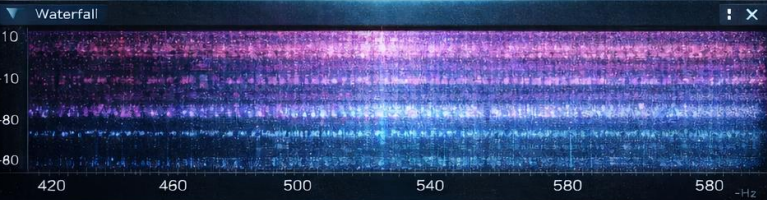


DRONE-SCAN

RF DETECTION PROGRAM



Cursor: Bin=476 F=455.200 MHz P=-31.3 dB



RF SCAN



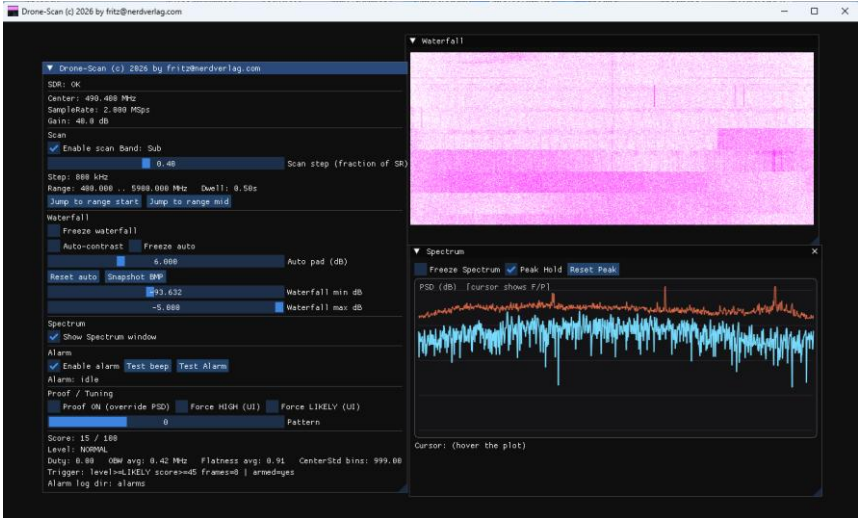
SPECTRUM & WATERFALL



ALARM DETECTION

© 2026 by fritz@nerdverlag.com

Inhaltsverzeichnis



..... 6

Drone-Scan – Installation Guide..... 6

1. Windows Installation (Windows 10 / 11, x64)..... 6

 1.1 Overview 6

 1.2 Requirements 7

 1.3 Extract the ZIP Package..... 7

 1.4 IMPORTANT: Start the Program via run.bat..... 8

 1.5 Windows Firewall Prompt (Normal!) 8

 1.6 Driver Installation (Windows) 9

 1.7 Hardware Configuration (config.ini) 10

2. Linux Installation (Ubuntu / Debian / Arch) 11

 2.1 Overview 11

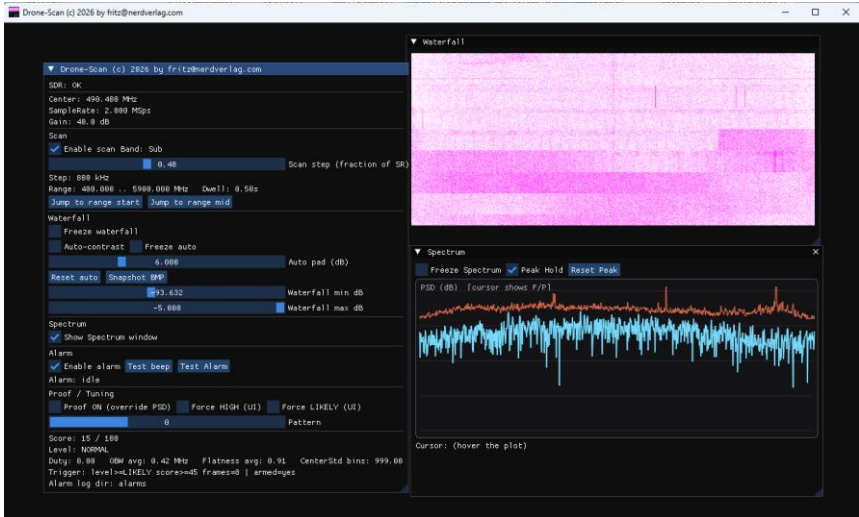
 2.2 Requirements 11

 2.3 Installation Using install_environment.sh (Recommended)....12

2.4 Manual Installation (Optional)	13
2.5 Build & Start	14
2.6 USB Access & Permissions	14
2.7 Test SDR Detection	14
2.8 Configuration (Linux is identical to Windows)	15
3. Important Note (Linux)	15
4. Common Errors & Quick Fixes	16
Drone-Scan – User Manual	17
5. What is Drone-Scan?	17
6. Basic Principle (Important to Understand)	17
7. Supported Hardware	18
7.1 ADALM-PlutoSDR (Recommended)	18
7.2 HackRF One	18
7.3 RTL-SDR (Limited!)	19
8. Start & Installation (Windows)	19
8.1 Important: Always Start via run.bat	19
8.2 Windows Firewall Prompt (No Reason to Worry)	20
9. Chapter: Configuration (config.ini)	20
9.1 Section [sdr] – SDR Hardware & Base Parameters	21
9.2 Section [scan] – Frequency Scan	21
9.3 Section [ui] – Display	23
9.4 Section [alarm] – Acoustic Alarm	23
9.5 Section [detection] – Detector Trigger (Critical!)	24
9.6 Section [proof] – Proof / Verification Mode	24
9.7 Important Notes	25

10. Recommended Default Configuration (Practical)	25
11. Scan Logic (Visible and Intentionally Slow)	26
12. Waterfall Display	26
13. Detection & Alarm	27
14. Alarm Control	28
15. Scanning Outside ISM Bands	28
16. Common Issues & Fixes	29
17. Linux Version	29
18. Important Disclaimer	30
