AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

CON307

GitOps as a continuous operating model

Nicholas Thomson (he/him)
Software Development Engineer, AWS Kubernetes
AWS

Sheetal Joshi (she/her)
Senior Developer Advocate, AWS Kubernetes
AWS

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

What is GitOps?

GitOps and Kubernetes best practices

Module 1: Deploying an application using Flux

Module 2: Introducing a change to the desired state

Module 3: Reconciling from detected drift

Module 4: Expanding GitOps to Infrastructure-as-Config

Planning for GitOps adoption

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is GitOps?

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Principles of GitOps



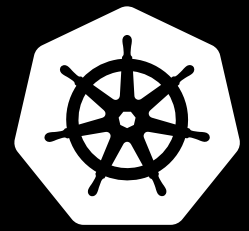
A system managed by GitOps must have its desired state expressed **declaratively**



The desired state is stored in a way that enforces **immutability and versioning,** and retains a complete version history



Software agents **automatically pull** the desired state declarations from the source



Software agents **continuously** observe actual system state and **attempt to apply** the desired state

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Comparison to traditional CD systems

Traditional CD Systems

- Push on an edge-triggered event
- Deploy only the difference
- The system may drift between deployments
- CD process must have access to deployment environment

GitOps-based systems

- The system continuously polls for changes
- The entire system is declared for any deployment
- The system will correct any drift
- Pipelines run within the system in-scope authorization

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Managing state

GitOps does **not** handle declaration of:

- Persistent application data – User uploads
- Schema-based deployments – Database schema

Backup and restoration plans are still vital

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

GitOps tools and Kubernetes best practices

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

History of GitOps and Kubernetes

The Kubernetes API server is founded on the same principles

Allows the developer to extend GitOps through the Kubernetes operator pattern

However, GitOps does **not** need to be Kubernetes

GitOps does **not** even need to be Git

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kubernetes tooling

Kubernetes requires an in-cluster software agent

Flux - developed by WeaveWorks

- Incubating project in CNCF

Implemented as a set of Kubernetes extensions



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Using Kubernetes templating engines

Raw Kubernetes manifests are simple but will not scale

Using a template engine will help with the DRY principle

Two options:

1. Kustomization
2. Helm charts

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kustomization

Override manifests using overlays

Only supports build/compile-time overrides

```
# base/kustomization.yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
resources:
- front-end.yaml
- database.yaml

# overlays/production/kustomization.yaml
resources:
- ../base
namePrefix: prod-
```

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Helm

Packaged as versioned “charts”

Can be indexed and stored in repositories

Supports value files for overrides

```
# deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: front-end
spec:
  replicas: {{ .Values.replicaCount }}
  template:
    spec:
      containers:
      - image: "{{ .Values.image.repository }}:{{
.Values.image.tag}}"

# values.yaml
replicaCount: 2
image:
  repository: ”front-end-image”
  tag: ”v1.0.0"
```

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Let’s get started

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Access the event engine

Go to

https://dashboard.eventengine.run

Use the event hash

96b4-108ab9a3f4-98

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module 1: Deploying an application with Flux

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Deploying an application with Flux

Module goals:

- Forking and creating your own demo repositories
- Creating a development cluster using KIND
- Installing Flux and authenticating using your GitHub credentials
- Applying your demo repository manifests to the dev cluster
- Watching the application be deployed

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hands-on time

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module conclusion

You have configured the purest form of GitOps with Kubernetes

- Declarative system state stored in Git
- Polling Git repository for changes
- Polling cluster for changes

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module 2: Introducing a change to the desired state

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing a change to the desired state

Module goals:

- Bootstrap a production cluster
- Modify the desired state using only Git
- Watch Flux reconcile your changes automatically

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hands-on time

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module conclusion

The system detected desired state updates

The developer did not need to access the cluster

No CD system needed credentials for the environment

Using pull requests as the primary mechanism for dev branch to production branch

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module 3: Reconciling from detected drift

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Reconciling from detected drift

Module goals:

- Adding observability into the cluster using the notification controller
- Modifying the deployment environment to simulate chaos in the production system
- Watching Flux detect and redeploy changes
- Observing the detected changes externally using SNS

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hands-on time

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module conclusion

The system continuously monitors the state of the deployed application

The system will re-apply the desired state

The notification controller can filter events and conditionally send to external services (for example, on-call rotations)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module 4: Expanding GitOps to IaC

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Expanding GitOps to IaC

Module goals:

- Using Flux to deploy third-party Helm charts
- Declaring infrastructure as Kubernetes resources
- Reconciling infrastructure-as-config using two GitOps reconcilers
  - Flux reconciler for custom resource
  - ACK reconciler for AWS resource

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hands-on time

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module conclusion

Multiple GitOps reconcilers working together

Declaratively define AWS resources

- Reconciliation of drift within AWS

Permissions handled in the cluster/IAM, away from developers

Clusters as config?

- Cluster API
- Amazon Controllers for Kubernetes (ACK) Amazon Elastic Kubernetes Service (Amazon EKS) controller
- Cross-plane project

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workshop conclusion

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Adopting GitOps in production

Handling secrets?

- Sealed secrets by Bitnami
- Kubernetes Secrets Store CSI Driver with Vault by HashiCorp

Tooling for better observability and automated health checking

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

> **Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.**

Melvin E. Conway

American Computer Scientist

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Embracing Conway's Law

GitOps can help you embrace Conway's Law

Design repositories and integration with GitOps based upon your organization's roles and responsibilities

Infrastructure teams can focus on application deployment and scaling

Developers can focus on the application code

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Adoption path

Broad guidance for adopting GitOps in existing systems

1. Declarative infrastructure, not reconciled, potentially imperative
2. GitOps for workloads, with automated reconciliation and drift detection
3. Pull-based deployments from source, Helm charts, images, and so on
4. Focus on visibility and socializing changes; approval workflows get refined

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Workshop take-aways

GitOps is more than just CI/CD or ”DevOps using Git”

An idea of how to deploy an application and IaC using GitOps

Kubernetes best practices

A path forward to your own team’s adoption

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Nicholas Thomson

Sheetal Joshi

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.