KubeCon | CloudNativeCon

North America 2024

GitOps at Production Scale with Flux

Leigh Capili
Senior DevRel Engineer, Flox

Priyanka “Pinky” Ravi
Platform Tech Advocate, G-Research

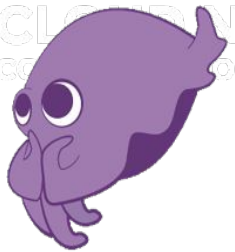
What is Flux?

- A foundation for Continuous and Progressive Delivery solutions for Kubernetes
- A flexible toolkit for you to build your own GitOps platform

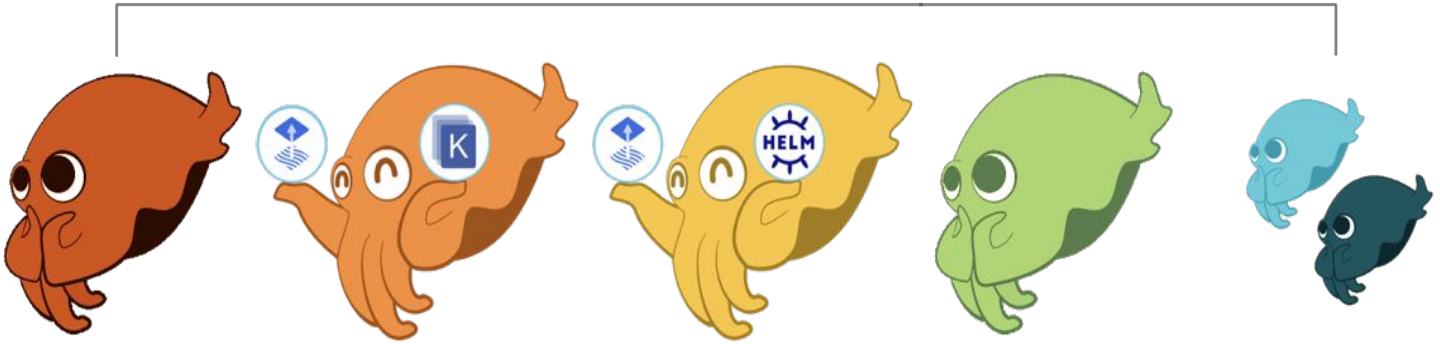
Benefits of Flux

- Reduces developer burden
- Extensible
- Comes with out of the box support for Kustomize and Helm
- Designed For Kubernetes
- Security at its core

