# // CODE ☑ CLUSTER: BOOSTING DEVELOPMENT WITH A LOCAL KUBERNETES OPS PLATFORM

Johannes Schnatterer, Cloudogu GmbH



@ @schnatterer@floss.social

in in/jschnatterer

Version: 202406052141-1c4d47a





## What is your profession?

Software Engineer / Developer



## What is your profession?

Platform Engineer / Ops person



## Who uses Kubernetes for local development?

## k3d Minkube Microk8s K3S

## KIND Docker Desktop KOS Rancher

Desktop



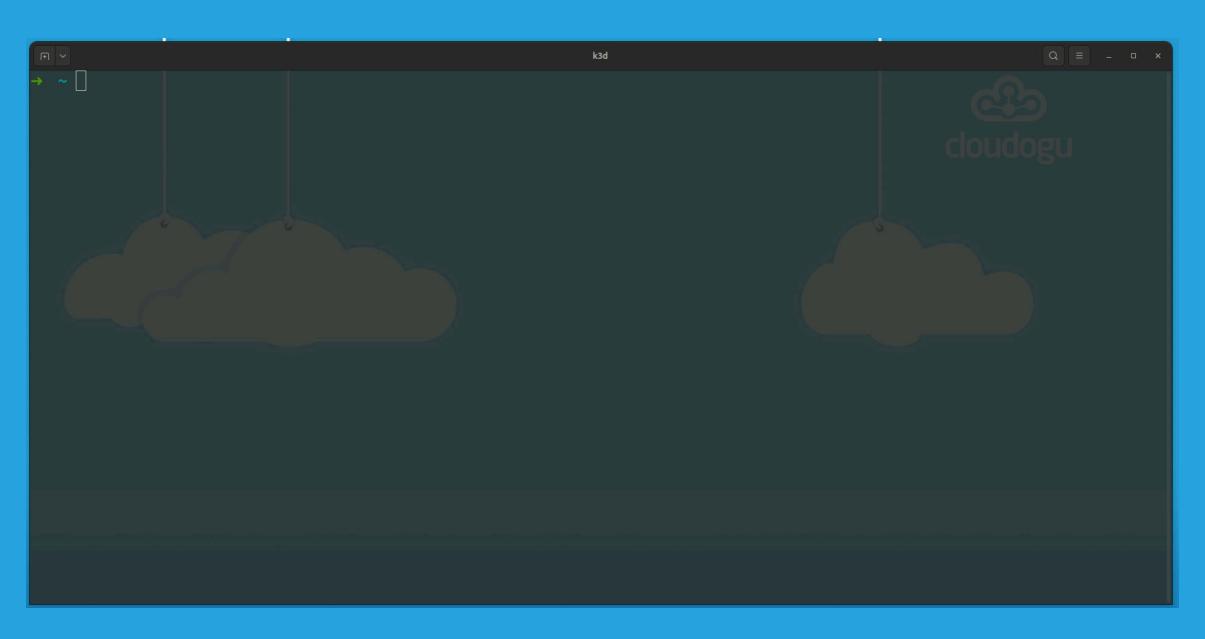
Kubernetes is a platform for building platforms. It's a better place to start; not the endgame.

