar

Estás usando:

```
rclone sync
```

Si el destino ya está igual que OneDrive, entonces:

- 👉 rclone termina casi inmediatamente
  - 👉 no muestra progreso
  - 👉 parece que “no hace nada”
- 

### Verificar manualmente

En CMD prueba:

```
rclone sync onedrive:/Backup E:\OneDriveBackup --progress -vv
```

Si ves:

```
There was nothing to transfer
```

Entonces todo está sincronizado y el script funciona perfecto.

---

## 2 Puede estar esperando autenticación

Prueba:

```
rclone lsd onedrive:
```

Si da error, tu configuración no quedó guardada correctamente.

---

## 3 Puede estar trabajando pero no ves salida

Actualmente el script manda logs a archivo:

```
logs/backup.log
```

Abre ese archivo y revisa si está escribiendo actividad.

---

## 4 Puede estar bloqueado por PATH en subprocess

En Python usamos:

```
"rclone",
```

Pero si el servicio no encuentra rclone en PATH, el comando no ejecuta nada visible.

---

## SOLUCIÓN DEFINITIVA (hacerlo totalmente visible)

Modifica la función `run_rclone` así:

```
def run_rclone(source, dest, cfg):
    cmd = [
        "C:\\\\rclone\\\\rclone.exe",
        "sync",
        source,
        dest,
        "--transfers", str(cfg["transfers"]),
        "--checkers", str(cfg["checkers"]),
        "--retries", "999",
        "--low-level-retries", "999",
        "--stats", "10s",
        "-vv"
    ]
    log(f"Iniciando sincronización hacia {dest}")
    process = subprocess.Popen(cmd)
    process.wait()
    return process.returncode
```

 Usa la ruta completa de rclone.exe.

---

## Para probar que realmente transfiera algo

Haz una prueba real:

1. Sube un archivo nuevo pequeño a OneDrive
  2. Ejecuta el script
  3. Debe comenzar a descargarlo
-

## Cómo saber si está funcionando realmente

Mientras corre, abre CMD y ejecuta:

```
tasklist | findstr rclone
```

Si aparece `rclone.exe`, está trabajando.

---

## Si quieres confirmación visual inmediata

Cambia temporalmente el comando a:

```
"copy" en vez de "sync"
```

Para forzar transferencia aunque ya exista.

---



## Resumen

Si no hace nada, casi siempre es:

- No hay cambios
- Ya está sincronizado
- Rclone termina correctamente en segundos

No es error.

---

Siquieres, ahora podemos:

- agregar contador visual en consola
- mostrar porcentaje real en tiempo real
- forzar comparación por hash (ultra seguro)

- agregar modo “solo descargar nuevos archivos”
- convertirlo en servicio definitivo 24/7

Dime cuál quieres activar.

2026-02-15 10:45:44.621132 - Error detectado, reintentando

Perfecto, ahora ya tenemos información importante 

El mensaje:

```
Error detectado, reintentando
add codereplace
```

significa que **rclone devolvió un código distinto de 0**, o sea, ocurrió un error real durante la ejecución.

Pero ahora necesitamos ver **cuál es el error real**, porque el script solo muestra el resumen.

---



## PASO 1 — Ver el error real

Abre:

logs/backup.log

Busca las últimas líneas.

Ahí debe aparecer algo como:

- Failed to create file
- directory not found
- authentication error
- Access denied
- didn't find section in config file
- etc.

 Copia aquí las últimas 10 líneas del log si quieres que lo analicemos exacto.

---

# Mientras tanto, las 4 causas más comunes

---

## 1 Rclone no encuentra la configuración

Prueba en CMD:

```
rclone lsd onedrive:
```

Si aparece error tipo:

```
didn't find section in config file
```

Significa que el servicio no está usando el mismo perfil de usuario donde configuraste rclone.

### Solución rápida

Ejecuta:

```
rclone config file
```

Te mostrará algo como:

```
C:\Users\Alejandro\AppData\Roaming\rclone\rclone.conf
```

Luego modifica el script y agrega esta bandera en el comando:

```
--config="C:\\\\Users\\\\Alejandro\\\\AppData\\\\Roaming\\\\rclone\\\\rclone.conf",
```

Eso lo vuelve 100% estable.

---

## 2 Carpeta remota no existe

En config tienes:

```
"remote": "onedrive:/Backup"
```

Pero tal vez esa carpeta no existe.

Prueba:

```
rclone lsd onedrive:
```

Y verifica el nombre exacto.

