SignalScope

Comprehensive User Guide

Broadcast Signal Intelligence Platform

Current as of build SignalScope-3.5.104

https://github.com/itconor/SignalScope

Open Source • Flask + RTL-SDR + Python

For broadcast engineers and radio monitoring professionals

Table of Contents

Chapter 1

Introduction

SignalScope is a web-based radio monitoring and signal analysis platform for broadcast engineers. It runs as a Flask web application, accessible via any browser on any device. It supports stand-alone monitoring nodes and distributed multi-site hub deployments, giving engineering teams a single pane of glass across an entire broadcast estate.

Key Capabilities

- FM monitoring via RTL-SDR dongles with full RDS decoding (PS, RadioText, stereo, TP, TA, PI)
- DAB digital radio monitoring via RTL-SDR and welle-cli
- Livewire / AES67 (RTP multicast) monitoring with packet loss and jitter metrics (RFC 3550 EWMA)
- HTTP / HTTPS audio stream monitoring — any format decodable by ffmpeg
- Local sound device monitoring — microphone, line-in, USB audio, loopback
- Real-time level (dBFS), LUFS Momentary / Short-term / Integrated (EBU R128), and silence detection
- AI-powered anomaly detection — per-stream ONNX autoencoder trained on 14 audio features
- Broadcast chain fault location — identifies the first failed node in a signal path
- Hub mode for multi-site aggregation — centralised monitoring of unlimited remote nodes
- Codec Monitor — real-time connection tracking for Comrex, Tieline, Prodys, and APT contribution codecs
- PTP / GPS-disciplined wall clock for studios, accurate on any browser
- Push notification server for iOS and Android via APNs and FCM
- SignalScope Player — desktop companion app (Windows & macOS) for logger recordings
- Icecast Streaming plugin — re-stream any monitored input to an Icecast2 server
- Listener plugin — polished stream player for presenters and producers
- Producer View plugin — simplified chain fault display for non-technical on-air staff
- AzuraCast plugin — live now-playing integration and silence correlation with AzuraCast web radio
- Sync Capture plugin — multi-site simultaneous audio capture for simulcast alignment verification
- Extensible plugin system for adding custom functionality alongside the core application

Architecture Overview

SignalScope is a single Python file (**signalscope.py**) built on Flask. It can run in three modes:

Mode	Description
Standalone	Monitors local inputs only. All dashboards, alerts, and history are local.
Hub Client	Monitors local inputs and reports to a central hub. The hub aggregates status from all clients.
Hub Server	Aggregates data from connected client nodes. Provides a unified dashboard, chain monitoring, and hub reports. May also monitor local inputs.

Chapter 2

Installation

System Requirements

Requirement	Notes
OS	Linux — Ubuntu 22.04 LTS recommended. Also runs on Raspberry Pi OS (64-bit). macOS supported for development.
Python	3.9 or later
ffmpeg	Required for Logger plugin audio recording and clip export
welle-cli	Required for DAB digital radio monitoring
rtl-sdr tools	rtl_sdr, rtl_fm, rtl_power — required for FM, DAB, and Scanner features
libportaudio2	Required for local sound device monitoring
RAM	512 MB minimum; 1 GB recommended for hub deployments with many clients

Quick Install (Recommended)

The recommended method uses the automated installer script. Run the following command as a regular user (sudo access is required for system package installation):

```
/bin/bash <(curl -fsSL
https://raw.githubusercontent.com/itconor/SignalScope/main/install_signalscope.sh)
```

The installer performs the following steps automatically:

- Detects any existing SignalScope installation and offers to update in-place, preserving configuration
- Installs system dependencies: rtl-sdr, welle.io, libportaudio2, ffmpeg
- Creates a Python virtual environment under the SignalScope directory
- Installs Python dependencies (Flask, numpy, scipy, onnxruntime, etc.)
- Configures a systemd service for automatic start and restart on failure
- Configures a self-healing watchdog (see Watchdog section below)
- Optionally configures an NGINX reverse proxy
- Optionally obtains a Let's Encrypt TLS certificate via Certbot
- Starts the SignalScope service automatically on completion

Manual Installation

```
git clone https://github.com/itconor/SignalScope.git
cd SignalScope
```

```
bash install_signalscope.sh
```

Accessing SignalScope

Configuration	URL
Default (no NGINX)	http://localhost:5000
NGINX reverse proxy (HTTP)	http://your-domain.com
NGINX + TLS (Let's Encrypt)	https://your-domain.com

Watchdog Service

The installer configures a systemd watchdog service (**signalscope-watchdog**) that monitors TCP port 5000 (SignalScope) and ports 443/80 (NGINX if installed). Each service is restarted independently if it stops responding. The watchdog itself is monitored by systemd and automatically restarted on failure.

To view watchdog logs:

```
journalctl -t signalscope-watchdog
```

Updating

Go to **Settings** → **Maintenance** → **Apply Update & Restart**. This downloads the latest *signalscope.py* from GitHub, validates the file, replaces the running file, and restarts via systemd — all configuration and data is preserved. The update takes approximately 10–30 seconds.

Chapter 3

First Run & Setup Wizard

The setup wizard runs automatically on the first browser access after installation. It guides you through the essential configuration steps before the main dashboard is shown.

Setup Steps

Step 1: Authentication

Set an admin username and password. Passwords are hashed using PBKDF2-SHA256 with a random salt. The password is never stored in plaintext.

Step 2: SDR Configuration

The wizard scans for connected RTL-SDR dongles and displays their serial numbers. Assign each dongle a role: FM, DAB, Scanner, or None. Roles determine which features can use each dongle.

Step 3: Hub Configuration

Choose the operating mode: Standalone (no hub), Hub Client (connect to a central hub), or Hub Server (act as the aggregation point). Enter hub URL and secret key for client/server modes.

Step 4: Monitoring Settings

Configure global defaults: silence detection threshold (dBFS), minimum silence duration before alert, alert cooldown period, and notification channels (email, Teams, Pushover, webhook).

After Setup

After completing the wizard, the main dashboard loads automatically. You can revisit and modify all settings at any time via **Settings** in the top navigation bar. Settings changes take effect immediately without restart (except plugin installation and removal, which require a restart).

Chapter 4

Dashboard

The dashboard is the main view of SignalScope, showing a live card for every monitored stream. Cards update in real time without page reload.

Stream Cards

Each monitored stream is displayed as a card containing the following elements:

- Live level bar showing current audio level with a dBFS numeric readout
- LUFS Momentary, Short-term, and Integrated values (EBU R128 compliant)
- RDS Programme Service (PS) name and RadioText (RT) for FM sources
- DAB service name and DLS (Dynamic Label Segment) now-playing text for DAB sources
- AI status badge: Learning (during the 24-hour training period), OK, or Anomaly
- Trend badge when the level deviates from the expected hour-of-day baseline (amber = moderate, red = significant)
- 24-hour availability timeline bar — click to cycle between 24h, 6h, and 1h views
- Alert or warning status strip on the card border (red = alert, amber = warning)
- ■ Signal History button — expands an inline chart for any metric
- ■ Listen button — opens a live audio stream in the sticky mini-player at the bottom of the page

Card Organisation

Cards are drag-to-reorder. Alert cards automatically sort to the top of the grid. You can manually drag cards to any position; the order is persisted in your browser.

