Docker Environment

Overview

All services on Centerpoint run as Docker containers, managed through Docker Compose project files and monitored via Portainer EE. Traefik v3 serves as the reverse proxy and TLS termination point for every service.

Docker Engine

Property	Value
Docker Version	29.5.3
Docker Compose	v5.1.4
Storage Driver	`overlayfs`
Docker Root	`/var/lib/docker`
Total Containers	112
Running	97
Stopped	15
Images	108

Project Structure

All compose projects live under `/home/jeeves/docker/`, with one subdirectory per logical stack:

```
/home/jeeves/docker/
├── adguard/
├── ai-stack/         ← Ollama, Open Web UI, Faster-Whisper, Kokoro
├── arr/              ← Sonarr, Radarr, Prowlarr, Bazarr, NZBGet, etc.
├── authentik/
├── bookstack/
```

```
├── crowdsec/
├── homepage/
├── immich/
├── paperless/
├── traefik/
└── ...               (one directory per stack)
```

Each directory contains a `docker-compose.yml` and any local config files or bind-mount targets specific to that stack.

Container Management — Portainer EE

Portainer Enterprise Edition provides the web UI for container lifecycle management, log viewing, stack deployment, and environment monitoring.

Portainer connects to the Docker daemon via a `dockerproxy` sidecar container (Tecnativa Docker Socket Proxy) rather than mounting the Docker socket directly. This limits the API surface exposed to Portainer and reduces the blast radius of any container compromise.

Traefik v3 — Reverse Proxy

Traefik is the single ingress point for all named HTTP/HTTPS traffic. It runs permanently on `traefik-net` and discovers routes automatically from Docker container labels — no manual reload required when stacks are added or removed.

Property	Value
HTTP port	`80` — auto-redirects all traffic to HTTPS
HTTPS port	`443`
Dashboard port	`8080` — internal only (`traefik.home.local`)
Static config	`/home/jeeves/docker/traefik/traefik.yml`
Dynamic config	`/home/jeeves/docker/traefik/dynamic.yml` (file-watched)
Access logs	`/var/log/traefik/access.log` (JSON, buffered)

Certificate Resolvers

Resolver	Scope	Method	Certificate Authority
`letsencrypt`	External domains	HTTP challenge	Let's Encrypt
`step-ca`	`*.home.local`	ACME	Internal Step-CA (`ca.home.local`)

Internal certificates have a 720-hour (30-day) duration and auto-renew via Traefik's built-in ACME client against the Step-CA instance.

Active Plugins

Plugin	Version	Purpose
`PascalMinder/geoblock`	v0.3.6	Country-level block on external-facing routes
`maxlerebourg/crowdsec-bouncer-traefik-plugin`	v1.3.0	Blocks IPs flagged by the local CrowdSec LAPI

Standard Routing Pattern

Each service defines Traefik labels in its own `docker-compose.yml`. The typical pattern for a dual-route service (internal + external) is:

```
labels:
  - "traefik.enable=true"

  # External route — Let's Encrypt TLS + security middleware
  - "traefik.http.routers.<name>-ext.rule=Host(`<svc>.jeeves5454.ddns.net`)"
  - "traefik.http.routers.<name>-ext.entrypoints=websecure"
  - "traefik.http.routers.<name>-ext.tls.certresolver=letsencrypt"
  - "traefik.http.routers.<name>-ext.middlewares=authentik-auth@docker,plex-geoblock@file,crowdsec-bouncer@file"

  # Internal route — Step-CA TLS, no extra middleware
  - "traefik.http.routers.<name>-int.rule=Host(`<svc>.home.local`)"
  - "traefik.http.routers.<name>-int.entrypoints=websecure"
  - "traefik.http.routers.<name>-int.tls.certresolver=step-ca"

  # Backend service port
  - "traefik.http.services.<name>-svc.loadbalancer.server.port=<port>"
```

Services that are internal-only omit the `-ext` router entirely. Services that require OAuth2 authentication on external routes add `authentik@file` middleware.

Notes / Gotchas

- `traefik-net` is an **externally created** network and must exist before any Traefik-fronted stack is started:

  ```
  docker network create traefik-net
  ```

- Never restart `dockerproxy` while Portainer is actively being used — it will lose its Docker connection until the proxy is back up.
- Compose files use the stack subdirectory as their working directory — relative bind mount paths resolve from there.
- `overlayfs` can accumulate orphaned image layers over time. Prune periodically:

  ```
  docker image prune
  docker volume prune   # caution — only remove truly unused volumes
  ```

Last Updated: 2026-06-16

Revision #2
Created 2026-06-16 20:36:33 UTC by Admin
Updated 2026-06-16 20:40:54 UTC by Admin