19. Main Window “Drone-Scan”	30
19.1 SDR Status	30
20. Scan Control	31
21. Waterfall Display	31
22. Spectrum Window	32
23. Alarm Control	33
24. Proof / Tuning	34
25. Detection Status	34
26. Window Layout	35
27. Recommended Operating Procedure	35
28. Alarm Logging & Event Documentation	35
28.1 When is logging performed?	36
28.2 What is stored?	36
28.3 Storage location	37
28.4 File naming	37
28.5 Latching alarm vs. one-time logging	37

29. Appendix	38
1. What you can do with it right now (no code changes).....	38
1.1 Create an RF situational picture.....	38
1.2 Identify interference sources (indirectly)	38
1.3 Training & education	39
2. Things that are almost obvious.....	39
2.1 Silent perimeter monitor	39
2.2 RF recorder (post-event analysis)	40
2.3 Compare hardware / antennas.....	40
3. Things you can build from it	41
3.1 “Drone probability” instead of alarm	41
3.2 Mobile field solution	41
3.3 Complement to other systems	41



Drone-Scan – Installation Guide

Windows & Linux

1. Windows Installation (Windows 10 / 11, x64)

1.1 Overview

Drone-Scan for Windows is delivered as a ready-to-run ZIP package.

✓ No installer

- ✓ No registry entries
 - ✓ No system changes
 - ✓ Fully portable – runs from a single folder
-

1.2 Requirements

- Windows 10 or Windows 11 (64-bit)
 - Administrator rights only required for driver installation
 - Supported SDR hardware:
 - ADALM-PlutoSDR
 - HackRF One
 - RTL-SDR (E4000 / R820T2, limited)
-

1.3 Extract the ZIP Package

1. Download the ZIP file
2. Run Drone-Scan-Windows-x64.exe.
3. The zip file is extracted to, for example,

`C:\Drone-Scan\`

The folder contains, among others:

- `drone_scan.exe`
- `run.bat`
- `config.ini`
- **required .dll files**
- `SoapySDR\modules0.8\` (SDR drivers/modules)

⚠ Do not move individual files separately – everything must remain in the same directory structure.

1.4 IMPORTANT: Start the Program via run.bat

👉 On Windows, Drone-Scan must always be started via `run.bat`.

Why?

Windows searches for DLLs in the system path first.