Mini-Player

Clicking ■ Listen on any card opens the sticky mini-player at the bottom of the page. The mini-player shows the stream name and level while playing. Audio uses the browser's Web Audio API and remains active while you navigate between dashboard tabs. Click the × button to close the player.

Hub Dashboard — Live Level Meters

The hub dashboard shows a **PPM-style bouncing level bar** for every stream on every connected site. These bars update at **5 Hz** — independently of the 10-second heartbeat cycle — so you see smooth real-time level animation without waiting for the next full data refresh.

Each bar has three colour zones matching standard broadcast PPM practice:

Zone	Range	Colour
Programme	–80 dBFS to –20 dBFS	Green
Near-clip warning	–20 dBFS to –9 dBFS	Amber
Clip zone	–9 dBFS to 0 dBFS	Red

A **peak-hold marker** tracks the highest recent level and decays slowly after 2 seconds, making transient peaks easy to spot at a glance.

On page load the bars immediately show the last-known levels (restored from hub state on restart) and begin animating as soon as live data arrives — typically within one to two seconds of a stream connecting to audio.

Chapter 5

Inputs

Source Types & Address Formats

Source Type	Address Format	Example
FM via RTL-SDR	fm://	fm://96.3
FM with specific dongle	fm://?serial=&ppm;=	fm://96.3?serial=00000001&ppm;=-2
DAB service	dab://?channel=	dab://Cool FM?channel=12D
Livewire / AES67 RTP	rtp://:	rtp://239.192.10.1:5004
HTTP / HTTPS stream	Full URL	http://relay.example.com:8000/stream
Local sound device	sound://	sound://2

SDR Dongle Roles

Each RTL-SDR dongle must be assigned a role in **Settings** → **SDR Devices** before it can be used. Only dongles with a compatible role appear in the dropdown when adding an input.

Role	Used By
FM	FM monitoring inputs
DAB	DAB monitoring inputs
Scanner	FM Scanner and Web SDR on-demand tuning
None	Unassigned — not available to any feature

Adding FM Sources

- 1. Settings → Inputs → + Add Input → FM
- 2. Enter the frequency in MHz (e.g. 96.3)
- 3. Select the dongle from the dropdown — must have role FM
- 4. Optionally set the PPM calibration offset for your dongle
- 5. Save — monitoring starts within seconds

Adding DAB Sources

- 1. Settings → Inputs → + Add Input → DAB
- 2. Select the DAB channel (e.g. 12D = 229.072 MHz)
- 3. Click ■ Scan Mux — welle-cli scans the multiplex and enumerates all available services

- 4. Select one or more services from the list
- 5. Click ■ Add Selected Services — each service is added with its broadcast name and correct dab:// address

Note: Multiple DAB services on the same multiplex share a single welle-cli process, saving CPU and dongle resources.

Adding Livewire / AES67

Enter the multicast RTP address and port (e.g. **rtp://239.192.10.1:5004**). SignalScope joins the multicast group using the OS network stack. Packet loss and jitter are measured using RFC 3550 EWMA and reported in the stream card and metrics history.

Adding HTTP Streams

Enter the full stream URL. HTTP and HTTPS are both supported. Any audio format decodable by ffmpeg is accepted, including MP3, AAC, Ogg/Opus, HLS, and Icecast streams.

Adding Local Sound Devices

Select **Local Sound Device** from the source type dropdown. The dropdown is populated from the OS audio device list via PortAudio. Select the desired microphone, line-in, USB audio interface, or loopback device. The device index is stored as **sound://<index>** and resolved to the device name on start; if the device index changes after a reboot, update the input address in Settings.

Chapter 6

FM Scanner

The FM Scanner provides on-demand FM reception in the browser, with live RDS decoding, tuning history, presets, and band scan. Navigate to **Hub** → ■ **Scanner** or visit **/hub/scanner**. At least one dongle with role **Scanner** must be configured.

Using the Scanner

- 1. Select a site from the dropdown — only sites with a Scanner dongle are shown
- 2. Enter an FM frequency in MHz in the frequency field
- 3. Click ■ Start — audio streams to the browser after approximately 1–2 seconds
- 4. While streaming: type a new frequency and click Tune to retune without stopping
- 5. Click ■ Stop to disconnect and release the dongle

Press **Spacebar** at any time to pause or resume audio.

Features

Feature	Description
Live RDS	PS name, RadioText, stereo flag, TP (Traffic Programme), TA (Traffic Announcement), PI (Programme Identification)
Tuning History	Recently tuned frequencies shown as clickable buttons. Click to retune from idle or while streaming.
Presets	Save frequencies with a label. Click any preset to tune immediately.
Band Scan	■ Scan Band runs rtl_power over the FM band and displays strong stations as clickable peak markers. Dongle must be free (stop streaming first).
Keyboard Shortcut	Spacebar toggles audio play/pause without clicking the button.

Audio Pipeline

The audio path is: **RTL-SDR** → **rtl_fm** → **Python resampling (scipy)** → **PCM 48 kHz 16-bit** → **hub relay slot** → **browser Web Audio API**. End-to-end latency is typically 1–2 seconds from RF signal to browser speaker. The client batches multiple PCM chunks per HTTP POST when the WAN round-trip time exceeds the block duration (0.1 s), maintaining real-time throughput over slow connections.

Chapter 7

Logger Plugin

The Logger plugin provides continuous 24/7 compliance recording of any monitored stream. Recordings are stored as clock-aligned 5-minute segments with inline silence detection, now-playing metadata integration, and a comprehensive browser-based timeline for playback and clip export. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. Requires **ffmpeg** to be installed on the host machine.

Installation & First Use

- 1. Settings → Plugins → Check GitHub for plugins
- 2. Find Logger → click ■ Install
- 3. Restart SignalScope when prompted
- 4. Navigate to Logger → Settings tab
- 5. Enable recording for each stream and configure format and retention
- 6. Save — recording starts immediately, no further restart needed

Recording Formats

Each stream has its own format setting. Existing recordings are unaffected when changing format; only new segments use the new setting.

Format	Extension	Encoder	Notes
MP3 (default)	.mp3	built-in	Universal compatibility
AAC	.aac	aac (ADTS)	~0.5× storage vs MP3 at equivalent quality
Opus	.ogg	libopus (OGG)	~0.25× storage; Chrome/Firefox/Edge/Safari 16.4+

Files are stored at: **logger_recordings/{stream-slug}/{YYYY-MM-DD}/HH-MM.{ext}**

Quality Tiers & Retention

Setting	Default	Description
Format	MP3	Recording format: MP3 / AAC / Opus
HQ Bitrate	128k	Bitrate for new recordings
LQ Bitrate	48k	Bitrate after quality downgrade
LQ after (days)	30	Re-encode to lower quality after N days
Delete after (days)	90	Permanently delete recordings after N days

A background maintenance thread runs hourly, performing re-encodes and deletions according to the configured schedule.