Overview of Flux



Flux is a set of
Kubernetes Controllers



Source controller

Kustomize controller

Helm Controller

Notification Controller

Image Reflector & Automation Controller

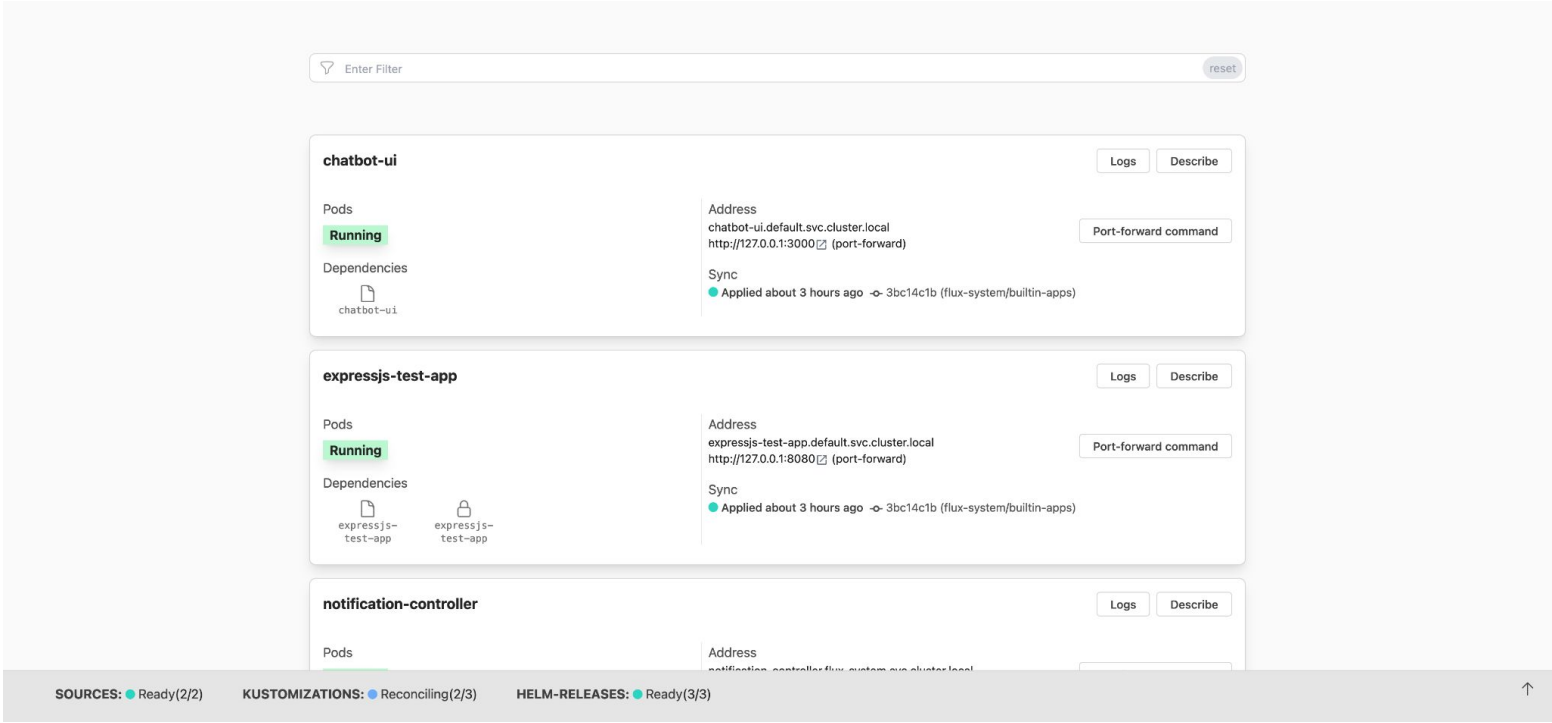
Flux is easy to manage

```
❯ flux bootstrap
The bootstrap sub-commands push the Flux manifests to a Git repository
and deploy Flux on the cluster.

Usage:
  flux bootstrap [command]

Available Commands:
  bitbucket-server Deploy Flux on a cluster connected to a Bitbucket Server repository
  git              Deploy Flux on a cluster connected to a Git repository
  gitea            Deploy Flux on a cluster connected to a Gitea repository
  github           Deploy Flux on a cluster connected to a GitHub repository
  gitlab           Deploy Flux on a cluster connected to a GitLab repository
```

