TeslaMate Install Guide

Self-hosted Tesla data logging & dashboards — cloud or local server

Worked example uses Vultr (cloud hosting) and Porkbun (domain registrar)
any equivalent providers work the same way

Summary

What this document is

A complete, start-to-finish guide to installing TeslaMate — an open-source tool that logs your Tesla's drives, charges, and vehicle data to your own database, with Grafana dashboards to visualize it. This covers everything from choosing where to host it, to a fully working, HTTPS-secured installation.

Who it's for

Anyone comfortable using a command-line text editor (nano or vi) and following step-by-step instructions, but who doesn't necessarily know Linux server administration or cloud hosting platforms in depth. No prior Docker or Vultr experience is assumed.

Why it was created

TeslaMate's own documentation covers the software itself well, but assumes you already have a server and domain set up. This guide fills that gap — including a full walkthrough of setting up cloud hosting from a blank account, since that's often the hardest part for newcomers.

High-level steps in this document

- 1. What You're Building
- 2. Before You Start — Decisions to Make
- 3. Provision a Server (Vultr Walkthrough)
- 4. Secure Shell (SSH) Access & Fail2ban
- 5. OS-Level Configuration
- 6. Install Docker & Docker Compose
- 7. Get a Domain & Point DNS at Your Server
- 8. Create the Project Files
- 9. Deploy the Stack
- 10. First-Time Setup
- 11. Verify Everything Works
- 12. What's Next
- Appendix A: Full docker-compose.yml
- Appendix B: .env Template
- Appendix C: Change Log

1. What You're Building

TeslaMate is a set of small programs (“containers”) that work together, managed by a tool called Docker Compose:

- TeslaMate — talks to Tesla's API, logs every drive, charge, and state change
- Postgres — the database where that history actually lives

- Grafana — turns that data into dashboards and maps you view in a browser
- Mosquitto — a small messaging broker TeslaMate uses internally
- Traefik — a reverse proxy that gives you a real https://yourdomain.com address with a free, auto-renewing SSL certificate, instead of a bare IP address

All five run on one server, communicating with each other privately — only the TeslaMate and Grafana web interfaces are exposed to the internet, and only over HTTPS.

2. Before You Start — Decisions to Make

2.1 Where will it run?

You have two options, and this guide primarily walks through the cloud path since it's the easier starting point for most people:

Option	What it involves
Cloud hosting (this guide's main path)	Rent a small virtual server from a provider like Vultr (~$5–$6/month). The provider handles power, internet connectivity, and physical hardware. This guide walks through Vultr as a concrete example.
Your own local server	Run it on a spare computer, Mac Mini, or similar, on your home network. You take on everything a cloud provider would normally handle — see the callout below.

Hosting locally instead of in the cloud?

Everything from Section 4 onward (SSH, Docker, the docker-compose.yml, DNS, etc.) works the same regardless of where the machine physically sits. What changes is that you take on the jobs a cloud provider normally does for you: (1) a way for the outside internet to reach your server — either a static public IP from your ISP, or a Dynamic DNS service if your IP changes; (2) port forwarding on your home router, pointing external ports 80 and 443 at your server's internal IP; (3) keeping the machine powered on and connected 24/7 — a UPS is a good idea; (4) applying OS security updates yourself (Section 5 covers this and works the same locally); and (5) checking your ISP's terms of service, since some residential plans restrict running always-on servers. Skip Section 3 (Vultr-specific) and start from Section 4 instead.

2.2 A domain name

You'll want a real domain (or subdomain) rather than a bare IP address, both for a valid HTTPS certificate and so the address is easy to remember. Any registrar works — this guide uses Porkbun as a worked example in Section 7, but GoDaddy, Namecheap, Cloudflare, and others all support the same basic DNS records.

2.3 A Tesla account

You'll authorize TeslaMate to read your vehicle's data using your normal Tesla account login during Section 10 — no separate developer account is required.

3. Provision a Server (Vultr Walkthrough)

Skip this section if you're hosting locally — go to Section 4.