The Timeline View

The Timeline tab shows a full-day overview of a single stream. The layout from top to bottom:

Audio Overview Bar

A green waveform minimap showing the full 24-hour day. Click anywhere to jump to that time and load the nearest segment.

Metadata Bands

Three thin colour-coded bands between the overview bar and the hour grid:

- **Show band (purple):** show names from now-playing metadata. Consecutive events with the same show name are merged into one block.
- **Mic band (green):** on-air periods logged via the Mic REST API. Bright green = currently live.
- **Track band (amber):** individual song positions at exact timestamps. Hover for track title and time.

Zoom & Navigation Controls

Control	Action
1× / 2× / 4× / 8× zoom	Zoom the timeline horizontally. At 8× individual 5-minute blocks are large enough to read song and show labels.
↕ Expand	Doubles all band heights for easier reading.
Click and drag	Pan the timeline left and right at any zoom level.
Spacebar	Toggle playback without scrolling the page.
Right-click	Set export mark-in / mark-out directly from the timeline position.

Hour Grid

288 colour-coded 5-minute blocks arranged in an hourly grid:

Colour	Meaning
Green	Recorded, audio present
Amber	Recorded, partial silence detected
Red	Recorded, mostly or fully silent
Dark / Grey	No recording for this period

Click any block to load and play that 5-minute segment.

Clip Export

- 1. Play a segment and locate the start of the section you want to export
- 2. Click Mark In (or right-click the timeline at the start position)
- 3. Navigate to the end of the section and click Mark Out (or right-click again)
- 4. Select the export format from the dropdown
- 5. Click ■ Export Clip — ffmpeg extracts and downloads the clip

Format	Extension	Bitrate	Notes
MP3	.mp3	Original	Instant stream copy if recordings are already MP3
AAC	.m4a	128 kbps	~half size, universal support
Opus	.webm	96 kbps	Smallest; Chrome/Firefox/Edge/Safari 16.4+

Clips can span multiple 5-minute segments up to 2 hours. The filename includes stream name, date, start time, and duration.

Now-Playing Metadata Integration

Logger polls a now-playing API every 30 seconds to populate show names and track data. Configure per-stream in Logger Settings:

- **Planet Radio API:** select station from dropdown — URL auto-filled
- **Triton Digital:** enter the Triton JSON endpoint URL
- **Custom JSON API:** enter any URL; Logger extracts title, artist, and show fields
- **Fallback:** DLS (DAB) or RDS RadioText (FM) used automatically if no API is configured

Mic On-Air REST API

Record mic-on/off events from broadcast automation systems, physical hardware controllers, or any HTTP-capable device. Events appear immediately as green spans on the mic band.

Endpoint: POST /api/logger/mic

Authentication: Bearer token (set Mic API Key in Logger Settings) or logged-in session

```
{
"stream": "cool-fm",
"state": "on",
"label": "Studio A",
"ts": 1711234567 // optional Unix timestamp, defaults to server time
}
```

Hub Mode — Centralised Playback

When SignalScope is running as a hub, the Logger aggregates recordings from all connected client nodes for centralised playback. No files are copied — audio streams on demand through the hub relay pipeline in real time.

Select any site from the site dropdown in the Logger header to browse and play that site's recordings without logging into the client node individually. The hub issues a **stream_file** command to the client, which opens the requested segment with ffmpeg and pushes raw audio bytes through the relay slot to your browser or desktop player.

Seeking: when you click a specific time on the day bar or use the skip controls, the hub passes a **seek_s** value to the client. ffmpeg performs a fast key-frame seek before streaming, so playback starts at the requested wall-clock position within 1–2 seconds.

Multi-Node Shared Recordings (Sidecar JSON)

Multiple Logger instances on different nodes can share a common recording storage directory (NFS, SMB, or any shared filesystem). Each instance writes a per-day metadata sidecar file (**meta_{owner}.json**) alongside the audio segments. The Logger UI merges all sidecar files at load time, so the timeline shows show names, mic spans, and track bands from every contributing node without any network coordination or locking.

SignalScope Player Integration

SignalScope Player is a desktop companion application (see Chapter 17) that connects directly to a hub's Logger to browse and play recordings offline. In Hub mode the player authenticates with the hub URL and an API token, lists available sites and dates, and streams segments through the same hub relay pipeline used by the browser player. Skip controls and exact-time seeking work identically to the browser interface.

Chapter 8

DAB Scanner Plugin

The DAB Scanner plugin enables on-demand scanning of DAB Band III channels to discover services, and streams any discovered service as MP3 audio directly to the browser. Install from **Settings** → **Plugins**. Requires **welle-cli** and **ffmpeg** on the client machine. Navigate to **/hub/dab**.

Features

- Scans all Band III DAB channels (5A–13F) to discover available ensembles and their services
- Streams any selected service as MP3 audio via welle-cli and ffmpeg
- Displays DLS (Dynamic Label Segment) scrolling text in real time
- Channel list shows signal quality indicators (SNR, FIC quality)
- Automatically stops scanning when a channel is selected for playback

Chapter 9

Meter Wall Plugin

The Meter Wall plugin provides a full-screen audio level display for all monitored streams across all connected hub sites. Install from **Settings** → **Plugins**. Navigate to **/hub/meterwall**. Designed for a dedicated monitor in a transmission control room or master control room.

Features

- PPM-style level bars with peak hold and natural decay for every stream
- LUFS Integrated readout per stream
- Alert flash on silence detection or clip event
- RDS Programme Service name and DLS now-playing text shown per stream
- Site grouping — streams organised by connected hub site
- Configurable grid density (compact / standard / large)
- Auto-hiding fullscreen kiosk mode — press F or click the fullscreen icon
- No controls required during normal operation — suitable for unattended display

Chapter 10

Zetta Integration Plugin

The Zetta Integration plugin connects SignalScope to RCS Zetta broadcast automation systems, providing live now-playing data and commercial block detection. Install from **Settings** → **Plugins**. Navigate to **/hub/zetta**. No additional Python packages are required.

Features

- Polls the Zetta SOAP service to show live now-playing data: title, artist, cart number, and category
- Detects commercial/spot blocks in real time with a visual indicator on the Zetta dashboard
- Provides /api/zetta/status JSON endpoint for broadcast chain integration and external systems
- WSDL discovery tool to help locate your Zetta SOAP endpoint URL automatically
- Raw SOAP debug console for testing queries and troubleshooting integration issues

Configuration

Enter your Zetta server URL (usually http://your-zetta-server/ZettaWS.asmx) and credentials in the plugin Settings page. Use the WSDL discovery button if unsure of the exact URL. Test the connection with the Test button before saving.

Chapter 11

Morning Report Plugin

The Morning Report plugin auto-generates a daily engineering briefing covering the previous calendar day. Install from **Settings** → **Plugins**. Hub only — navigate to **/hub/morning_report**. The report is generated at 06:00 by default (configurable in plugin settings).