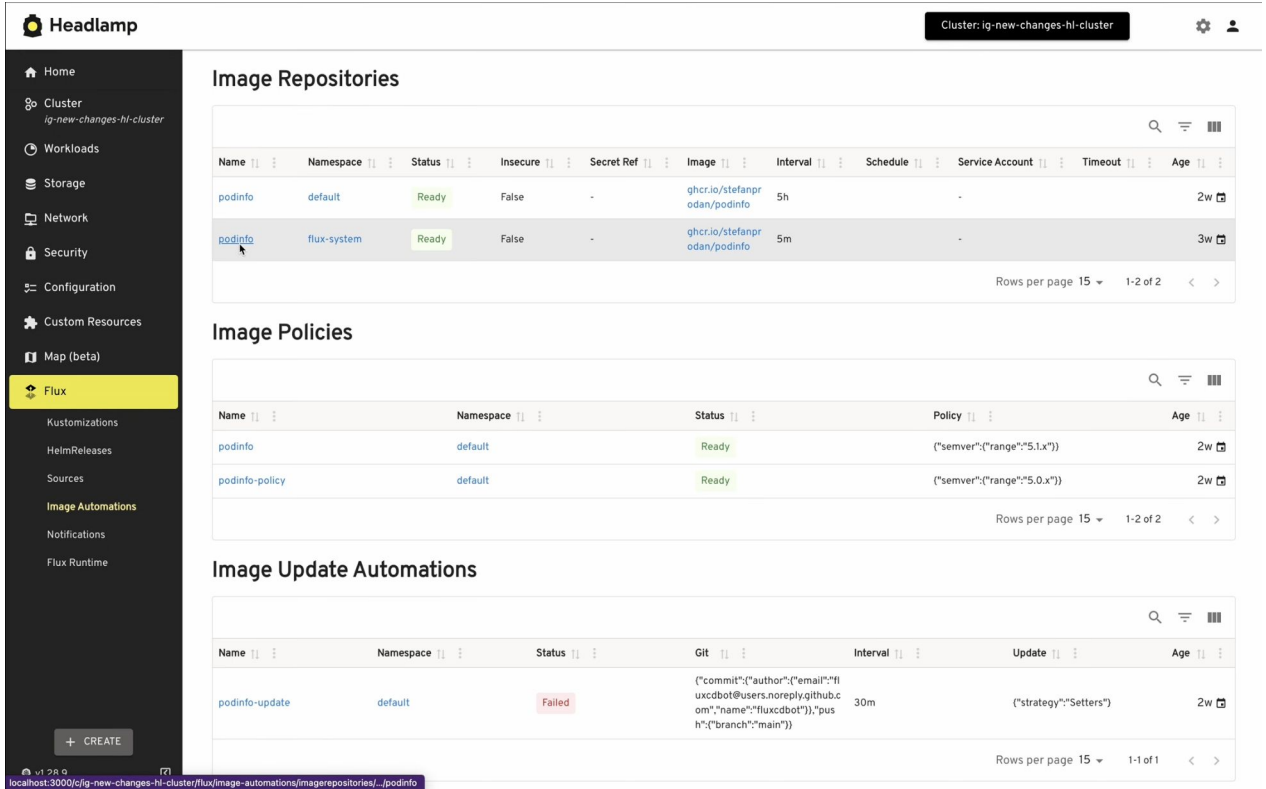
Flux Operator

The Flux Operator manages the lifecycle of the CNCF Flux project.

Flux UIs


Enter Filter
reset
chatbot-ui
Logs
Describe
Pods
Running
Dependencies
chatbot-ui
Address
chatbot-ui.default.svc.cluster.local
http://127.0.0.1:3000 (port-forward)
Port-forward command
Sync
Applied about 3 hours ago 3bc14c1b (flux-system/builtin-apps)
expressjs-test-app
Logs
Describe
Pods
Running
Dependencies
expressjs-test-app
expressjs-test-app
Address
expressjs-test-app.default.svc.cluster.local
http://127.0.0.1:8080 (port-forward)
Port-forward command
Sync
Applied about 3 hours ago 3bc14c1b (flux-system/builtin-apps)
notification-controller
Logs
Describe
Pods
Address
SOURCES: Ready(2/2)
KUSTOMIZATIONS: Reconciling(2/3)
HELM-RELEASES: Ready(3/3)

Flux UIs



CLOUD NATIVE COMPUTING FOUNDATION

Flux adapts to your organizational structure and growth velocity

CD growth factors

Microservices proliferation
Apps made out of many moving pieces each with its own release cadence and delivery workflow

SLAs & High availability & Disaster Recovery
Multi-AZ, Multi-region app instances

Security constraints
Multi-tenancy, Network isolation, GDPR

Business expansion
New apps are developed, while legacy apps still require maintenance

CLOUD NATIVE
COMPUTING FOUNDATION

We'll need to deploy thousands of workloads over hundreds of clusters 😱

Can Flux do it?

