Analizador de Perfiles Wi-Fi (CLI + GUI)

Bienvenido al repo de un pequeño utilitario que lista perfiles Wi-Fi guardados, extrae sus contraseñas (cuando es posible) y genera un reporte bonito en texto. Además, puede enviar el informe por correo y trae una GUI sencilla en modo oscuro.

> USO AUTORIZADO: Proyecto con fines educativos. Úsalo solo en equipos propios o con permiso explícito.

¿Qué hace?

- **Escanea perfiles Wi-Fi** guardados en el sistema y **obtiene contraseñas**.
- **Genera un reporte** en texto con tabla alineada.
- **Envía el reporte por correo** (SMTP).
- **Extras opcionales** (según SO):
  - Windows: `netsh` (interfaces, drivers, export, etc.)
  - Linux: `nmcli/ip` (estado de interfaces, drivers/mac, ip address…)
- **GUI en Tkinter** en tema oscuro.

Requisitos

- Python 3.12+ (probado)
- Windows o Linux
- Para Linux:
  - NetworkManager (`nmcli`) para ver conexiones Wi-Fi y PSK
  - Paquete `python3-tk` para la GUI: `sudo apt install -y python3-tk`
- Para Windows:
  - `netsh` (nativo) y Python con `tkinter` (instalador oficial de Python lo incluye)

Configuración rápida con `.env` (recomendado)

Para evitar teclear credenciales cada vez, crea un archivo `.env` en la raíz del proyecto con:

```
SMTP_SERVER="smtp.gmail.com"
PORT=587
EMAIL_USER="tu_correo@gmail.com"
APP_PASSWORD="xxxx xxxx xxxx xxxx"  # App Password (Gmail) con o sin espacios
```

```
RECIPIENT_EMAIL="destinatario@correo.com"
SUBJECT="Reporte de Wi-Fi"
```

- **CLI y GUI** cargan automáticamente estos valores como predeterminados.
- El código sanea espacios invisibles y elimina espacios del `APP_PASSWORD`.
- `.env` ya está en `.gitignore` para evitar subir credenciales al repo.

Notas:

- Gmail requiere 2FA y **App Password**. Usa puerto 587 (STARTTLS) o 465 (SSL).
- Asegura que `EMAIL_USER` (remitente) sea el mismo usuario con el que autenticas.

Estructura rápida

- `main.py`: punto de entrada CLI (y abre la GUI si se empaqueta)
- `wifi_analyzer/platform_windows.py`: funciones específicas de Windows (`netsh`)
- `wifi_analyzer/platform_linux.py`: funciones específicas de Linux (`nmcli`, `ip`)
- `wifi_analyzer/report.py`: formatea el reporte (tabla + extras)
- `wifi_analyzer/email_utils.py`: envío de correo SMTP
- `wifi_analyzer/ui.py`: interfaz gráfica (Tkinter)

Uso (CLI)

Desde terminal en el directorio del proyecto:

```
python3 main.py
```

Flujo típico:

- El programa analiza perfiles.
- Muestra un menú con opciones:
  1. Solo ver resultados en pantalla
  2. Guardar resultados en archivo
  3. Enviar reporte por correo
  4. Salir
  5. Abrir interfaz gráfica
- Puedes elegir incluir **información adicional** (interfaces, drivers, etc.).

Guardar a archivo (ejemplo):

- Elige opción 2, da un nombre o presiona Enter para `wifi_results.txt`.

Enviar correo (ejemplo):

- Opción 3 y completa SMTP/puerto/credenciales/destino.
  - Si existe `.env`, los campos aparecerán prellenados; presiona Enter para aceptar.

Uso (GUI)

Tienes dos caminos:

- Abrir desde el **menú CLI** (opción 5).
- O desde Python:

```
from main import WiFiAnalyzer
from wifi_analyzer.ui import run_gui

run_gui(WiFiAnalyzer())
```

En la GUI encontrarás:

- Botón **Analizar** para generar el reporte.
- **Checkbox** para incluir información adicional.
- **Botones de secciones**: ver solo Interfaces, Drivers, IP, MAC, Export.
- **Guardar** y **Enviar correo** con diálogos sencillos.
  - Si existe `.env`, el diálogo se prellena automáticamente.

Ejecutables precompilados

Si ya generaste los binarios, puedes ejecutarlos directamente:

- **Linux**: en `dist_linux/`

```
chmod +x dist_linux/main
./dist_linux/main
```

- **Windows**: en `dist_windows/`

```
.\dist_windows\main.exe
```

Notas:

- El `.exe`/binario abre la **GUI por defecto** cuando está empaquetado (según `main.py`).
- Para usar la **CLI** en Windows, ejecuta desde una terminal y/o genera el build con consola (`--console`).
- En Linux, asegúrate de tener `python3-tk` instalado para la GUI.

Generar ejecutable (build)

Linux (binario nativo)

1. Crear venv e instalar PyInstaller

```
python3 -m venv .venv
source .venv/bin/activate
pip install --upgrade pip
pip install pyinstaller
```

2. Construir

```
pyinstaller --onefile --noconsole main.py
# si quieres ver logs en terminal:
# pyinstaller --onefile --console main.py
```

3. Ejecutar

```
./dist_linux/main
```

Windows (.exe)

1. En una máquina Windows:

```
py -m venv .venv
.\.venv\Scripts\activate
pip install --upgrade pip
pip install pyinstaller
```

2. Construir `.exe` (GUI por defecto)

```
pyinstaller --onefile --noconsole main.py
```

3. Ejecutar

```
.\dist_windows\main.exe
```

Si alguna vez no detecta Tkinter:

```
pyinstaller --onefile --noconsole --hidden-import=tkinter main.py
```

Problemas comunes

- "No module named 'tkinter'":
  - Linux: `sudo apt install -y python3-tk`
  - Windows: usa instalador oficial de Python (incluye Tcl/Tk)
- PEP 668 / entorno gestionado (Ubuntu):
  - Usa un **virtualenv** antes de `pip install`:

```
sudo apt install -y python3.12-venv
python3 -m venv .venv
source .venv/bin/activate
```

- En Linux no aparecen extras:
  - Asegúrate de tener `nmcli` (NetworkManager) y `ip` disponibles.
- En Windows, SMTP falla:
  - Gmail requiere App Password con 2FA.

Aviso legal

Este proyecto es **exclusivamente educativo**. No está diseñado para acceder a información de terceros. Úsalo **solo** en equipos propios o con autorización explícita.

Créditos y licenciamiento

- Hecho con Python estándar (`os`, `subprocess`, `re`, `smtplib`, `tkinter`).
- Empaquetado con **PyInstaller**.
- Ajusta y reutiliza a tu gusto respetando el aviso legal.