Report Contents

- **At-a-glance summary:** total faults, number of affected chains, longest single outage duration
- **Per-chain health table:** fault count and total downtime per broadcast chain
- **Hourly fault heatmap:** cyan-tinted grid showing when during the day problems occurred
- **Auto-detected patterns:** fault clustering, above-average fault days, recurring issues, clean streaks
- **Stream quality summary:** level and availability data across all monitored inputs

Storage & Access

Reports are stored locally and remain accessible for past dates. Navigate to Morning Report and select a date from the date picker to view any historical report.

Chapter 12

Signal Path Latency Plugin

The Signal Path Latency plugin tracks broadcast chain processing delay over time, helping engineers detect equipment changes, processing substitutions, or STL latency drift before they cause on-air issues. Install from **Settings** → **Plugins**. Hub only. Navigate to **/hub/latency**.

Features

- Polls all connected sites for comparator delay measurements every 30 seconds
- Stores 90 days of latency history in a local SQLite database
- Computes rolling baselines per comparator pair (14-day rolling average)
- Displays per-comparator SVG sparklines showing the latency trend over the last 24 hours
- Status badges per comparator: Stable / Drifting / Alert
- Configurable alert thresholds per comparator pair — alert when drift exceeds N milliseconds from baseline

Chapter 13

Web SDR Plugin

The Web SDR plugin provides a browser-based software defined radio with waterfall display and live audio demodulation. Install from **Settings** → **Plugins**. Navigate to **/hub/sdr**. Requires at least one RTL-SDR dongle with role **Scanner**.

Features

- Scrolling waterfall display with colour-coded signal intensity (colour = power level)
- Click anywhere on the waterfall to tune to that frequency immediately
- Demodulation modes: WFM (wide FM broadcast), NFM (narrow FM / PMR), AM
- Live audio streamed to the browser using the same relay infrastructure as the FM Scanner
- Frequency readout and signal level indicator
- Persistent frequency presets

Chapter 14

Codec Monitor Plugin

The Codec Monitor plugin provides real-time connection tracking for broadcast contribution codecs. It polls each configured device and fires alerts when a codec drops or reconnects, giving operations teams immediate visibility of STL and remote contribution link status. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. Requires **pysnmp** for SNMP-based devices; HTTP-scraping devices need no additional packages.

Supported Devices

Manufacturer	Models	Protocol
Comrex	ACCESS NX, BRIC-Link II, BRIC-Link III	HTTP status page scraping
Tieline	Gateway, ViA	HTTP status page scraping
Prodys	Quantum ST	HTTP status page scraping
APT / WorldCast	Quantum	SNMP (requires pysnmp)

Configuration

Navigate to **Codec Monitor** in the nav bar and click **Add Codec**. For each device provide:

- **Label:** friendly display name (e.g. 'Studio A Comrex')
- **Host / IP:** IP address or hostname of the device
- **Type:** select from supported device list
- **Poll interval:** how often to query the device (default 30 s)
- **Alert on disconnect:** enable to fire CODEC_FAULT when link drops

Connection States

State	Meaning
Connected	Device is reachable and reports an active connection
Idle	Device is reachable but no active connection (line idle)
Offline	Device is unreachable or HTTP/SNMP request timed out

Alerts

A **CODEC_FAULT** alert fires when a codec transitions from Connected → Idle or Offline. A **CODEC_RECOVERY** alert fires when it returns to Connected. Both are sent to all configured notification channels and appear in the Reports alert history.

Mobile API

The endpoint **GET /api/mobile/codecs/status** returns all monitored devices with their current state, label, host, device type, and ISO-8601 last-seen timestamp. Requires a valid mobile API Bearer token.

Chapter 15

Push Server Plugin

The Push Server plugin turns a SignalScope hub into a centralised push notification server for iOS and Android. It holds APNs and FCM credentials and delivers notifications on behalf of any SignalScope installation that points its Push Server URL here — eliminating the need to configure credentials on every client node. Hub-only. Install from **Settings** → **Plugins** → **Check GitHub for plugins**.

Credentials

Platform	Credential	Notes
iOS (APNs)	.p8 key file	Download from Apple Developer portal. Also requires Team ID, Key ID, Bundle ID.
Android (FCM)	Service account JSON	Download from Firebase Console → Project Settings → Service Accounts.

Upload credentials in **Plugins** → **Push Server** → **Settings**. One-click migration copies existing credentials from the local Settings page to the Push Server.

Connecting Client Nodes

On each client node navigate to **Settings** → **Notifications** → **Push Notifications** and enter the hub's URL as the Push Server URL. The client will send all push notifications via the hub's Push Server endpoint instead of calling APNs/FCM directly.

Delivery Flow

- 1. iOS/Android app registers its device token with the local SignalScope instance
- 2. Client node relays the token registration to the Push Server hub
- 3. When an alert fires on any client, the notification is sent to the Push Server
- 4. Push Server signs the APNs JWT (valid 60 minutes, auto-renewed) and POSTs to APNs or FCM
- 5. Device receives the push notification within seconds

Chapter 16

PTP Clock Plugin

The PTP Clock plugin provides a full-screen GPS-accurate wall clock for broadcast studios. Time is served from the GPS or PTP-disciplined server clock — any browser connected to the server displays the same accurate time regardless of local clock drift. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. No additional packages required.

Display Modes

Mode	Description
Digital	Large HH:MM:SS display with tenths-of-a-second. Shows PTP sync status, offset, and jitter below the clock.
Analog	Studio-style analog clock face with sweep second hand. Suitable for on-air studio display.

Usage

Navigate to **PTP Clock** in the nav bar. Switch between digital and analog modes with the toggle button. Append **?brand=MyStation** to the URL to display a custom branding label below the clock — useful for studio monitor displays.

The clock page is suitable for fullscreen display on a dedicated browser tab or kiosk display. Press F11 (or use the browser's fullscreen mode) for a distraction-free studio clock.

PTP Sync Status

When the host machine is synchronised via PTP (IEEE 1588) or GPS the clock page shows a sync status badge, current offset from the grandmaster clock, and jitter (standard deviation of recent offset samples). A green badge indicates sync within ±1 ms of the grandmaster.

Chapter 17

Icecast Streaming Plugin

The Icecast Streaming plugin re-streams any monitored input to an Icecast2 server, turning SignalScope nodes into live internet radio relay points. It taps the same PCM buffer used by the Logger plugin, so recording and streaming run simultaneously without conflict. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. Requires **ffmpeg** and an Icecast2 server installed separately.

Features

- Re-stream any monitored input type: FM/RTL-SDR, DAB, ALSA, RTP, HTTP
- Per-stream stereo toggle: HTTP inputs preserve native stereo; all others upmix mono to dual-mono
- Hub overview shows live listener counts and stream status from all connected sites
- Create and manage streams on any client node directly from the hub interface
- Multiple streams can share a single Icecast2 server on different mount points

Configuration

Navigate to **Icecast Streaming** in the nav bar and click **Add Stream**. For each stream provide:

- **Input stream:** select from monitored inputs on this node
- **Icecast server URL:** e.g. http://localhost:8000
- **Mount point:** e.g. /coolFM
- **Source password:** the Icecast source password
- **Bitrate:** output MP3 bitrate in kbps (default 128)
- **Stereo:** enable for stereo inputs

Hub Overview

On a hub deployment, the Icecast Streaming page shows a unified overview of all streams across all connected sites. Each stream card shows the mount point, current listener count, bitrate, and Start/Stop controls. Streams can be managed on remote client nodes from the hub without logging into individual nodes.