`run.bat` ensures that the included, tested DLLs are used.

Correct:

- Double-click `run.bat`

Wrong:

- Launch `drone_scan.exe` directly
-

1.5 Windows Firewall Prompt (Normal!)

On first launch you may see:

“Do you want to allow this app to access private networks?”

Reason:

- SoapySDR scans for SDR devices on startup
- ADALM-PlutoSDR can be accessed via USB or network

Important:

- Drone-Scan does not open a server
- No data is transmitted
- No internet access is required

Recommended choice:

- ✓ Private network → Allow
- ✗ Public network → Optional

This prompt appears only once.

1.6 Driver Installation (Windows)

ADALM-PlutoSDR

1. Download driver package:
2. <https://github.com/analogdevicesinc/plutosdr-m2k-drivers-win/releases>
3. Run the installer
4. Connect the Pluto
5. Check in Device Manager (no warning symbol)

RTL-SDR

- Driver is included
- If needed: use Zadig (WinUSB)
(only if you have problems)

HackRF One

1. Install HackRF driver:
 - o libusb + HackRF driver (e.g., via Zadig or a driver package)
 2. libhackrf.dll must be present in the Drone-Scan folder
-

1.7 Hardware Configuration (config.ini)

Hardware selection is done via `config.ini`.

A sample `config.ini` is provided for each supported device.

PlutoSDR

```
sdr.uri = ip:pluto.local  
driver=pluto
```

or

```
sdr.uri = usb:*  
driver=pluto
```

RTL-SDR

```
sdr.uri = driver=rtlsdr
```

HackRF

```
sdr.uri = driver=hackrf
```

Enter only `driver=`!

After any changes:

→ Restart the program

2. Linux Installation (Ubuntu / Debian / Arch)

2.1 Overview

On Linux, Drone-Scan runs natively.

To simplify installation, a setup script is included:

```
install_deps.sh
```

This script installs all required dependencies automatically (compiler, SDL2, SoapySDR, correct SDR modules).

- ✓ Recommended for all users
- ✓ Saves time
- ✓ Avoids version mismatches

2.2 Requirements

- Linux x86_64
- Internet access for package installation
- sudo rights
- Supported distributions:
 - Ubuntu / Debian
 - Arch Linux
 - compatible derivatives

2.3 Installation Using `install_environment.sh` (Recommended)

Step 1 – Make the script executable

Extract the `Drone-Scan-linux-multi.tar.gz` file:
`tar -xzvf Drone-Scan-linux-multi.tar.gz`

If necessary, run ``chmod +x install_deps.sh`

Step 2 – Run the installer

```
install_deps.sh
```

or (if required):

```
sudo ./ install_deps.sh
```

What does the script do?

`install_environment.sh` installs automatically:

- Base build tools:
 - GCC / Clang
 - CMake
 - SDL2
 - SoapySDR
- SDR modules:
 - RTL-SDR
 - HackRF
 - LimeSDR
 - PlutoSDR
- Sets required udev rules (USB permissions)
- Verifies SoapySDR installation with:
 - `SoapySDRUtil --find`

⚠ The script does not install proprietary drivers (e.g., Pluto firmware remains unchanged).

2.4 Manual Installation (Optional)

⚠ For advanced users only

⚠ Only works if you own the source code.

If you do not want to use the script:

If you do not want to use the script:

Ubuntu / Debian

```
sudo apt update
sudo apt install -y \
  build-essential cmake git \
  libsdl2-dev \
  soapy-sdr libsoapysdr-dev \
  soapysdr-tools \
  soapysdr-module-rtlsdr \
  soapysdr-module-hackrf \
  soapysdr-module-lms7 \
  soapysdr-module-plutosdr
```

Arch Linux

```
sudo pacman -S \
  base-devel cmake git \
  sdl2 \
  soapysdr \
  soapysdr-rtlsdr \
  soapysdr-hackrf \
  soapysdr-lms7 \
  soapysdr-plutosdr
```

2.5 Build & Start

```
mkdir build
cd build
cmake ..
make -j
```

Start:

```
./drone_scan
```

2.6 USB Access & Permissions

Recommended:

```
sudo usermod -a -G plugdev $USER
```

Then:

- logout / login

Alternative (not recommended):

```
sudo ./drone_scan
```

2.7 Test SDR Detection

```
SoapySDRUtil --find
```

Example:

```
Found device 0
  driver = hackrf
```

If no device appears here → driver issue, not a Drone-Scan issue.