Flux scaling strategies

Source Optimizations
Controller Fine Tuning

↓

Vertical Scaling

↓

Horizontal Scaling & Sharding

Optimizations & Fine Tuning

Source optimizations

- Migrate from Helm HTTP/S repositories to OCI
- Use dedicated Git repositories for Flux
- Split the Kubernetes resources into multiple Flux Kustomizations to take advantage of concurrent reconciliations

Controller fine tuning

- Use persistent storage for Flux internal artifacts
- Use tmpfs for in-memory kustomize builds

Vertical scaling

Resource limits

- Increase controller CPU and Memory limits
- Increase the controller reconciliation concurrency level based on CPU limits

Rate limits

- Monitor Kubernetes API requests rate limits error rate and adjust the reconciliation concurrency level

https://fluxcd.io/flux/installation/configuration/vertical-scaling

Flux “Mean Time To Production” benchmark

Starting with Flux v2.2, we have set up benchmarks that measure Mean Time To Production (MTTP). The MTTP benchmark measures the time it takes for Flux to deploy application changes into production.

Specs:

GitHub hosted-runner
ubuntu-latest-16-cores

Kubernetes KinD
v1.28.0 / 3 nodes

Flux source-controller
1CPU / 1Gi / concurrency 10

Flux kustomize-controller
1CPU / 1Gi / concurrency 20

Flux helm-controller
2CPU / 1Gi / concurrency 10

Objects	Type	Flux component	Duration	Max Memory
500	OCIRepository	source-controller	45s	65Mi
500	Kustomization	kustomize-controller	2m2s	72Mi
500	HelmChart	source-controller	45s	68Mi
500	HelmRelease	helm-controller	2m55s	350Mi
1000	OCIRepository	source-controller	1m30s	67Mi
1000	Kustomization	kustomize-controller	4m15s	112Mi
1000	HelmChart	source-controller	1m30s	110Mi
1000	HelmRelease	helm-controller	8m2s	620Mi

https://github.com/fluxcd/flux-benchmark

Horizontal scaling

To enable horizontal scaling, each controller can be deployed multiple times with a unique label selector which is used as the sharding key.

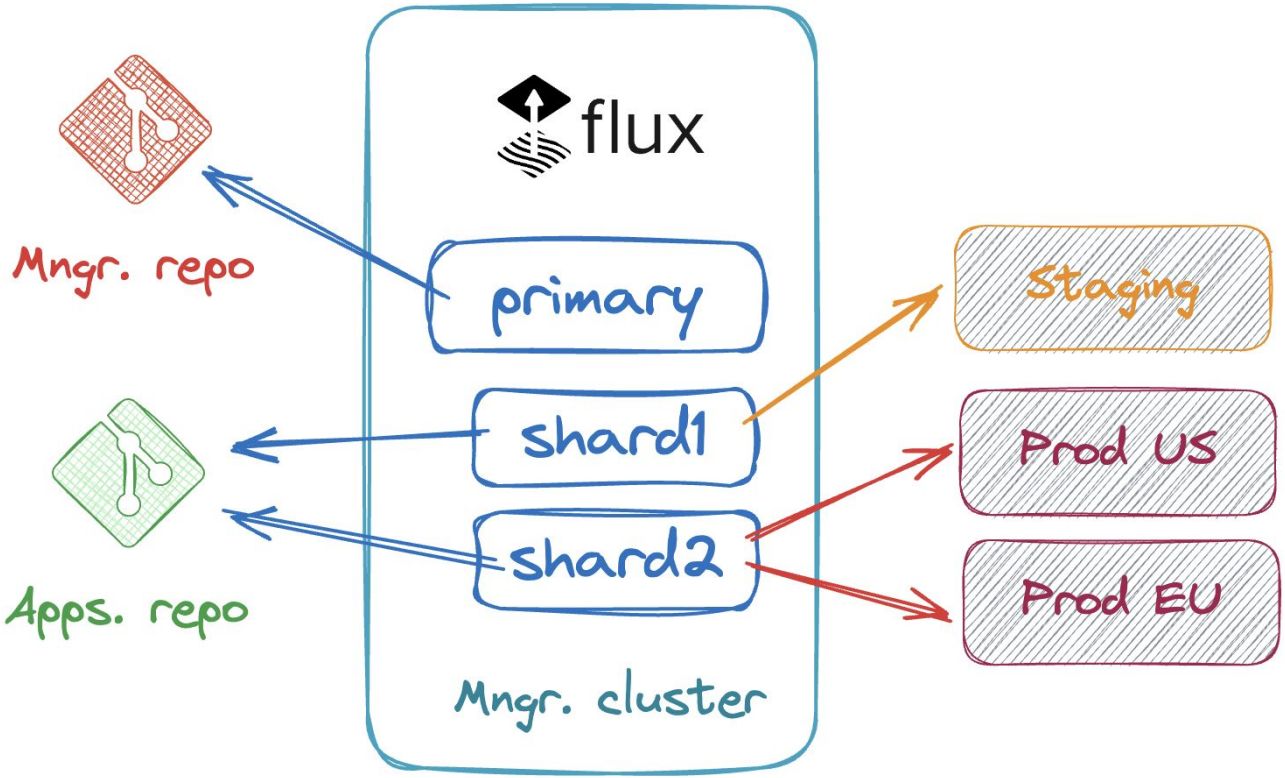
To spread the load between controller instances, the Flux resources can be assigned to a particular instance using the sharding.fluxcd.io/key label.