Chapter 18

Listener Plugin

The Listener plugin provides a polished, consumer-grade audio player for presenters and producers. It shows all streams the authenticated user has access to as station cards with live level meters, one-tap playback, and auto-reconnect. Hub-only. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. No additional packages required.

Features

- Station cards with stream name, site, and live PPM-style level meter
- One-tap audio playback — tap any card to start listening immediately
- Stereo badge on stereo streams; STEREO also shown in the now-playing bar
- Animated equaliser bars while audio is playing
- Volume control slider in the now-playing bar
- Auto-reconnects silently if the stream drops
- Mobile-friendly layout — designed for tablets and phones as well as desktop
- Users only see streams they have permission to access (respects the plugin role user system)

Access

Navigate to **Listener** in the nav bar, or direct non-technical users to the URL directly. Users assigned the **Producer** role are directed to the Producer View on login; Listener is available to all other authenticated users at **/hub/listener**.

Chapter 19

Producer View Plugin

The Producer View plugin provides a simplified, plain-English hub interface for producers and presenters — technical detail is replaced with at-a-glance status and a human-readable fault history. Hub-only. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. No additional packages required.

Features

- Station status cards with green / amber / red status indicator and live level bar
- All-clear banner displayed prominently when everything is running normally
- Plain-English fault history — e.g. 'Audio lost for 4 minutes, recovered at 14:32'
- Audio replay buttons for fault clips — producers can hear exactly what went to air
- Chain-filtered and site-filtered: users only see chains and sites they have permission to view
- Producer-role users are directed here automatically on login

Producer Role

Assign the **Producer** role to a user in **Settings** → **Users**. Producer-role users are redirected to the Producer View on every login — they never see the full engineering dashboard. Requires SignalScope 3.4.85 or later with the plugin-role user system enabled.

Chapter 20

AzuraCast Plugin

The AzuraCast plugin connects SignalScope to AzuraCast web radio installations, providing live now-playing data, listener counts, and station health monitoring with automatic fault alerting. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. No additional packages required.

Features

- Station cards show current track, artist, album art, progress bar, next track, live/AutoDJ status, and listener count
- AZURACAST_FAULT alert fires when a station goes offline or becomes unreachable
- AZURACAST_RECOVERY alert fires when a station comes back online
- AZURACAST_SILENCE alert fires when a station is broadcasting but its linked SignalScope input is silent
- Hub overview aggregates all AzuraCast stations from all connected sites in one page
- Supports multiple AzuraCast servers; each site can manage its own server list

Configuration

Navigate to **AzuraCast** in the nav bar and click **Add Server**. Enter:

- **Server URL:** your AzuraCast installation URL (e.g. https://radio.example.com)
- **API Key:** an AzuraCast read-only API key (generate in AzuraCast → Admin → API Keys)

Once added, stations are discovered automatically via the AzuraCast Now Playing API. To enable silence correlation, link each AzuraCast station to a SignalScope monitored input in the station settings panel — the plugin will cross-reference audio levels with broadcast status.

Chapter 21

Sync Capture Plugin

The Sync Capture plugin performs multi-site simultaneous audio capture across all connected SignalScope nodes. It is ideal for checking simulcast alignment, verifying network contribution quality, or making reference recordings across a transmitter network. Hub-only. Install from **Settings** → **Plugins** → **Check GitHub for plugins**. No additional packages required.

How It Works

- 1. Select any combination of inputs from any connected sites in the Sync Capture page
- 2. Set a capture duration (5–300 seconds)
- 3. Press Capture — the hub broadcasts a timestamped command to all selected sites simultaneously
- 4. Each client grabs the last N seconds from its rolling audio buffer at the agreed wall-clock moment
- 5. Clips are uploaded to the hub and presented together with inline audio players
- 6. Listen to all captures side by side to compare alignment, contribution quality, or identify faults

Analysis Tools

Tool	Description
■ Align	Computes cross-correlation between clips to find the sub-sample timing offset of each site relative to the reference. Displays lag in milliseconds and correlation quality.
EBU R128 LUFS	True peak and integrated loudness per clip for level comparison across sites.
Octave-band spectrum	Per-clip octave-band energy plot to identify frequency-domain differences between sites.
Stereo L/R analysis	Level and phase for left and right channels independently on stereo clips.
RDS/DLS snapshot	Metadata captured at the moment of the clip for FM and DAB inputs.

DAW Session Export

Click ■ **DAW Session** on any completed capture to download a ZIP containing:

- All WAV clips in a **clips/** subdirectory
- A **REAPER .rpp** project file — one track per site, clips positioned at timeline 0
- An **Adobe Audition .sesx** session file — one track per site, clips positioned at timeline 0

If ■ **Align** has been run before export, the alignment offsets are baked into the session files as source in-points — open the project in REAPER or Audition and the clips will be perfectly time-aligned automatically.

BWF Export

Individual clips can be downloaded as Broadcast Wave Format (BWF) files with embedded origination timestamp metadata. The BWF timestamp corresponds to the wall-clock capture time on the originating site, enabling forensic time-stamping of recorded material.

Chapter 22

SignalScope Player

SignalScope Player is a standalone desktop application for Windows and macOS that provides offline access to logger recordings. It connects to a SignalScope hub's Logger API, browses recordings by site and date, and streams audio on demand — without needing a browser.

Connection Modes

Mode	Description
Hub	Connect to a remote SignalScope hub. Enter the hub URL and a mobile API token. The player lists all sites and dates available through the hub's Logger. Audio streams through the hub relay pipeline in real time.
Direct	Point directly at a logger recordings folder on a shared drive or local disk. Audio is read and played directly from the filesystem — no network relay required.

Features

- Site and stream selector — browse all available sites and slugs
- Date picker — navigate to any recorded date
- Day bar — full-day minimap with colour-coded 5-minute blocks; click to jump to any time
- Track band — amber song markers on the day bar showing exact start and end times
- Segment-level playback with transport controls (play, pause, stop)
- Skip controls — ±30 s and ±60 s buttons that cross segment boundaries automatically
- Exact-time seeking — clicking the day bar seeks within the segment, not just to its start
- Volume control and mute
- Now-playing display — stream name, date, and wall-clock position

Hub Mode Connection

- 1. In SignalScope web UI: Settings → Mobile API → Generate token
- 2. Copy the hub URL (e.g. https://hub.example.com) and the API token
- 3. In SignalScope Player: Settings → Mode → Hub, paste URL and token
- 4. Click Connect — the player authenticates and loads available sites
- 5. Select site, stream, and date — recordings list populates automatically
- 6. Click any segment or day bar position to start playback

Direct Mode

In Direct mode the player reads audio files from a local or network path. Point it at the **logger_recordings/** directory or any path structured as **{slug}/{YYYY-MM-DD}/HH-MM.{ext}**. No hub connection or API token

is needed. Sidecar metadata files (**meta_*.json**) are loaded automatically for track bands and show names if present.