2.8 Configuration (Linux is identical to Windows)

`config.ini` is platform independent.

A sample config exists for each hardware type.

Examples:

PlutoSDR

```
sdr.uri = ip:pluto.local  
driver=pluto
```

RTL-SDR

```
sdr.uri = driver=rtlsdr  
driver=rtlsdr
```

HackRF

```
sdr.uri = driver=hackrf  
driver = hackrf
```

Enter only `driver = ...!`

3. Important Note (Linux)

Drone-Scan does not use special kernel modules and does not modify the system, except for:

- installing packages
- optional udev rules

All changes are transparent and can be reverted.

4. Common Errors & Quick Fixes

Program does not start (Windows):

- use `run.bat`
- are all DLLs present in the folder?

No SDR found:

- driver installed?
- check `SoapySDRUtil --find`
- is `config.ini` correct?

No alarm:

- signal \neq drone
 - confidence too low
 - check detection parameters
-
-

Drone-Scan – User Manual

RF-based drone signal detection with SoapySDR

Version: 0.1.x

Platforms: Windows (x64), Linux (x86_64)

Supported SDRs: ADALM-Pluto, HackRF, RTL-SDR (limited)

5. What is Drone-Scan?

Drone-Scan is a passive RF analysis tool that analyzes radio activity in typical drone frequency ranges and detects drone-like signal patterns.

The program:

- does not receive telemetry
- does not decode Remote-ID
- does not control any devices
- does not send data to the internet

Drone-Scan detects signal behavior, not drone models.

6. Basic Principle (Important to Understand)

Drone-Scan works in four steps:

1. SDR receives IQ samples

2. FFT / PSD analysis
3. Feature extraction
(bandwidth, time behavior, spectral shape)
4. Scoring (0–100)

An alarm is not triggered by “a strong signal”, but by:

- sufficiently wide signals
- stable time behavior
- typical dynamics (duty cycle, drift, flatness)

- 👉 Therefore, a strong signal may appear without an alarm
- 👉 and a weaker but typical signal may trigger an alarm

7. Supported Hardware

7.1 ADALM-PlutoSDR (Recommended)

- Frequency range: 70 MHz – 6 GHz
- Full support for all scan bands
- Very stable
- Best choice for 2.4 GHz / 5.8 GHz

Recommended usage:

- USB or network (`pluto.local`)
- fixed `sdr.uri` recommended

7.2 HackRF One

- Frequency range: 1 MHz – 6 GHz
 - Supports 2.4 GHz & 5.8 GHz
 - Lower dynamic range than Pluto
 - Very flexible
-

7.3 RTL-SDR (Limited!)

- Frequency range: ~24 MHz – 1.7 GHz
- NO 2.4 GHz
- NO 5.8 GHz
- Only Sub-GHz makes sense

👉 Drone-Scan automatically blocks unsupported scan bands when RTL-SDR is selected.

8. Start & Installation (Windows)

8.1 Important: Always Start via run.bat

Do not start `drone_scan.exe` directly!

`run.bat` ensures:

- correct DLL path setup
 - correct loading of SoapySDR modules
 - reproducible behavior
 - ❌ absolutely no changes to the Windows system
-

8.2 Windows Firewall Prompt (No Reason to Worry)




On first start:

“Allow access to public/private networks?”



Why?

- SoapySDR searches for devices
- Pluto can be accessed via network (`pluto.local`)

Important:

-  Drone-Scan does not run a server
-  no telemetry
-  no internet traffic


Recommendation:

-  Private: Allow
-  Public: Optional

9. Chapter: Configuration (`config.ini`)

The `config.ini` controls hardware, scan behavior, detection, alarm, and visualization.

All settings are loaded at program start.

 Changing `config.ini` requires restarting the application.