10:04 PM · Nov 27, 2017

237 Reposts 44 Quotes 748 Likes 22 Bookmarks

twitter.com/kelseyhightower/status/935252923721793536

## Start a local k8s cluster with one command



## Next, start the platform









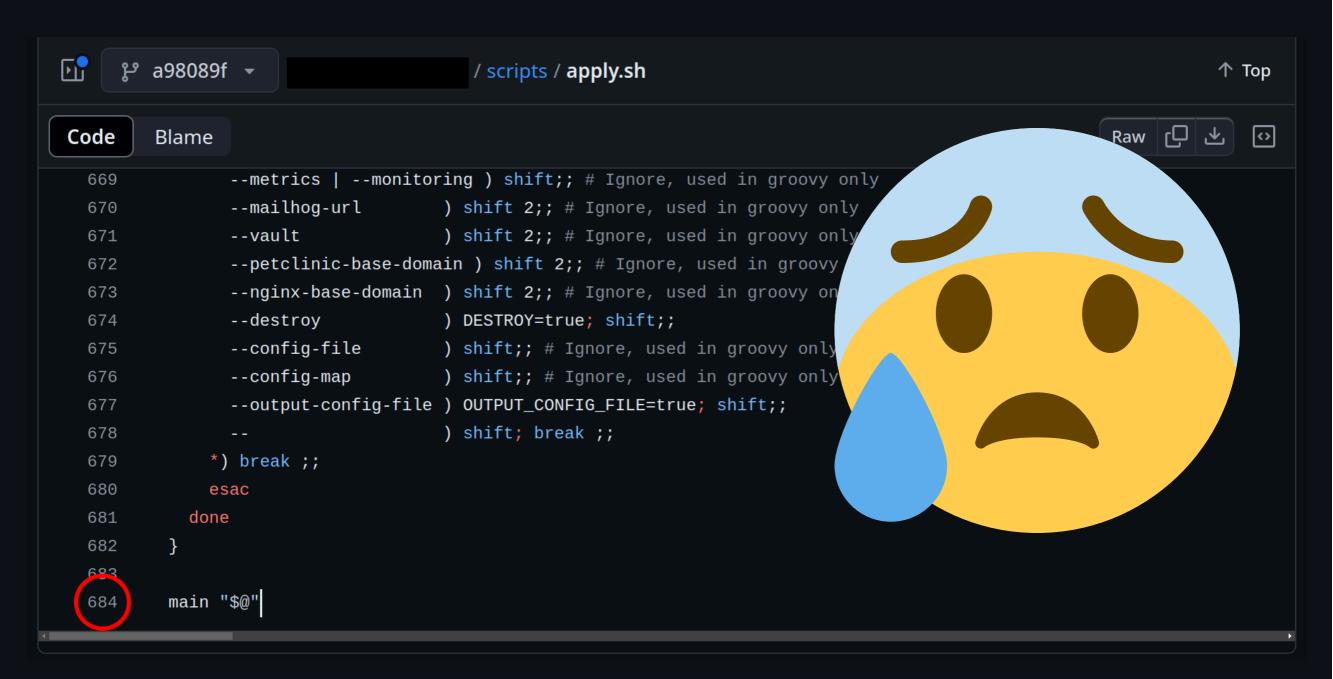








## So, let's write a little script...



Why not start the platform with one command?

## Meet GOP

a GitOps-based operational stack (platform)