Download & Installation

Platform	Download	Notes
Windows	SignalScopePlayer.exe (GitHub Releases)	Double-click to run. No installer needed.
macOS	SignalScopePlayer-macOS.zip (GitHub Releases)	Unzip, drag SignalScopePlayer.app to Applications. Right-click → Open on first launch to bypass Gatekeeper.

Built with Python and PyQt6. Requires no Python installation on the host machine — all dependencies are bundled in the executable.

Chapter 23

Alerting

SignalScope generates alerts for a wide range of signal quality and fault conditions. All alerts are logged to the alert history and, depending on configuration, sent via one or more notification channels.

Level Alert Types

Alert	Condition
SILENCE	Level below configured silence floor for the minimum silence duration
CLIP	Level at or above clip threshold (default -1.0 dBFS)
HISS	High-frequency noise floor above configured threshold
LUFS_TP	True peak exceeds dBTP threshold (default -1.0 dBTP)
LUFS_I	30-second integrated loudness outside EBU R128 target (-23 LUFS ±3 LU)
GLITCH	Brief dropout — onset rate, recovery rate, and dip depth all evaluated to distinguish fades from genuine glitches

Composite Fault Alerts

When silence is detected on an FM, DAB, or RTP source, SignalScope automatically diagnoses the likely fault location using carrier presence, RDS lock, and packet loss data:

Alert	Source	Meaning
STUDIO_FAULT	FM	Silence + carrier present + RDS lock present → fault upstream of transmitter
STL_FAULT	FM	Silence + carrier present + RDS absent → STL or studio-to-transmitter link fault
TX_DOWN	FM	Silence + weak/absent carrier + no RDS → transmitter or antenna fault
DAB_AUDIO_FAULT	DAB	Silence + mux locked + SNR ≥ 5 dB → audio fault within the DAB service
DAB_SERVICE_MISSING	DAB	Ensemble locked but service absent from multiplex
RTP_FAULT	RTP	Silence + ≥10% packet loss → Livewire/AES67 network fault

Metadata Mismatch Alerts

Alert	Condition
FM_RDS_MISMATCH	Received RDS PS name differs from the expected (pinned) name
DAB_SERVICE_MISMATCH	Received DAB service name differs from expected

AI & Chain Alerts

Alert	Condition
AI_ANOMALY	Autoencoder reconstruction error exceeds trained baseline
CMP_ALERT	Post-processing stream is silent while pre-processing stream has audio
CHAIN_FAULT	First down node identified in a broadcast chain
CHAIN_RECOVERED	Broadcast chain returns to fully OK state
CHAIN_FLAPPING	Chain has faulted and recovered 3 or more times within 10 minutes

Codec Alerts

Alert	Condition
CODEC_FAULT	A monitored contribution codec (Comrex, Tieline, Prodys, APT) transitions from connected to idle or offline
CODEC_RECOVERY	A previously faulted codec reconnects and returns to connected state

Setting Expected Names

Click ■ **Set** on any FM card to pin the current RDS PS name as the expected name. ✓ = match, ■ = mismatch. Click ■ **Update** to re-pin if the station rebrands. The same mechanism applies to DAB service names.

Notification Channels

Configure via **Settings** → **Notifications**. A test button is available for each channel.

Channel	Notes
Email (SMTP)	Standard SMTP with TLS. Supports authenticated and unauthenticated relays.
MS Teams	Adaptive Card format with colour-coded severity, or plain text incoming webhook.
Pushover	Mobile push notifications with priority levels (normal, high, emergency).
Webhook	Generic HTTP POST with JSON payload — integrate with any third-party system.

Alert Cooldown & Escalation

A 60-second cooldown prevents duplicate notifications per alert type per stream. The alert history is always written regardless of cooldown. Per-stream escalation timeout (minutes): if an alert is unacknowledged after N minutes, all channels re-fire. Set to 0 to disable escalation.

Chapter 24

Broadcast Chains

Broadcast Chains model the physical signal path as an ordered sequence of monitoring points. The hub identifies the first failed node and fires a named fault alert with specific fault location information, enabling rapid fault diagnosis.

Creating a Chain

- 1. Hub → Broadcast Chains → + New Chain
- 2. Enter a chain name (e.g. 'Cool FM Distribution')
- 3. Click + Add Node for each monitoring point in source-to-destination order
- 4. For each node: select Site (this node or any connected remote site), select Stream, optionally add a Label and Machine tag
- 5. Click ■ Save Chain

Node Stacking

Place multiple streams at the same position in a chain for parallel monitoring:

- **Fault if ALL silent:** all streams must fail before this position is considered faulted. Use for redundant receivers.
- **ANY down = fault:** any single stream failing triggers a fault at this position. Use when every path is required.

Ad Break Handling

Ad break handling suppresses false fault alerts during commercial breaks. Two modes:

- **With mix-in point:** mark one node as the ad mix-in point (where the ad server feeds in). While that node carries audio, upstream silence is held for the configured fault confirmation delay before alerting.
- **Without mix-in point:** when a fault-if-ALL-silent stack goes silent but a downstream node has audio, SignalScope detects an ad break automatically. Shows AD BREAK (amber) instead of FAULT (red). If audio returns within the window, no alert is sent and no SLA downtime is recorded.

Maintenance Bypass

Mark any node as **In Maintenance** to exclude it from fault detection for a set duration. A maintenance badge is shown on the chain view and the node is skipped during evaluation. Chain SLA is not affected during maintenance windows.

Chain Health Score (0–100)

Component	Weight
30-day SLA uptime	0–70 points
Fault frequency (last 7 days)	0–20 points
Stability (flapping penalty)	0–10 points
Trending-down nodes	-5 per node (maximum -15)
RTP packet loss	0 to -10 points

Score	Label
≥ 90	Healthy
75–89	Watch
50–74	Degraded
< 50	Poor

Fault History & Audio Replay

Each chain logs the last 50 fault and recovery events. At fault time, audio clips are saved for every node in the chain. Click ■ **Replay** to open the inline replay timeline:

- Clips are colour-coded by signal-path position
- Fault point = red, last-good node = green, recovery positions = amber/cyan/purple
- ■ Play All plays clips sequentially with the active node highlighted

Historical Chain View

Click ■ **View History** to reconstruct the chain's appearance at any past date and time using stored metric history. Useful for post-incident review and SLA reporting.

A/B Group Monitoring

Hub → Broadcast Chains → **+ New A/B Group**. Tracks two chains as A (active) and B (standby). Raises an alert if the active chain faults while the standby chain is also degraded.

Chapter 25

Hub Mode

Hub mode enables multi-site aggregation. A central hub node collects data from all connected client nodes, providing a unified view of the entire broadcast estate.

