Portainer Product Strategy

September 2026

Portainer 3.0 represents the start of a multi-release evolution of the product, leading up to our next LTS release. This document lays out the pieces, what each one does for you, and where they land. Timing reflects our current plan. The next Long Term Support release in December 2026 is the first point where most of it comes together.

One idea underneath everything: every change is a Git commit

Modern operations team expect Git to be the source of truth. Any state that lives only inside a management tool is drift waiting to happen. Portainer 3.0 is built so that everything you do through Portainer becomes a Git commit, and environment state is reconciled out of Git rather than applied directly.

What this looks like in the product:

- **GitOps Workflows and GitOps Sources** move to the front of Portainer BE. Define an application once, connect a repository once, and deliver to many environments with the rollout visible from one place.
- **A rebuilt reconciliation engine** based on the open source Flux project replaces our original git reconciliation implementation. This means more reliable reconciliation, richer status, and behaviour Kubernetes teams already understand. However, this reconciliation engine is only for Kubernetes. Docker environments can still use our classic gitops implementation.
- **Fleet Governance Policies become Git-aware.** Today a policy lives in Portainer Server. We plan for the policy UI to write a commit to your repository, and the policy content reconciles to your clusters from there. Your governance posture lives in the same place as your applications.
- **We launch new add-on portals that are GitOps first.** Portainer-Run, Portainer-IDP, and Portainer-Command are new add-on portals that application owners and platform owners can use in the browser, just like Portainer BE. But behind the scenes these tools are git clients that write a git commit on the users behalf, then Portainer Server reconciles that commit with cluster state.

The end goal here is that you get a complete, reviewable, rollback capable record of what changed, who or what changed it, and when. If something goes wrong, you can roll back from Git history.

Modernising from Docker to Kubernetes, at your pace

Docker remains fully supported. The Portainer BE interface you use today continues as the home for Docker Standalone, Swarm, and Podman, and over time it becomes an optional add-on called Portainer-Classic in the app switcher. Nothing is removed and nobody is forced to move.

For those who want to modernise, three new pieces work together as one package:

- **KubeSolo** is single-node Kubernetes that fits where Docker fits: a small business server, an industrial or IoT device, any place that does not need horizontal scaling. You get the power and security of Kubernetes in a footprint that is easier to understand and cheaper to run.
- **Portainer-Migrate** is a new add-on portal that discovers the Docker workloads on environments already registered in Portainer and translates them into Kubernetes workloads on Kubernetes environments registered in Portainer.
- **D2K** is a Docker-to-Kubernetes API translator. Run a KubeSolo node, talk to it with the Docker and Docker Swarm API you already use, and Kubernetes does the work underneath. Existing scripts and developer processes keep working while the backend modernises.

Put together: you can keep your existing footprint, launch KubeSolo on it, move workloads with Portainer-Migrate, and use D2K to bridge the parts of your process you are not ready to change yet. We are seeing a lot of CVEs against Portainer that trace back to Docker CE itself, which we cannot patch. Our Docker-compatible Kubesolo environment does not have that exposure. You interact with it like Docker. It runs like Kubernetes, with all the underlying security patches and maintenance of Kubernetes.

Add-on portals, each built for one job

Rather than one interface trying to serve every kind of user, Portainer 3.0 delivers focused portals on a shared platform. Each is an add-on you enable, and each inherits the same identity, access control, policy, and audit from the Portainer platform underneath.

Available now

- **Portainer-Run** gets small applications up and running fast on your own infrastructure, including apps built with AI coding tools. Drag and drop, no

Kubernetes knowledge required. Available inside Portainer and via a standalone installer.

Targeted for the December 2026 LTS, with early beta tests available to select customers

- **Portainer-IDP** is a GitOps portal for platform owners and application teams. Deploy and manage applications through an interface that records every intended change in Git.
- **Portainer-Command** is governed infrastructure access for AI agents (see below). It also helps you adopt GitOps: it discovers existing workloads in a cluster and proposes declarative definitions for them, so a cluster that was set up by hand can be brought under Git without rewriting everything manually.
- **Portainer-Migrate**, described above.

Following

- **Portainer-Admin** takes over platform setup, identity, access control, and policy.
- **Portainer-Ops** brings fleet visibility, workload health, troubleshooting, and deployment status into one operations view.

Safe infrastructure access for AI agents

More and more work is being done by AI agents acting on someone's behalf. Handing an agent direct access to infrastructure is risky, especially for edge, IoT, and industrial devices. Portainer already sits between users and your fleet through its edge agent and secure proxy, with no inbound ports to expose. In 3.0 that position becomes the way you let AI in safely.

Portainer-Command is an MCP server for Kubernetes that works like an access vending machine. An agent gets temporary read-only access to your fleet. Any change AI wants to make goes into a Git pull request, through human review, and only then reconciles onto the fleet. If something goes wrong, you roll back from Git history. You can see what the agent did, control what it can do, and undo what it got wrong.

Alongside this, our longer term plan to add **state snapshots** before and after changes, so that if a human approves an AI-proposed change that damages stateful data, the data can still be recovered.

AI workloads on infrastructure you own

The same platform that runs your applications can run your AI workloads, on your hardware, without you operating Kubernetes by hand.

- **Portainer-AiGrid** distributes AI workloads across fleets of corporate infrastructure. You focus on the use case, such as dropping in a folder of PDFs and getting back a vector database ready for RAG workloads. Portainer handles the Kubernetes and the workload distribution.
- **KubeSolo** is the runtime for AI at the edge. Docker and Podman are poor platforms for AI workloads; Kubernetes is much better but heavy. KubeSolo gives industrial and IoT customers a light orchestrator for AI workloads on the devices they already have.

Devices and industrial fleets

Portainer's edge agent and GitOps reconciliation already deliver software to remote devices at scale. The **Portainer Industrial App Portal** is our experimental application distribution portal for factory and IoT environments, letting operators deploy approved applications

across a plant hierarchy without touching containers. We are eager to hear more about your industry specific requirements, so we can help build the best IoT application delivery platform for your needs.

Our Next LTS and Beyond

By our next December 2026 LTS, we aim to have Portainer-Run, Portainer-IDP, Portainer-Command, and Portainer-Migrate available as add-ons, GitOps Workflows and Sources front and centre, and KubeSolo and D2K solid as the recommended path from Docker.

Over the longer term, Portainer remains the platform businesses use to run and govern software on infrastructure they control. You adopt Portainer to solve your specific problem: modernising a Docker estate, launching an internal tool, governing a fleet, running AI at the edge. No matter which entry point you start from, it’s all powered by the same platform, with shared identity, policy, delivery, and visibility, and each new Portainer addon product you choose to adopt reuses the infrastructure and controls you already have.

Got questions? Get in touch; we’re happy to help

If you would like to talk to our team about what migrating from Docker to Kubernetes might look like for your business, please get in touch with us here. If you’d like to talk to a technical salesperson or get a demo of Portainer and the new functionality, reach out.