11 - RouteTrack Pi — Logger Service Cleanup & Boot Reliability

Date: December 25, 2025
Category: Raspberry Pi / systemd / GPS / Reliability

Backlink: 10 – RouteTrack Pi — Shift Mode (SQLite + Flask API + Dashboard Controls)

Project Goal

Before taking RouteTrack mobile (car/truck use), I wanted to ensure the system behaves like an appliance:

- Power on → services come up automatically
- No manual “status checks” required every boot
- Logging survives reboots and unexpected shutdowns
- Logging is independent of network availability (portable use)

This entry documents the final cleanup to the `routetrack-logger.service` unit so it is:

- Dependency-safe (waits for GPSD)
- Restart-safe (always recovers)
- Logging-clean (journald only, no conflicting directives)

Problem Identified

The logger service originally contained **conflicting output directives**, which can create confusion about where logs are actually going.

Example conflict pattern:

- `StandardOutput=append:/path/to/file.log`
- followed later by `StandardOutput=journal`

In systemd, the **last directive wins**, meaning file append logging may silently stop even though it appears configured.

To keep RouteTrack stable and simple, the logger service was standardized to **journald-only** logging.

Final Logger Service Configuration (Clean + Portable)

View the service

```
sudo systemctl cat routetrack-logger.service
```

Final contents used

```
# /etc/systemd/system/routetrack-logger.service

[Unit]
Description=RouteTrack GPS Logger

# Do not start logger until gpsd is up
After=gpsd-standalone.service network.target
Wants=gpsd-standalone.service
Requires=gpsd-standalone.service

[Service]
Type=simple
```

```
# Run from project root
WorkingDirectory=/opt/routetrack

# Use the Python venv so RouteTrack is isolated from system packages
ExecStart=/opt/routetrack/venv/bin/python /opt/routetrack/bin/routetrack-logger.py

# Restart forever on crash/disconnect
Restart=always
RestartSec=3

# Send stdout/stderr into journald
StandardOutput=journal
StandardError=journal

[Install]
WantedBy=multi-user.target
```

Why This Unit File Is “Truck Safe”

GPS Dependency Enforcement

Because of:

```
Requires=gpsd-standalone.service
After=gpsd-standalone.service
```

systemd will not attempt to start RouteTrack logging until GPSD is up.

Crash Recovery

Because of:

```
Restart=always
RestartSec=3
```

if the logger crashes or the GPS temporarily drops, RouteTrack automatically restarts.

Logging That Doesn’t Break

Because of:

```
StandardOutput=journal
StandardError=journal
```

all logs are always available via systemd journald, which is reliable even with power cycling.

Reload & Restart Procedure

After any systemd file changes:

```
sudo systemctl daemon-reload
sudo systemctl restart routetrack-logger.service
```

Confirm Boot Enablement

The portable goal is to skip status checks entirely, so the services must auto-start on boot.

Verify enablement:

```
systemctl is-enabled gpsd-standalone.service routetrack-logger.service routetrack-dashboard.service
```

If any return `disabled`, enable them:

```
sudo systemctl enable gpsd-standalone.service routetrack-logger.service routetrack-dashboard.service
```

“No Status Checks” Workflow

Once enabled:

- If the dashboard loads from my phone → dashboard service is running

- If I can press **Start Shift** → API is live
- If route points update while driving → GPSD + logger are working

This allows RouteTrack to behave like a real vehicle appliance system.

Quick Troubleshooting (Single Command)

If anything seems off after boot, the first and best check is:

```
journalctl -u routetrack-logger -n 50 --no-pager
```

Result

RouteTrack is now ready for mobile testing:

GPS logger waits for GPSD
Logger restarts automatically if anything fails
Journald logging is clean and consistent
Boot enablement supports “power on and go” use

Next Steps

The next phase is real-world validation:

- Start a shift before leaving
- Drive a short route (gas station test)
- End shift on return
- Run processor and confirm stop events + summaries update

Revision #2
Created 25 December 2025 16:52:44 by Nate Nash
Updated 25 December 2025 21:46:08 by Nate Nash