Cyber-Lab Dashboard Launcher

A minimal example showing how to turn a Windows **batch script** into a polished `` with a custom icon — **entirely from Kali Linux** using the MinGW cross-compiler.

Table of Contents

What it does

When run on Windows, `` will:

1. Relaunch itself **minimized** via PowerShell (so the user never sees a console window).
2. `ping` a lab IP (`10.10.10.1`).
3. Wait 5 seconds.
4. Start Microsoft Edge with your internal URL: `http://10.10.10.30:8000/`.

All logic is implemented in plain C; the icon is embedded as a Win32 resource.

Repository layout

```
.
├── src/
│   ├── open_dash.c      # C implementation of the original batch logic
│   ├── icon.rc          # Win32 resource script (references the ICO)
│   └── open-dash.ico    # Multi-resolution icon (16-256 px)
├── build.sh             # One-liner build helper
└── README.md            # You are here 📖
```

> **Note:** `open_dash.c` works as both **GUI** and **console** app. Add or remove the `-mwindows` flag while linking depending on your need.

Prerequisites

Tool	Package on Kali	Purpose
MinGW-w64	`mingw-w64`	Cross-compiling Windows binaries
Win resource	part of MinGW	`windres` compiles `.rc` → `.res`
Wine (optional)	`wine`	Quick local testing of the generated `.exe`
ImageMagick OR icoutils	`imagemagick` **or** `icoutils`	Convert PNG → ICO

Install everything in one go:

```
sudo apt update
sudo apt install mingw-w64 wine imagemagick -y
```

(swap `imagemagick` *for* `icoutils` *if preferred)*

Quick build

Inside the repo root, run:

```
./build.sh
```

That produces `open-dash.exe` in the project root. Test it with:

```
wine open-dash.exe   # optional, Linux-only test
```

build.sh contents

```
#!/usr/bin/env bash
set -e
cd "$(dirname "$0")/src"

# 1) ensure icon.rc exists (over-write in case you changed the ICO path)
echo 'IDI_ICON1 ICON "open-dash.ico"' > icon.rc

# 2) compile resources → COFF object
x86_64-w64-mingw32-windres icon.rc -O coff -o icon.res
```

```
# 3) compile + link (GUI app, stripped)
x86_64-w64-mingw32-gcc open_dash.c icon.res -o ../open-dash.exe -mwindows -s

echo "[+] Built ../open-dash.exe"
```

Make it executable:

```
chmod +x build.sh
```

Step-by-step

1. Create / convert the icon

```
# using ImageMagick
convert open-dash-512.png -define icon:auto-resize=256,128,64,48,32,16 open-
dash.ico
```

2. Resource script

```
// src/icon.rc
IDI_ICON1 ICON "open-dash.ico"
```

3. C source

```
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>

int main() {
    const char *tmpDir = getenv("TEMP");
    char flagPath[MAX_PATH];
    snprintf(flagPath, MAX_PATH, "%s\\minimized.flag", tmpDir);

    if (GetFileAttributesA(flagPath) == INVALID_FILE_ATTRIBUTES) {
        FILE *f = fopen(flagPath, "w");
        if (f) fclose(f);

        char exePath[MAX_PATH];
        GetModuleFileNameA(NULL, exePath, MAX_PATH);
```

```
        char cmd[MAX_PATH + 200];
        snprintf(cmd, sizeof(cmd),
                 "powershell -windowstyle minimized -command \"Start-Process
'%s'\"",
                 exePath);
        system(cmd);
        return 0;
    }

    DeleteFileA(flagPath);
    system("ping 10.10.10.1");
    Sleep(5000);
    system("start msedge.exe http://10.10.10.30:8000/");

    return 0;
}
```

4. Compile manually (without `build.sh`)

```
cd src
x86_64-w64-mingw32-windres icon.rc -O coff -o icon.res
x86_64-w64-mingw32-gcc open_dash.c icon.res -o ../open-dash.exe -mwindows -s
```

Credits

- Original batch logic by **Jason**.
- Cross-compilation & write-up: ChatGPT assist.

Happy hacking! Pull requests welcome.