Sharding strategies

- Multi-tenant (instance per tenant)
- Multi-cluster (instance per cluster group)

https://fluxcd.io/flux/installation/configuration/sharding

Flux sharding for multi-cluster delivery



CLOUD NATIVE COMPUTING FOUNDATION

Flux Updates 2024

https://fluxcd.io/roadmap

General Availability

Flux v2.3 (Q2 2024)
Promoted to GA the Flux Helm APIs and the Flux Helm functionalities.

- HelmRepository
- HelmChart
- HelmRelease

Flux v2.4 (Q3 2024)
Promoted to GA the S3-compatible storage API.

- Bucket

Helm OCI improvements

Flux v2.3 came with support for HelmReleases to refer to OCIRepositories, as an alternative to HelmRepositories/HelmCharts.

- Better UX when debugging Helm releases
- Allow reusing the same chart between releases
- Allow pinning Helm charts by OCI digests
- Allow automating Helm upgrades using semver ranges scoped to release candidates only
- Provenance verification with Cosign and Notary Notation

Flux Operator and OpenShift Compatibility

Flux can be installed on Red Hat OpenShift cluster directly from OperatorHub or from the RedHat production-ready catalog.

The Flux Operator, a new component in the Flux CD ecosystem **developed by ControlPlane** that automates the lifecycle management of Flux components and streamlines the GitOps workflows for Kubernetes clusters.

Terraform provider improvements

The Flux Terraform provider has undergone a major refactoring and now supports

- air-gapped bootstrap
- drift detection and correction for Flux components
- the ability to upgrade and restore the Flux controllers in-cluster

Starting with this release, the provider is fully compatible with OpenTofu.

Azure DevOps OIDC Authentication

Starting with Flux v2.4, you can configure source-controller and image-automation-controller to authenticate against Azure DevOps repositories using AKS Workload Identity.

Big thanks to **Dipti** from **Microsoft** for contributing this feature and for helping the Flux team maintaining the Azure integration on the long-run.

GitHub App OIDC Authentication (WIP)

In a future release, Flux will support OIDC-based authentication against Git repositories using GitHub App credentials.

This feature is **work-in-progress** and will be available early next year.

CLOUD NATIVE
COMPUTING FOUNDATION

v2.5 (Q1 2025)

Status: In progress

The primary goal of this milestone is to make a generally available release for the Flux image automation APIs.

https://fluxcd.io/roadmap

Get involved with Flux!

We invite the community to help us shape the future of Flux.

We have in place an **RFC process** for new features and enhancements proposals. We are keen to work with the community on RFCs and drive the project forward in a sustainable manner.

We want to **enable community members to take full ownership** of Flux features and share the responsibility of feature stability and longer-term maintenance.

https://github.com/fluxcd/flux2/tree/main/rfcs