9.1 Section [sdr] – SDR Hardware & Base Parameters

```
[sdr]
uri =
driver =
sample_rate = 2000000
bandwidth = 2000000
fft_size = 1024
gain = 20
center_freq = 433000000
```

Parameter	Meaning
uri	Optional device address (e.g. ip:192.168.2.1 for PlutoSDR)
driver	Optional driver hint (plutosdr, hackrf, rtl-sdr, empty = auto)
sample_rate	Sample rate in Hz
bandwidth	RF bandwidth in Hz (usually same as sample_rate)
fft_size	FFT size for analysis and display
gain	RX gain in dB
center_freq	Initial center frequency (overridden by scan)

Notes:

- Supports PlutoSDR, HackRF, RTL-SDR
- `center_freq` is only an initial value, not a scan limit

9.2 Section [scan] – Frequency Scan

```
[scan]
enabled = true
```

```
start_freq_sub = 433000000
end_freq_sub   = 434800000
```

```
dwell_time = 0.35
step_frac   = 0.40
```

Basic principle:

➔ Only ONE band may be active at a time.

A band is active if start and end frequency are > 0 .

Band priority:

1. Sub-GHz
2. 2.4 GHz
3. 5.8 GHz

Parameter	Meaning
enabled	scan on/off
start_freq_sub / end_freq_sub (start/end for 2.4 and 5.8 GHz accordingly)	Sub-GHz scan range
dwell_time	dwell time per hop (seconds)
step_frac	step size relative to sample_rate

Example:

sample_rate = 2,000,000 Hz


step_frac = 0.40

→ step = 800 kHz

9.3 Section [ui] – Display

```
[ui]
waterfall_min_db = -95
waterfall_max_db = -25
```

Parameter	Meaning
waterfall_min_db	lower dB bound of waterfall
waterfall_max_db	upper dB bound

 These are start values. Auto-contrast can adjust dynamically during runtime.

9.4 Section [alarm] – Acoustic Alarm

```
[alarm]
enabled = true
repeat_interval = 2.0
sound_frequency = 1000
sound_duration = 0.4
```

Behavior:

- alarm is latching (stays active)
- ends only after manual acknowledgment
- audio output via SDL audio (Windows & Linux)

Parameter	Meaning
enabled	enable/disable alarm
repeat_interval	time between beep repeats
sound_frequency	beep frequency in Hz
sound_duration	beep duration in seconds

9.5 Section [detection] – Detector Trigger (Critical!)

```
[detection]
trigger_level = LIKELY
trigger_score = 45
trigger_frames = 8
rearm_release_frames = 20
```

Parameter	Meaning
trigger_level	minimum detection level (LIKELY or HIGH)
trigger_score	minimum score (0–100)
trigger_frames	required consecutive hits
rearm_release_frames	frames without hits to re-arm

Goal:

- no random alarms
- no flickering
- stable, reproducible triggering

9.6 Section [proof] – Proof / Verification Mode

```
[proof]
enabled = false
pattern = 0
```

Purpose: Proof mode generates synthetic drone-like spectra for:

- functional testing
- demonstrations
- documentation
- training
- customer proof

⚠ No RF transmission – software simulation only.

Patterns:

Value	Characteristics
0	FHSS + broad occupancy (DJI / ELRS-like)
1	multi-carrier / OFDM-like
2	very wide noise-like band

9.7 Important Notes

- `config.ini` changes take effect only after restart
 - scan and proof mode can be used together
 - UI “Force HIGH / LIKELY” affects display only, not alarm
 - alarm stays active until acknowledged
-

10. Recommended Default Configuration (Practical)

```
[detection]
trigger_level = LIKELY
trigger_score = 45
trigger_frames = 8
```

```
rearm_release_frames = 20
```

➔ Stable, realistic, demo-friendly, without needing “Force HIGH”.

What happens with wrong settings?

- RTL-SDR + 2.4 GHz → scan automatically disabled
- frequencies outside device range → no crash
- scan disabled → fixed center frequency

11. Scan Logic (Visible and Intentionally Slow)

- scan hops step-by-step
- step size \approx sample_rate \times factor
- intentionally not too fast so that:
 - waterfall remains visually understandable
 - detection becomes more stable

12. Waterfall Display

Manual settings:

- waterfall_min_db
- waterfall_max_db

Auto-Contrast:

- dynamically adjusts scaling based on PSD min/max
- Freeze Auto keeps the current scale

Freeze Waterfall:

- stops visual update
- SDR + detection continue running

Snapshot:

- saves current waterfall as BMP
 - useful for documentation/analysis
-

13. Detection & Alarm

Key point:

Not every “strong signal” is a drone.