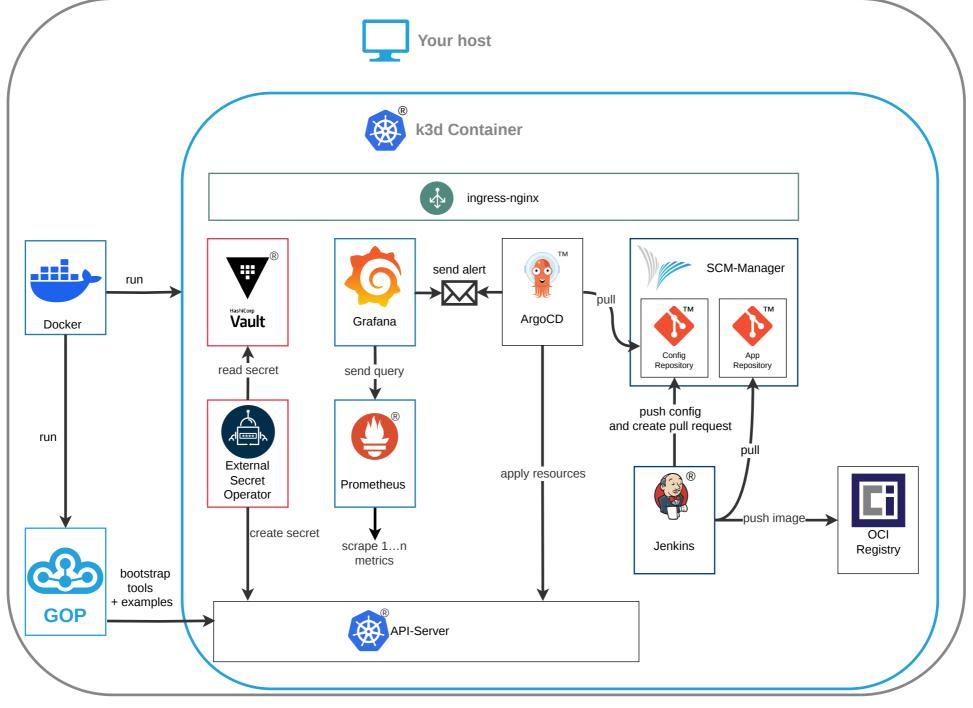


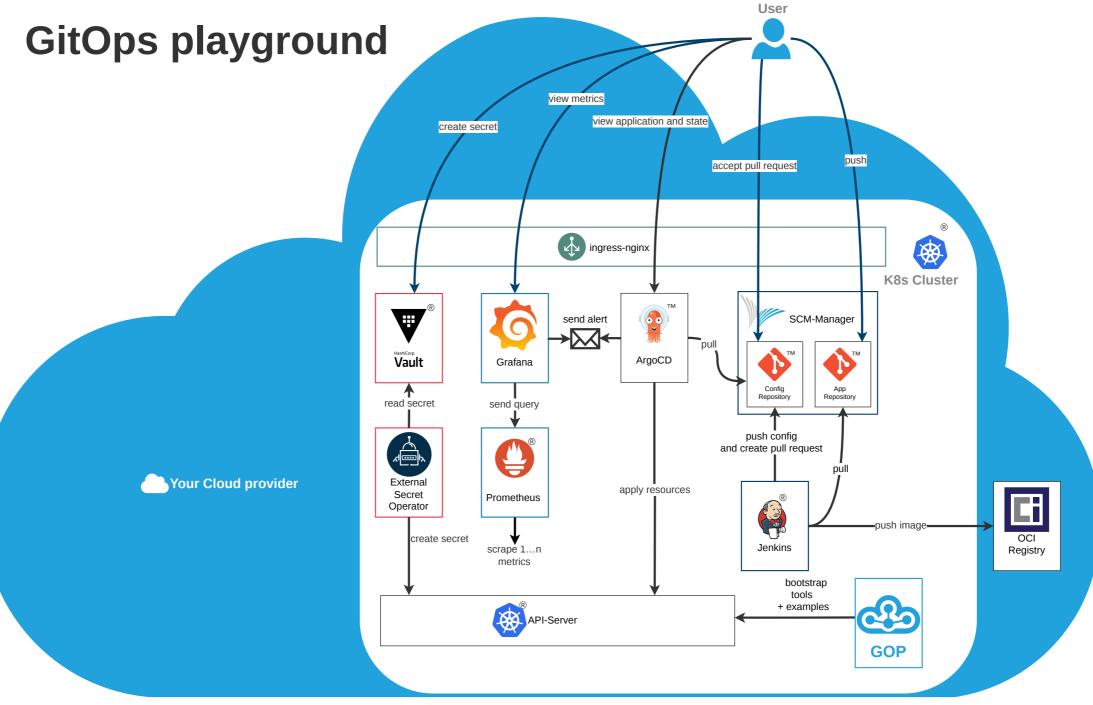




```
COMMIT='42e5f80'
bash <(curl -s \
  "https://raw.githubusercontent.com/cloudogu/gitops-playground/$COMMIT/scripts/init-cluster.sh") \
  --bind-ingress-port=80 \
   && docker run --rm -t -u (id -u) \setminus
    -v ~/.config/k3d/kubeconfig-gitops-playground.yaml:/home/.kube/config \
    --net=host \
    ghcr.io/cloudogu/gitops-playground:$COMMIT --yes --base-url=http://localhost --ingress-nginx \
      --argord --monitoring --vault=dev --mailhog
```







### scripts/init-cluster.sh

```
k3d cluster create gitops-playground \
  # Mount port for ingress
  -p 80:80@server:0:direct \
  # Pin image for reproducibility
  --image=rancher/k3s:v1.29.1-k3s2 \
  # Disable built-in ingress controller, because we want to use the same one locally and in prod
  --k3s-arg=--disable=traefik@server:0 \
  # Allow node ports < 30000
  --k3s-arg=--kube-apiserver-arg=service-node-port-range=8010-65535@server:0 \
  # Hacks to make Docker available in Jenkins
  -v /var/run/docker.sock:/var/run/docker.sock@server:0 \
  -v /etc/group:/etc/group@server:0 -v /tmp:/tmp@server:0 \
  -p 30000:30000@server:0:direct
# Write kubeconfig to ~/.config/k3d/kubeconfig-gitops-playground.yaml
k3d kubeconfig write gitops-playground
```

## docker run

```
docker run
  # Remove container after running, keeping your device clean
  # (remove in case of error to preserve logs)
  - - rm
  # Colorful output, please
  -t
  # Run as current user, so files written to /tmp are accessible for you
  -u $(id -u) \
  # Mount kubeconfig for k3d
  -v ~/.config/k3d/kubeconfig-gitops-playground.yaml:/home/.kube/config \
  # Make k3d cluster available
  --net=host \
ghcr.io/cloudogu/gitops-playground:$COMMIT #Params for image go here
```