Setup

Hub server: Settings → Hub → Enable hub mode. Set a hub secret key.

Clients: Settings → Hub → enter the hub URL and the same secret key.

Site Approval

New client connections appear in the hub's **Pending Approval** queue. The hub admin approves each site explicitly. Approved sites persist indefinitely and are not removed for going offline.

Remote Management

From the hub dashboard you can, without logging into individual nodes:

- Start and stop monitoring on any client
- Add and remove input sources (including DAB bulk-add)
- View hub reports covering all sites
- Browse Logger recordings from any connected site

Hub Notification Delegation

Configure each client to suppress its own notifications and delegate to the hub. The hub applies per-site forwarding rules and deduplication by event UUID, preventing duplicate alerts when both hub and client notification channels are configured.

Wall Mode

Navigate to **/hub?wall=1** or click ■ **Wall Mode**. Wall mode provides a heads-up display optimised for a large screen:

- Header bar: live clock, summary pills (alert/warn/offline counts), exit button
- Connected Sites strip: one pill per site with status dot and alert count
- Broadcast Chains panel: each chain as a card with signal path left-to-right. Stacks show compact rows (status dot · name · level bar · dB). Positions labelled P1, P2, P3...
- Stream Status grid: one card per stream across all sites

Wall mode auto-reloads every 60 seconds.

Hub Architecture & Security

Each client monitors local RF/IP sources and reports via HMAC-signed, AES-256-GCM encrypted heartbeats every ~10 seconds. In addition, clients push slim live metric frames (level, peak, silence state) to the hub at **5 Hz** so that level bars and broadcast chain evaluation update in sub-second time. The hub issues commands back on heartbeat ACKs (listen_requests, commands). Clients cannot be directly called by the hub (NAT traversal is not required — clients always initiate outbound connections to the hub).

Chapter 26

AI Anomaly Detection

Each stream has its own ONNX autoencoder model trained on 14 audio features. The AI engine learns the normal behaviour of each stream and alerts when the signal deviates significantly from the learned pattern.

How It Works

Phase	Duration	Behaviour
Learning	First 24 hours from stream creation	Model trains on incoming feature vectors. No anomaly alerts are generated during this phase.
Detection	After training completes	Reconstruction error compared to learned baseline. 3 consecutive anomalous windows trigger AI_ALERT or AI_WARN.
Adaptive baseline	Ongoing	Model tracks slow long-term changes via exponential moving average. Gradual changes (e.g. seasonal noise floor shift) do not trigger re-alerts.

Feedback-Driven Retraining

In the Reports page, provide feedback on AI anomaly events:

- ■ **(false alarm):** marks the event as normal behaviour
- ■ **(confirmed fault):** marks as a genuine anomaly

5 false-alarm labels trigger automatic retraining using the full original 24-hour corpus plus all corrected samples. This allows the model to adjust to known-good patterns that were initially treated as anomalies.

Chapter 27

Stream Comparator

The Stream Comparator pairs two streams to measure processing delay and detect signal faults caused by processing chain issues. Configure in **Settings** → **Comparators**.

Features

- Cross-correlates PRE and POST streams to measure processing delay in milliseconds
- CMP_ALERT fires when the post-processing stream goes silent while the pre-processing stream has audio
- Gain drift alerts when the level difference between PRE and POST exceeds the configured threshold
- Delay measurements are available as metrics in Signal History and the Latency plugin

Chapter 28

Metric History & Analytics

Stored Metrics

SignalScope writes metrics to **metrics_history.db** once per minute, retained for 90 days.

Metric	Source	Description
level_dbfs	All	Audio level dBFS
lufs_m, lufs_s, lufs_i	All	LUFS Momentary / Short-term / Integrated (EBU R128)
silence_flag	All	1.0 = currently silent
clip_count	All	Clipping events per snapshot
fm_signal_dbm	FM	RF carrier strength dBm
fm_snr_db	FM	Signal-to-noise ratio dB
fm_stereo	FM	1.0 = stereo pilot present
fm_rds_ok	FM	1.0 = RDS lock confirmed
dab_snr	DAB	DAB signal SNR dB
dab_ok	DAB	1.0 = service present in ensemble
dab_sig	DAB	DAB signal level dBm
dab_bitrate	DAB	Service bitrate kbps
rtp_loss_pct	RTP	Packet loss percentage
rtp_jitter_ms	RTP	Jitter milliseconds (RFC 3550 EWMA)
ptp_offset_us	PTP	PTP clock offset microseconds
ptp_jitter_us	PTP	PTP jitter microseconds
chain_status	Chains	1.0 = OK, 0.0 = faulted
health_pct	Hub	Heartbeat success rate percentage
latency_ms	Hub	Round-trip latency milliseconds

Signal History Charts

Click ■ **Signal History** on any stream card. Select the time range (1h / 6h / 24h) and the metric to display. Charts are rendered inline using SVG.

Availability Timeline

The colour-coded bar below each stream card:

Colour	Meaning
Green	Signal present
Red	Silence / audio floor
Amber	DAB service missing
Dark	No data recorded for this period

Click the bar to cycle between 24h, 6h, and 1h views.

Trend Analysis

SignalScope maintains a 14-day rolling hour-of-day baseline and a 28-day rolling day-of-week baseline for each stream. When the current level deviates more than ±1.5σ from the expected baseline, a trend badge is shown on the stream card. The badge escalates from amber to red after 10 or more minutes of sustained deviation.

Chapter 29

SLA Tracking

SignalScope calculates monthly per-stream uptime percentages, providing SLA reporting for broadcast monitoring obligations.

How SLA Is Calculated

SLA is calculated as the percentage of time a stream was not in a silence/fault state, expressed as a monthly figure. The following periods are **excluded** from SLA downtime calculations:

- Ad break countdown windows (when ad break detection is active)
- Maintenance bypass periods

Accessing SLA Data

- Dashboard: availability percentage shown on each stream card
- Hub Reports: SLA summary table covering all sites and streams
- Raw data: stored in **sla_data.json** in the application directory

Chapter 30

Security

Feature	Details
Authentication	PBKDF2-SHA256 password hashing with random salt. Session timeouts. Login rate limiting.
CSRF Protection	All state-changing routes (POST, PUT, DELETE) require a valid CSRF token in the X-CSRFToken header.
Hub Communications	HMAC-SHA256 request signing, AES-256-GCM payload encryption, 30-second replay window, 60 RPM rate limit per client site.
Path Traversal Prevention	All file-serving routes validate paths against the application directory before serving any file.
SDR API	DAB channel parameter validated against an allowlist of valid Band III channels. PPM offset validated as a signed integer within ±1000.
Plugin Install	Install URL must originate from the official GitHub repository. Downloaded file must contain SIGNALSCOPE_PLUGIN string before being written to disk.
Content Security Policy	Strict CSP applied to all pages. Inline scripts use nonces generated per-request. Dynamic event handlers use data-* attributes and delegated listeners.