Detector evaluates, among others:

- occupied bandwidth
- time stability
- spectral flatness
- center drift

Alarm requires:

- score above threshold
- stable over multiple frames
- drone-like characteristics

Therefore:

- Wi-Fi → often no alarm
 - narrow carriers → no alarm
 - wide, stable modulated signals → alarm
-

14. Alarm Control

```
[alarm]
enabled = true
repeat_interval = 2.0
sound_frequency = 1000
sound_duration = 0.4
```

UI functions:

- alarm on/off
- test beep
- acknowledge alarm

Audio is asynchronous and does not block the UI.

15. Scanning Outside ISM Bands

Technically:

- ✓ possible
- ✓ no SDR issue
- ✓ detector still works

Legally:

⚠ depends on your country

- △ receiving is often allowed
- △ transmitting is not allowed!

Drone-Scan never transmits, it is fully passive.

16. Common Issues & Fixes

“No alarm although signal is visible”

- bandwidth too small
- signal too unstable
- gain set wrong
- detection thresholds too strict

“Waterfall empty”

- gain too low
- wrong frequency range
- scan disabled

“RTL-SDR crashes at 2.4 GHz”

- hardware limitation (expected)
 - use Sub-GHz instead
-

17. Linux Version

- same config.ini
- same scan logic

- same UI
- same modules

Differences:

- no Windows beep → terminal beep (if audio disabled)
 - udev permissions required for SDR
-

18. Important Disclaimer

Drone-Scan is:

- an analysis tool
- not a surveillance system
- no identification system
- no legal assessment tool

Results support situational assessment, not legal proof.

19. Main Window “Drone-Scan”

19.1 SDR Status

At the top you see SDR device status:

- **SDR: OK** → device detected and running
- **SDR: NOT OK** → device unavailable or init failed
 - reason displayed below

Also shown:

- current center frequency
 - sample rate
 - gain
-

20. Scan Control

Section: **Scan**

Controls frequency scanning.

Elements:

- Enable scan
- Band: shows active band (Sub / 2.4G / 5.8G from config.ini)
- Scan step (fraction of SR): hop step size relative to sample rate
- Step: effective hop step size in kHz
- Range: scan range in MHz + dwell time

Buttons:

- Jump to range start
 - Jump to range mid
-

21. Waterfall Display

Section: **Waterfall**

Controls:

- Freeze waterfall
- Auto-contrast
- Freeze auto
- Auto pad (dB)

Buttons:

- Reset auto
- Snapshot BMP

Manual scale:

- Waterfall min dB
 - Waterfall max dB
-

22. Spectrum Window

Window: **Spectrum**

Features:

- live PSD curve
- Peak Hold (max per bin)
- mouse cursor/marker

Mouse shows:

- FFT bin
- absolute frequency (MHz)
- power (dB)

Controls:

- Freeze Spectrum
 - Peak Hold
 - Reset Peak
-

23. Alarm Control

Section: **Alarm**

Elements:

- Enable alarm
- Test beep
- Test Alarm

Alarm behavior (important):

- alarm is latching
- stays active until acknowledged
- no auto shut-off

Display:

-  ALARM ACTIVE (latched)

Button:

- Acknowledge alarm
After acknowledging, a short re-arm phase prevents immediate retriggering.

24. Proof / Tuning

Section: **Proof / Tuning**

Elements:

- Proof ON (override PSD)
- Force HIGH (UI)
- Force LIKELY (UI)



Important:

These options affect display only – not the alarm.

Pattern (0–2): selects synthetic test signal type.