## ghcr.io/cloudogu/gitops-playground

- OCI image
- Contains logic to install and configure the tools
- App written in Groovy (and bash (2))
- Additional resources to run e.g. in air-gapped envs







## Your turn



@ giphy.com/gifs/JIX9t2j0ZTN9S

## Exercises "\"





Deploy broken app via GitOps, get alerted and fix problem

• GitOps process 🕠 🗀 😭









Promote a change in code all the way to production using GitOps

Monitoring (2)





Deploy a Grafana dashboard for an app using GitOps

Secrets Management





Integrate secrets into app, propagate updates automatically

Progressive Delivery



Watch a canary release live with argo roll

## **Getting started**

- Login: admin / admin
- scmm.localhost
- argocd.localhost
- ografana.localhost
- wault.localhost
- mailhog.localhost
- jenkins.localhost



## **Exercise: GitOps+Alerting Parting**

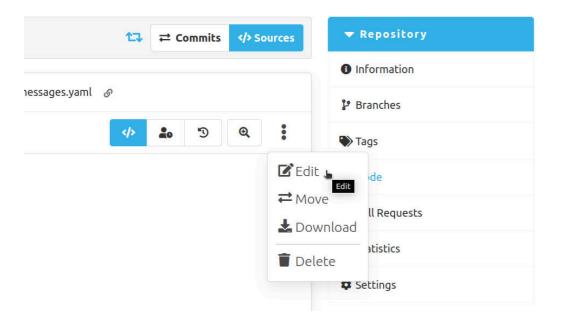




Deploy broken app via GitOps, get alerted and fix problem

#### **Exercise: Alerting**

- 1. Add repo to Argo CD
- 2. Create Argo CD Application YAML
- 3. Deploy, receive alert and fix app
- Hint: Edit file in SCM-Manager



#### 1. Add repo to Argo CD

Add this repo to Argo CD (via GitOps):

http://scmm-scm-manager.default.svc.cluster.local/scm/repo/exercises/broken-application

- 1. Add to repositories in Argo CD's config:
  - scmm.localhost/scm/repo/argocd/argocd/code/sources/main/argocd/values.yaml
- 2. Authorize Project to use the repo by adding it to sourceRepos here:
  - scmm.localhost/scm/repo/argocd/argocd/code/sources/main/projects/example-apps.yaml

#### 2. Create Argo CD Application YAML

- Go to argocd.localhost, click + NEW APP
- Enter Name: broken
- Click on Project Name, choose example apps
- Click on Repository URL, choose the broken-application repo
- Enter Path: .
- Click on Cluster URL, choose https://kubernetes.default.svc
- Enter Namespace: example-apps-staging
- At the top, click EDITAS YAML and copy content

#### 3. Deploy, receive alert and fix app

- Paste content here:
  - scmm.localhost/scm/repo/argocd/example-apps/code/sourceext/create/main/argocd/
- Enter Filename: broken.yaml, and commit message, then click



Go to argocd.localhost/applications/argocd/broken, click



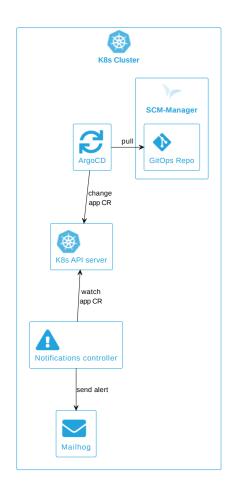
- Check email in mailhog.localhost
- Follow link to ArgoCD-UI, analyse error
- Fix error in repo:



Go to argocd.localhost/applications/argocd/broken, click

- Follow X ingress link to open application in browser
- Have a closer look at the concepts behind this

## **Alerting in GOP**





### Argo CD config:

- github.com/cloudogu/gitops-playground/blob/0.2.0/argocd/argocd/argocd/values.ftl.yaml
- scmm.localhost/scm/repo/argocd/argocd/code/sources/main/argocd/values.yaml