Si tu carpeta se llama por ejemplo:

```
Respaldo
```

Entonces debe ser:

```
"remote": "onedrive:/Respaldo"
```

---

### 3 Permiso denegado en USB

Si el log dice:

```
Access is denied
```

Entonces ejecuta el script como Administrador.

---

### 4 El servicio NSSM está usando otro usuario

Si lo instalaste como servicio, puede estar corriendo como:

```
Local System
```

Ese usuario no tiene acceso a tu configuración rclone.

#### Solución profesional

En:

```
services.msc
```

Buscar:

```
RcloneBackup
```

Propiedades → pestaña **Log On**

Seleccionar:

This account

Y poner tu usuario de Windows.

---

## Paso inmediato

Ejecuta manualmente en CMD:

```
C:\rclone\rclone.exe sync onedrive:/Backup E:\OneDriveBackup -vv
```

Eso nos dirá el error real.

---

En este punto ya estamos en fase de ajuste fino.  
El sistema está casi listo para modo servidor definitivo.

Pásame el error exacto del log y lo dejamos funcionando perfecto.

```
2026/02/15 10:48:27 ERROR : Local file system at //?:E:/OneDriveBackup: not deleting files as there were IO errors 2026/02/15 10:48:27 ERROR : Local file system at //?:E:/OneDriveBackup: not deleting directories as there were IO errors 2026/02/15 10:48:27 ERROR : Attempt 500/999 failed with 1 errors and: directory not found 2026/02/15 10:48:28 ERROR : OneDrive root 'Backup': error reading source root directory: directory not found 2026/02/15 10:48:28 ERROR : Local file system at //?:E:/OneDriveBackup: not deleting files as there were IO errors
```

Perfecto 👍 ahora sí tenemos el diagnóstico real.

El error clave es este:

```
ERROR : OneDrive root 'Backup': error reading source root directory: directory not found  
add codereplace
```

Eso significa claramente:

 La carpeta **Backup** NO existe en tu **OneDrive**

No es problema del USB.  
No es problema del servicio.  
No es problema de permisos.

Es simplemente que esta ruta es incorrecta:

```
"remote": "onedrive:/Backup"
```

---

## Por qué ocurre

Cuando configuras un remote llamado:

```
onedrive
```

La raíz es tu OneDrive principal.

Si ejecutas:

```
rclone lsd onedrive:
```

Verás algo como:

```
-1 2026-02-10  Documents
-1 2026-02-11  Fotos
-1 2026-02-14  Trabajo
```

Si no aparece una carpeta llamada **Backup**, entonces esa ruta no existe.

---

## SOLUCIÓN 1 (más simple)

Si quieres sincronizar TODO tu OneDrive:

Cambia config.json a:

```
{
  "remote": "onedrive:",
  "usb_label": "BACKUP_USB",
  "usb_folder": "OneDriveBackup",
  "check_interval": 60,
  "transfers": 4,
  "checkers": 8
}
```

Observa que ahora es:

```
"onedrive": "
```

SIN /Backup

---

## SOLUCIÓN 2 (si quieres una carpeta específica)

Primero ejecuta:

```
rclone lsd onedrive:
```

Copia exactamente el nombre de la carpeta que sí existe.

Ejemplo si aparece:

```
Trabajo
```

Entonces usar:

```
"remote": "onedrive:/Trabajo"
```

Debe coincidir EXACTAMENTE mayúsculas/minúsculas.

---

## Sobre este error adicional

```
Local file system at //?:E:/OneDriveBackup: not deleting files as there  
were IO errors
```

Esto aparece porque estás usando:

```
rclone sync
```

y cuando el origen no existe, rclone intenta borrar el destino (porque sync = espejo).

Pero al fallar el origen, se detiene para no borrar nada por seguridad.

Eso es normal y en realidad es protección anti-catástrofe 

---

## Qué hacer ahora

**1** Ejecuta:

```
rclone lsd onedrive:
```

**2** Decide si quieres:

- sincronizar todo (`onedrive:`)
- o una carpeta específica

**3** Corrige `config.json`

**4** Reinicia el script

---

## Consejo profesional

Si estás en fase de pruebas, usa temporalmente:

```
"sync" → "copy"
```

Así nunca borrará archivos en el USB mientras pruebas.

---

Estamos ya en la etapa final.

Cuando ajustes esa ruta, debería comenzar a transferir inmediatamente.

Si quieras, después podemos:

- activar verificación por hash
- limitar ancho de banda
- hacer solo descarga (sin borrar)
- convertirlo en backup incremental profesional

Confírmame qué te muestra `rclone lsd onedrive:` y lo dejamos cerrado.