1. Go to vultr.com and create an account (email + password, or sign in with Google/GitHub). You'll need to add a payment method before deploying a server.
2. Once logged in, click the "+" / "Deploy" button, then "Deploy New Server."
3. Under Server Type, choose "Cloud Compute — Regular Performance."
4. Choose a location close to you (lower latency, and keeps data in a jurisdiction you're comfortable with).
5. Choose Ubuntu 24.04 LTS x64 as the operating system.
6. Choose a server size. For personal TeslaMate use, the smallest plan (1 vCPU / 1GB RAM / 25GB SSD, roughly $5/month) is enough — this isn't a resource-heavy workload.
7. Under SSH Keys, add your public key now if you already generated one (see Section 4) — otherwise you can skip this and use the emailed root password instead, then switch to key-based login afterward.
8. Give the server a label you'll recognize (e.g. "teslamate") and click Deploy Now.

Note:
Deployment takes 1–2 minutes. Once it shows "Running," click into the server and note its IP address on the Overview tab — you'll need it for DNS (Section 7) and for connecting via SSH (Section 4).

4. Secure Shell (SSH) Access & Fail2ban

SSH is how you'll remotely run commands on your server. Set up key-based login now, before doing anything else — it's more secure than a password and you'll use it for every step from here on. Sections 4.4 and 4.5 then close the door behind you: password logins are switched off entirely, and Fail2ban blocks the automated login attempts that reach any server with port 22 open to the internet. Doing it this way means you never have to restrict SSH to a list of known IP addresses, so the server stays reachable from any network you happen to be on.

4.1 On Windows

1. Download and install PuTTY (which includes PuTTYgen) from putty.org.
2. Open PuTTYgen → click Generate → move your mouse until the key is created.
3. Add a passphrase (recommended) → Save private key (.ppk) somewhere safe → copy the public key text shown at the top of the window.
4. Open PuTTY → enter your server's IP → connect (as user root, using the password from your hosting provider if you didn't add a key at deploy time).
5. Once connected, add your public key so future logins don't need a password (commands below).

4.2 On macOS or Linux

These operating systems have SSH built in — no extra software needed. Open Terminal:

```
ssh-keygen -t ed25519 -C "your_email@example.com"   # press Enter to accept defaults,
add a passphrase if you like
ssh-copy-id root@YOUR_SERVER_IP                      # copies your public key to the
server automatically
```

4.3 Manually adding a key (if ssh-copy-id isn't available, e.g. on Windows)

Log into the server once with your provider's emailed password, then run:

```
mkdir -p ~/.ssh && chmod 700 ~/.ssh
nano ~/.ssh/authorized_keys        # paste your public key text as one line
chmod 600 ~/.ssh/authorized_keys
```

Verify:

Disconnect and reconnect — you should get in using your key (and passphrase, if you set one), not the server's password.

4.4 Turn off password logins

Adding a key does not remove the password — until you switch it off, anyone who guesses the root password is in, and bots guess around the clock. Do this once you have confirmed key-based login works, and not before.

On Ubuntu cloud images the setting usually lives in a drop-in file that overrides the main config, so edit that one rather than /etc/ssh/sshd_config:

```
nano /etc/ssh/sshd_config.d/50-cloud-init.conf
```

Change PasswordAuthentication yes to:

```
PasswordAuthentication no
```

If that file does not exist on your image, make the same change in /etc/ssh/sshd_config instead — and check that nothing in /etc/ssh/sshd_config.d/ sets it back to yes, because a drop-in file wins over the main one.

Check the configuration parses, then restart the daemon:

```
sshd -t
systemctl restart ssh
```

Verify:

Leave your current session open. Open a second connection using your key — it should still work. Then try a connection without pointing at the key: the server should refuse it straight away with "No supported authentication methods available" rather than prompting for a password.

> **Note:**
> Never close your only session while testing this. A mistake here can lock you out of the server completely, and the way back in is your hosting provider’s web console — which is worth finding before you start, not after.