Chapter 31

Backup & Migration

Use **Settings** → **Maintenance** → **Backup & Restore** to download a timestamped ZIP containing the complete application state.

Backup Contents

File	Contents
lwai_config.json	All configuration including stream settings, notification channels, hub config, comparators, chains
ai_models/	ONNX autoencoder models, learned baselines, feedback state, 24-hour training corpora
metrics_history.db	SQLite signal history database (90 days of metrics)
sla_data.json	SLA uptime records
alert_log.json	Full alert event history
hub_state.json	Hub site registrations and approval state

Restore & Migration

- **Restore:** Settings → Maintenance → Restore from Backup. Upload the ZIP file.
- **Migrate to new machine:** Install SignalScope on the new machine, then restore from backup. All streams, models, and history are transferred.

Chapter 32

Mobile API

All Mobile API endpoints require authentication via a Bearer token, X-API-Key header, or ?token= query parameter. Generate tokens in **Settings** → **Mobile API**.

Key Endpoints

Endpoint	Method	Description
/api/mobile/status	GET	All streams with live level, LUFS, RDS, DAB, and alert status
/api/mobile/faults	GET	Active fault chains with location and duration
/api/mobile/reports/events	GET	Alert history with pagination
/api/mobile/metrics/history	GET	Time-series metric data for charting
/api/mobile/hub/overview	GET	Hub sites summary — status, alert counts, heartbeat age
/api/mobile/register_token	POST	Register an APNs or FCM push notification token
/api/mobile/logger/sites	GET	List sites with logger recordings available on the hub
/api/mobile/logger/dates	GET	List recorded dates for a given site and stream slug
/api/mobile/logger/segments	GET	List 5-minute segment metadata for a given date
/api/mobile/logger/prepare_play	POST	Request a hub relay stream URL for a segment (with optional seek_s)
/api/mobile/codecs/status	GET	All monitored codec devices with connection state and last-seen time

iOS App Features

- Dashboard with live stream cards
- Active Faults list with push notification on new fault
- Reports page with alert history and filtering
- Hub Overview across all connected sites
- Signal History with Swift Charts visualisation
- Audio playback via AVPlayer — listen to any stream from the iOS app

Chapter 33

Plugin Development

SignalScope's plugin system allows you to extend the application with custom pages and functionality. Plugins are single Python files placed in the **plugins/** subdirectory that are automatically discovered and loaded at startup.

Note: older releases stored plugins alongside signalscope.py in the root directory. On first run after upgrading, SignalScope automatically migrates any root-level plugin files and their associated config files into the plugins/ subdirectory.

Minimal Plugin Skeleton

```
# plugins/myplugin.py

SIGNALSCOPE_PLUGIN = {
"id": "myplugin", # unique slug, matches filename stem
"label": "My Plugin", # nav bar label
"url": "/hub/myplugin", # nav bar href
"icon": "\U0001f527", # optional emoji
}

def register(app, ctx):
"""Called once at startup. Register Flask routes here."""
login_required = ctx["login_required"]
csrf_protect = ctx["csrf_protect"]
monitor = ctx["monitor"]
hub_server = ctx["hub_server"]
listen_registry = ctx["listen_registry"]
BUILD = ctx["BUILD"]

@app.get("/hub/myplugin")
@login_required
def myplugin_page():
return "<h1>My Plugin</h1>"
```

Context Keys (ctx)

Key	Type	Description
app	Flask	The Flask application instance
monitor	AppMonitor	Access monitor.app_cfg (config dataclass), monitor.log(), etc.
hub_server	HubServer | None	Hub state: _sites, _scanner_sessions, etc. None on client-only nodes.
listen_registry	ListenSlotRegistry	Create and get audio relay slots
login_required	decorator	Apply to routes that require an authenticated browser session
mobile_api_required	decorator	Apply to /api/mobile/... routes — accepts Bearer token auth from the iOS app. Always obtain as: ctx.get("mobile_api_required", ctx["login_required"])
csrf_protect	decorator	Apply to POST routes to validate CSRF token
BUILD	str	Current build string e.g. 'SignalScope-3.5.104'

Hub-Only Plugins

Plugins that only make sense in hub or both mode can set **"hub_only": True** in their SIGNALSCOPE_PLUGIN dict. The nav bar item is suppressed automatically when running in client-only mode.

Adding to the Public Registry

Add an entry to **plugins.json** at the repository root and open a pull request:

```
{
"id": "myplugin",
"name": "My Plugin",
"file": "myplugin.py",
"icon": "\U0001f527",
"description": "What it does.",
"version": "1.0.0",
"requirements": "numpy scipy",
"url": "https://raw.githubusercontent.com/itconor/SignalScope/main/plugins/myplugin.py"
}
```

Users will see the plugin listed in **Settings** → **Plugins** → **Check GitHub for plugins** and can install it with a single click. See the repository's CLAUDE.md for full plugin documentation including audio relay integration, hub-client command patterns, SDR IQ capture, and the browser audio pump JavaScript.

Chapter 34

Troubleshooting

FM Scanner shows no sites

Ensure at least one dongle is configured with role Scanner in Settings → SDR Devices. Wait one heartbeat cycle (~10 seconds) after changing dongle roles for the hub to update.

Logger not recording

- Check ffmpeg is installed: ffmpeg -version
- Ensure the stream is enabled for recording in Logger → Settings
- Check the base recordings directory is writable by the signalscope user
- View recent logs: journalctl -u signalscope -n 100

Logger recording in wrong format

The format setting only affects new 5-minute segments. Existing recordings keep their original format. Both old and new formats will play back correctly in the timeline player.

DAB scanner won't start

Ensure welle-cli is installed and on the PATH: which welle-cli. Verify the RTL-SDR dongle is not already in use by another process (e.g. a monitoring input).

No audio in FM Scanner

- Check the browser allows audio autoplay for the SignalScope domain
- Click anywhere on the page to unlock the Audio Context (required by browsers on first interaction)
- Ensure the Scanner-role dongle is free — it cannot be shared with monitoring inputs

Hub won't accept client connections

- Verify the hub URL (include port if not 443/80): https://your-hub.example.com
- Verify the secret key matches on both hub and client
- Approve the pending site in hub Settings → Hub → Pending Approval
- Check firewall allows inbound TCP on port 5000 (or 443 with NGINX)

NGINX 413 error on audio uploads

SignalScope automatically compresses WAV clips to MP3 before upload to stay within limits. If errors persist, add the following to your nginx server block:

```
client_max_body_size 50M;
```

Plugin not appearing after install

A restart is required after installing or removing a plugin. Go to Settings → Maintenance → Restart SignalScope, or run: sudo systemctl restart signalscope

AI showing 'Learning' for extended period

The learning phase runs for 24 hours of monitoring data from stream creation. If the stream was offline during this period, the clock does not advance. Ensure the stream is connected and receiving audio continuously.

RDS PS name not updating

The RDS reader requires 3 of the last 12 matching PS samples (minimum 6 characters) before updating the displayed name. This stabilisation prevents display glitches from noisy RDS. Stereo, TP, TA, and PI fields update immediately without stabilisation.