25. Detection Status

Section: **Detection output**

Shows:

- Score (0–100)
- Level (NORMAL / LIKELY / HIGH)
- Duty
- OBW (Occupied Bandwidth)
- Spectral Flatness
- Center Stability
- Trigger parameters
- armed / not armed status

26. Window Layout

- windows can be moved freely
- windows can be closed and reopened
- sizes are dynamically adjustable

27. Recommended Operating Procedure

1. connect SDR
2. enable scan
3. observe waterfall & spectrum
4. enable alarm
5. on alarm:
 - visual verification
 - acknowledge alarm
6. optional:
 - save snapshot
 - use proof mode for demonstration

28. Alarm Logging & Event Documentation

Drone-Scan can automatically generate documentation when a real alarm occurs. It creates a spectrum snapshot (PNG) and a text file (TXT) with relevant data, so events remain traceable later (time, frequency, detection result).

28.1 When is logging performed?

A log entry is created only if the detection logic actually triggers an alarm.

✓ Logging happens when:

- alarm is triggered by detector conditions
(trigger level + trigger score + trigger frames)

✗ Logging does not happen when:

- Test beep
 - Test Alarm
 - Proof mode
 - Force HIGH / Force LIKELY (UI debug/visual only)
 - “signal visible” without meeting trigger conditions
-

28.2 What is stored?

Two files per alarm event:

1. Spectrum snapshot (PNG)
 - screenshot of the spectrum at alarm time
 - provides visual evidence of the spectral situation
2. Text file (TXT)
Contains metadata such as:
 - date & time (timestamp)
 - center frequency at alarm time

- active band (Sub / 2.4G / 5.8G)
 - detection status (level, score)
 - trigger config (trigger_level, trigger_score, trigger_frames)
 - SDR and scan settings (sample rate, FFT size, gain, dwell, step)
-

28.3 Storage location

By default logs are stored in the program folder (or below it), typically:

- `./alarms/`
or the working directory used to start Drone-Scan.

Recommendation: do not delete this folder if logs are needed for traceability.

28.4 File naming

Example:

- `alarm_2025-12-29_08-41-12_433.920MHz.txt`
 - `alarm_2025-12-29_08-41-12_433.920MHz_spectrum.png`
-

28.5 Latching alarm vs. one-time logging

Drone-Scan uses a latching alarm:

- once active it remains active until acknowledged.

Logging is separate and normally happens once per alarm trigger, not continuously.

29. Appendix

1. What you can do with it right now (no code changes)

1.1 Create an RF situational picture

Drone-Scan is essentially an RF situational awareness tool. You can:

- see which bands are active
- observe when activity starts/ends
- compare time-of-day, locations, events

Ideal for:

- site survey
 - pre-checks (e.g., events)
 - long-term observation
-

1.2 Identify interference sources (indirectly)

Even though it's called "Drone-Scan", many RF systems behave differently:

- Wi-Fi
- video links
- FHSS systems
- wideband interferers
- faulty devices

You learn:

- what is *not* a drone signal
 - what looks "typically technical" vs. "typically UAV-like"
-

1.3 Training & education

Very valuable for:

- staff training
- RF fundamentals demonstrations
- "this is what 2.4 GHz really looks like"
- comparing SDRs

Drone-Scan is visual, stable, and explainable → perfect for training.

2. Things that are almost obvious

2.1 Silent perimeter monitor

- fixed frequency
- alarm on change
- no scanning
→ RF guard post

Use cases: rooftop, perimeter, temporary setups, passive monitoring.

2.2 RF recorder (post-event analysis)

You already have:

- PSD
 - time reference
 - detection metrics
 - event logging
→ offline analysis: “what happened at 21:37?”
-

2.3 Compare hardware / antennas

Perfect for objective comparisons:

- antennas
- locations
- gain settings
- Pluto vs HackRF vs RTL-SDR

Same software, same UI, same logic → comparable results.

3. Things you can build from it

3.1 “Drone probability” instead of alarm

You already have:

- score
- level

This invites:

- traffic-light logic
- heatmaps
- time charts

Operationally better than a binary “ALARM!”.

3.2 Mobile field solution

With: laptop + Pluto/HackRF + directional antenna you can:

- walk areas
- find hotspots
- observe changes manually

Not full direction finding, but strong “direction and proximity” intuition.

3.3 Complement to other systems

Drone-Scan works well as:

- passive sensor
- early warning
- confirmation sensor

Combine with optical systems, radar, ADS-B, acoustic sensors.