4.5 Install Fail2ban

With passwords off, no bot can guess its way in — but they will keep trying, and every attempt still costs a handshake and a line in the log. Fail2ban watches the authentication log and adds a firewall rule that drops traffic from an address that keeps failing. It uses almost no resources and it keeps the logs readable, which matters more than it sounds: a log full of noise is a log nobody reads.

```
apt update && apt install -y fail2ban
```

Create a local override file. Never edit jail.conf directly — package updates overwrite it:

```
cat > /etc/fail2ban/jail.local << 'EOF'
[DEFAULT]
bantime  = 3d      # how long a banned address stays blocked
```

```
findtime = 30m     # the window failures are counted in
maxretry = 3       # failures allowed before a ban

[sshd]
enabled = true
EOF
```

Reload to activate the rules immediately:

```
fail2ban-client reload
```

Verify:

```
fail2ban-client get sshd bantime   # 259200 — three days, in seconds
fail2ban-client status sshd        # the jail enabled, plus any addresses currently
banned
```

To release an address you banned by accident: fail2ban-client set sshd unbanip <IP>

Which log Fail2ban is reading:

Look for a "Journal matches" line in the status output — on Debian and Ubuntu the packaged sshd jail reads the systemd journal, and that line (_SYSTEMD_UNIT=sshd.service + _COMM=sshd) is how you confirm it. Elsewhere, or from a source install, Fail2ban tails /var/log/auth.log instead; if your image ships without rsyslog that file may not exist and the jail will run happily while counting nothing. The fix in that case is "backend = systemd" under [sshd] in jail.local, then a reload. Either way, a jail that reports zero failures on a server that is plainly being probed is reading the wrong place.

5. OS-Level Configuration

A couple of housekeeping items worth setting correctly from the start.

5.1 Timezone

By default, cloud servers are set to UTC. Setting your local timezone means log timestamps and any scheduled tasks make sense at a glance:

```
timedatectl set-timezone America/New_York   # replace with your own timezone
timedatectl   # verify
```

Find your exact timezone name (e.g. America/Chicago, Europe/London) by running: timedatectl list-timezones

5.2 Automatic Security Updates

Ubuntu can install security patches on its own, including rebooting automatically when a kernel update needs it. This is already installed on most Ubuntu cloud images — confirm and tune it:

```
dpkg -l | grep unattended-upgrades      # confirm it's installed
systemctl list-timers | grep apt        # confirm the timers are active
sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
   Unattended-Upgrade::Automatic-Reboot "true";
   Unattended-Upgrade::Automatic-Reboot-Time "03:00";   # pick a low-traffic time
```

Why this matters: your server will be reachable from the internet, so keeping the OS patched automatically is worth the small chance of an occasional auto-reboot.

TeslaMate Install Guide

6. Install Docker & Docker Compose

Docker is what actually runs TeslaMate's containers. This installs Docker Engine plus the Compose plugin, using Docker's official repository:

```
sudo apt update && sudo apt install -y ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-
compose-plugin
```

Verify:

```
docker --version
docker compose version
```

7. Get a Domain & Point DNS at Your Server

If you don't already own a domain, register one through any registrar — Porkbun, Namecheap, GoDaddy, and Cloudflare are all common, reputable choices. You can use a domain you already own instead of buying a new one; a subdomain like teslamate.yourdomain.com works just as well as a dedicated domain.

7.1 What you need to add

In your registrar's DNS management screen, add two A records, both pointing at your server's IP address from Section 3:

Type	Host	Answer	TTL
A	teslamate	<your server's IP>	600
A	grafana	<your server's IP>	600

("Host" is the subdomain part — adjust to your preference, e.g. "tm" instead of "teslamate." Whatever you choose here must match the FQDN_TM and FQDN_GRAFANA values you'll set in Section 8.)

7.2 Worked example: Porkbun

1. Log into porkbun.com → click the domain → "DNS Records."
2. Click "Add record" → Type: A → Host: teslamate → Answer: your server's IP → Save.
3. Repeat for Host: grafana.

DNS changes typically take a few minutes to an hour to become visible everywhere, though it's often much faster.

> **Note:**
> If you're using a domain that already has other services on it (like email), only add these two new records — don't touch existing MX or TXT records, which are unrelated and shouldn't be modified.

8. Create the Project Files

Everything lives in one folder. This guide uses ~ (your home directory) for simplicity.

8.1 Folders

```
mkdir -p ~/teslamate && cd ~/teslamate
mkdir -p import acme
touch acme/acme.json && chmod 600 acme/acme.json
```

8.2 Basic-auth file (protects the TeslaMate web app with a login prompt)

```
sudo apt install -y apache2-utils
htpasswd -B -c ~/teslamate/.htpasswd yourusername   # -B = bcrypt; you'll be prompted
to set a password
```

Note: -B selects bcrypt. Without it htpasswd writes Apache's older MD5 (apr1) hash — Traefik accepts it, but it stands up poorly to offline cracking, and this file ends up in your backups, which normally leave the server. Also worth knowing: HTTP Basic Auth has no sign-out. Once a browser has cached these credentials it keeps sending them, so the only way to revoke a device's access is to change the password (htpasswd -B ~/teslamate/.htpasswd yourusername) and restart the proxy container.

8.3 Create the Mosquitto config file

Mosquitto (the messaging broker TeslaMate uses internally) needs an explicit config file:

```
mkdir -p ~/teslamate/mosquitto-config
cat > ~/teslamate/mosquitto-config/mosquitto.conf << 'EOF'
listener 1883
allow_anonymous true
persistence true
persistence_location /mosquitto/data/
log_dest stdout
EOF
```

8.4 Create docker-compose.yml

Create ~/teslamate/docker-compose.yml with the content in Appendix A. It defines all five containers described in Section 1 and wires them together.

8.5 Create .env

Create ~/teslamate/.env with the content in Appendix B, filling in your own domain and generated secrets. Two values need generating rather than guessing:

```
openssl rand -base64 24    # use the output for TM_DB_PASS
openssl rand -base64 32    # use the output for ENCRYPTION_KEY
```

Gotcha:

If a generated value happens to contain a "$" character, generate again — "$" has special meaning in Docker Compose and can break startup.

9. Deploy the Stack

```
cd ~/teslamate
docker compose up -d
```

Verify — all 5 containers running:

```
docker ps
```

Verify — valid HTTPS certificate (repeat for each domain):

```
echo | openssl s_client -connect teslamate.yourdomain.com:443 -servername
teslamate.yourdomain.com 2>/dev/null | openssl x509 -noout -issuer -dates
```

Expect “issuer=... Let's Encrypt ...”. If no certificate appears within a minute or two, check the proxy container's logs (docker logs <proxy container name> --tail 50) — the most common cause is DNS not having propagated yet, or port 80/443 not reaching the server (check your cloud provider's firewall settings, or your router's port forwarding if hosting locally).

TeslaMate Install Guide

10. First-Time Setup

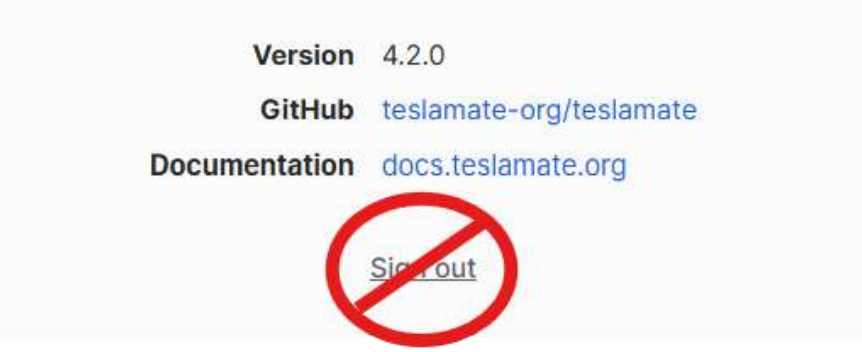
10.1 Connect TeslaMate to your Tesla account

TeslaMate doesn't generate API tokens itself — a small separate tool does this safely on your own computer, without TeslaMate (or anyone else) ever seeing your Tesla password.

1. Go to github.com/adriankumpf/tesla_auth/releases/latest and download the build for your computer (Windows, macOS, or Linux — both Intel and Apple Silicon Macs are supported).
2. Run it — it opens a small window with Tesla's real login page. Log in with your normal Tesla account.
3. It displays an Access Token and a Refresh Token. Copy both into the sign-in screen at https://teslamate.yourdomain.com → click Sign in.

Important:

Once signed in, never use the "Sign out" link at the bottom of TeslaMate's Settings page (https://teslamate.yourdomain.com/settings) — it deletes the stored tokens and stops all data collection until you repeat this section. There's no need to ever sign out under normal use.



Do not click this. It is at the bottom of the Settings page, below the version and documentation links.

10.2 Enter your URLs in Settings

After you've connected your TelsaMate to your Tesla account (see 10.1 above) and logged into http://teslamate.yourdomain.com you'll want to change the two URL pages in the settings page.

1. Log into https://teslamate.yourdomain.com
2. Click on "Settings" on the top banner.
3. Scroll down to the bottom of the page and change the two URL settings.



10.3 Secure Grafana

Grafana ships with a default admin/admin login. Replace it immediately, since Grafana will be reachable from the internet:

4. Log into https://grafana.yourdomain.com with admin / admin.
5. You'll be prompted to set a new password immediately — do so with something strong and unique.
6. Optional extra step: create a second named account for yourself (Administration → Users and access → Users → New user) with Admin role, then disable the default admin account once you've confirmed the new one works.

11. Verify Everything Works

Item	Confirm by
All 5 containers running	docker ps
Valid HTTPS certificates	openssl s_client checks in Section 9
TeslaMate receiving car data	docker logs <teslamate container> — look for “MQTT connection has been established”
Grafana dashboards loading	Log into your grafana domain, open a dashboard
Default Grafana password changed	You were prompted and changed it in Section 10.3
SSH password login refused	Connect without your key — expect "No supported authentication methods available"
Fail2ban SSH jail active	fail2ban-client status sshd — shows the jail enabled and any banned addresses

12. What's Next

Your installation is complete and collecting data. For ongoing care — backups, updates, and maintenance — see the companion document, Supporting TeslaMate. A few other useful resources:

- Official documentation: docs.teslamate.org
- Source code & issue tracker: github.com/teslamate-org/teslamate
- Grafana dashboards update automatically as TeslaMate is updated — no separate action needed for that
- Fail2ban day to day: fail2ban-client status sshd to see what it has blocked, and fail2ban-client set sshd unbanip <IP> to release an address

Appendix A: Full docker-compose.yml

```
services:
  teslamate:
    image: teslamate/teslamate:latest
    restart: always
    depends_on:
      - database
    environment:
      - DATABASE_USER=${TM_DB_USER}
      - DATABASE_PASS=${TM_DB_PASS}
      - DATABASE_NAME=${TM_DB_NAME}
      - DATABASE_HOST=database
      - MQTT_HOST=mosquitto
      - VIRTUAL_HOST=${FQDN_TM}
      - CHECK_ORIGIN=true
      - TZ=${TM_TZ}
      - ENCRYPTION_KEY=${ENCRYPTION_KEY}
    volumes:
      - ./import:/opt/app/import
    labels:
      - 'traefik.enable=true'
      - "traefik.http.services.teslamate.loadbalancer.server.port=4000"
      - "traefik.http.middlewares.redirect.redirectscheme.scheme=https"
      - "traefik.http.middlewares.auth.basicauth.usersfile=/auth/.htpasswd"
      - "traefik.http.routers.teslamate-insecure.rule=Host(`${FQDN_TM}`)"
      - "traefik.http.routers.teslamate-insecure.middlewares=redirect"
      - "traefik.http.routers.teslamate.rule=Host(`${FQDN_TM}`)"
      - "traefik.http.routers.teslamate.middlewares=auth"
      - "traefik.http.routers.teslamate.entrypoints=websecure"
      - "traefik.http.routers.teslamate.tls.certresolver=tmhttpchallenge"

  database:
    image: postgres:16
    restart: always
    environment:
      - POSTGRES_USER=${TM_DB_USER}
      - POSTGRES_PASSWORD=${TM_DB_PASS}
      - POSTGRES_DB=${TM_DB_NAME}
    volumes:
      - teslamate-db:/var/lib/postgresql/data

  grafana:
    image: teslamate/grafana:latest
    restart: always
    environment:
      - DATABASE_USER=${TM_DB_USER}
      - DATABASE_PASS=${TM_DB_PASS}
      - DATABASE_NAME=${TM_DB_NAME}
      - DATABASE_HOST=database
      - GF_SERVER_ROOT_URL=https://${FQDN_GRAFANA}
```

```
    volumes:
      - teslamate-grafana-data:/var/lib/grafana
    labels:
      - 'traefik.enable=true'
      - "traefik.http.services.grafana.loadbalancer.server.port=3000"
      - "traefik.http.middlewares.redirect.redirectscheme.scheme=https"
      - "traefik.http.routers.grafana-insecure.rule=Host(`${FQDN_GRAFANA}`)"
      - "traefik.http.routers.grafana-insecure.middlewares=redirect"
      - "traefik.http.routers.grafana.rule=Host(`${FQDN_GRAFANA}`)"
      - "traefik.http.routers.grafana.entrypoints=websecure"
      - "traefik.http.routers.grafana.tls.certresolver=tmhttpchallenge"

  mosquitto:
    image: eclipse-mosquitto:2
    restart: always
    ports:
      - 127.0.0.1:1883:1883
    volumes:
      - ./mosquitto-config:/mosquitto/config
      - mosquitto-data:/mosquitto/data

  proxy:
    image: traefik:v2
    restart: always
    command:
      - "--global.sendAnonymousUsage=false"
      - "--providers.docker"
      - "--providers.docker.exposedByDefault=false"
      - "--entrypoints.web.address=:80"
      - "--entrypoints.websecure.address=:443"
      - "--certificatesresolvers.tmhttpchallenge.acme.httpchallenge=true"
      - "--certificatesresolvers.tmhttpchallenge.acme.httpchallenge.entrypoint=web"
      - "--certificatesresolvers.tmhttpchallenge.acme.email=${LETSENCRYPT_EMAIL}"
      - "--certificatesresolvers.tmhttpchallenge.acme.storage=/etc/acme/acme.json"
    ports:
      - 80:80
      - 443:443
    volumes:
      - ./.htpasswd:/auth/.htpasswd
      - ./acme/:/etc/acme/
      - /var/run/docker.sock:/var/run/docker.sock:ro

volumes:
  teslamate-db:
  teslamate-grafana-data:
  mosquitto-data:
  acme:
```

Appendix B: .env Template

```
TM_DB_USER=teslamate
TM_DB_PASS=<generate: openssl rand -base64 24>
TM_DB_NAME=teslamate
FQDN_GRAFANA=grafana.yourdomain.com
FQDN_TM=teslamate.yourdomain.com
TM_TZ=America/New_York
LETSENCRYPT_EMAIL=you@yourdomain.com
ENCRYPTION_KEY=<generate: openssl rand -base64 32>
```

Note: Grafana's admin credentials are set through its first-login prompt (Section 10.2), not through this file — that's why GRAFANA_USER/GRAFANA_PW don't appear here.

Appendix C: Change Log

Tracks revisions to this document. The Reference # of the most recent entry appears in the footer of every page.

Ref #	Date	What changed
REV-001	August 25, 2026	Initial version.
REV-002	September 4, 2026	Section 8.2: the htpasswd command now uses -B (bcrypt), with a note explaining why the default MD5 (apr1) hash is a poor choice for a file that ends up in off-server backups, and that HTTP Basic Auth has no sign-out — the only way to revoke a device is to change the password and restart the proxy.
REV-003	September 5, 2026	Section 10.1: the Important callout now names where the Sign out link actually is — the bottom of the Settings page at https://teslamate.yourdomain.com/settings — so a first-time reader can recognise it rather than having to hunt for it. Added a screenshot below the callout showing the link crossed out. Page footers switched from "Created" to "Last edited" per the convention adopted this date; the original creation date remains as the REV-001 row below. Section 10.2: Added 10.2 Enter your URLs in Settings
REV-004	September 11, 2026	Section 4 became "Secure Shell (SSH) Access & Fail2ban". New 4.4 turns password logins off in /etc/ssh/sshd_config.d/50-cloud-init.conf (with /etc/ssh/sshd_config named as the fallback for images without the drop-in), covers sshd -t and the daemon restart, and adds a two-window verification plus a lockout warning. New 4.5 installs Fail2ban with a jail.local of bantime 3d, findtime 30m and maxretry 3, the reload and status checks, the unbanip command, and a note on the systemd backend for images without rsyslog. Section 4 intro rewritten to say why this is done instead of restricting SSH to known IP addresses. Section 11: two verification rows added (password login refused, Fail2ban jail active), and the "Default Grafana password changed" row now points at Section 10.3 rather than 10.2, which stopped being the Grafana step when 10.2 was inserted at REV-003. Section 12: a Fail2ban status and unban pointer. Sourced from TeslaMate_Build_Guide.docx REV-006. Corrected the same day, still REV-004: the 4.5 callout originally led with /var/log/auth.log, but on Debian and Ubuntu the packaged sshd jail already reads the systemd journal, so it now leads with the "Journal matches" check.