#### See also

- GitOps repo structure in GOP
- \*\* Exercise: GitOps process



## **Exercise: GitOps process** • Ci









Promote a change in code all the way to production using GitOps

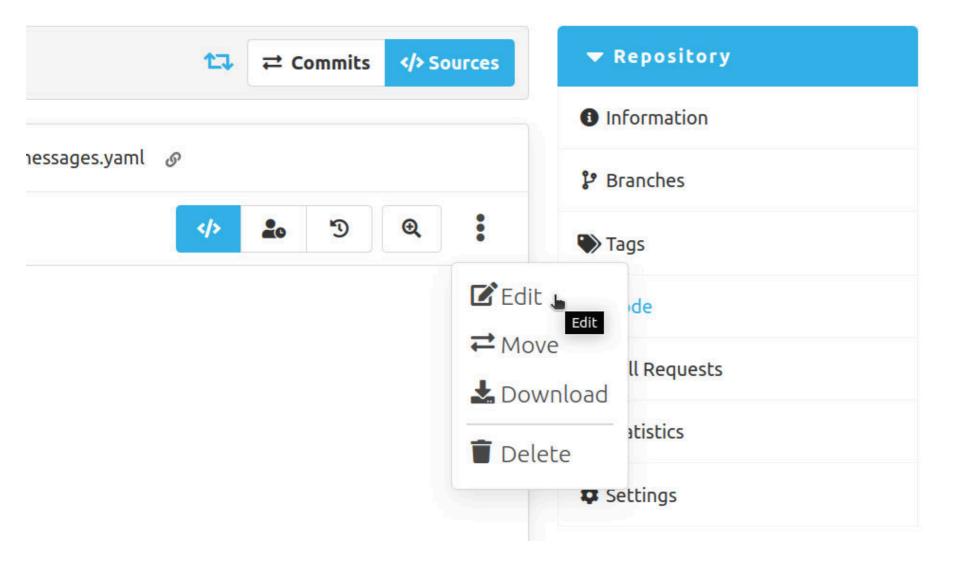
- Warmup
- GitOps with CI server and promotion



- 1. Open Argo CD Application:
  - argocd.localhost/applications/example-apps-staging/petclinic-plain
- 2. Open app in Browser:
  - staging.petclinic-plain.petclinic.localhost
- 3. Change welcome message in config repo:
  - scmm.localhost/scm/repo/argocd/example-apps/code/sources/main/apps/spring-petclinic-plain/staging/generatedResources/messages.yaml
- 4. Press CREFRESH in ArgoCD UI
- 5. Restart deploy in ArgoCD UI 🔁 Watch GitOps deployment
- 6. Reload app in Browser 🗅 Shows new message 🔯



## Hint: Edit file in SCM-Manager



## GitOps with CI server and promotion 🚀

#### First:

Accept Pull Request for petclinic-plain to deploy prod



#### Then:

- 1. Change app, build image, deploy staging
- 2. Accept PR, deploy production

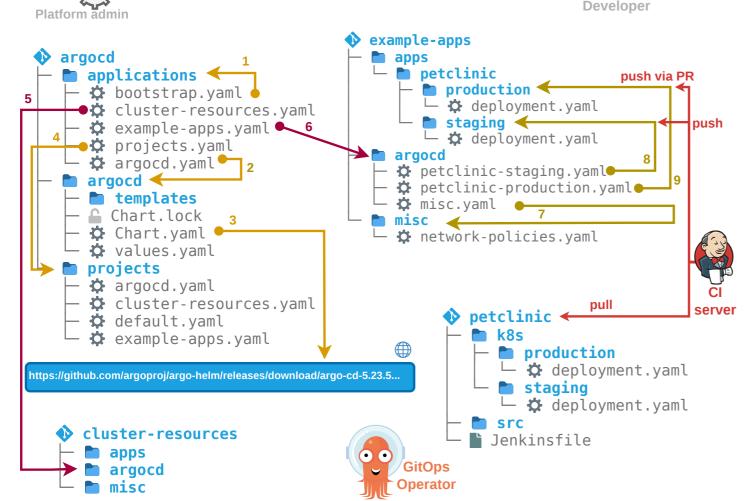
#### 1. Change app, build image, deploy staging

- 1. Open staging app in Browser:
  - staging.petclinic-plain.petclinic.localhost
- 2. Change welcome message in app repo
  - scmm.localhost/scm/repo/argocd/petclinic-plain/code/sources/main/src/main/resources/messages/messages.properties
- 3. Wait for Build
  - jenkins.localhost/job/example-apps/job/petclinic-plain/job/main
- 4. Open Argo CD Application for staging:
  - argocd.localhost/applications/example-apps-staging/petclinic-plain
- 5. Press Greffesh in ArgoCD UI Watch GitOps deployment
- 6. Reload app in Browser D Shows message in staging 🗟

#### 2. Accept PR, deploy production

- 1. Open production app in Browser:
  - production.petclinic-plain.petclinic.localhost
- 2. Open Argo CD Application for production:
  - argocd.localhost/applications/example-apps-production/petclinic-plain
- 3. Accept Pull Request for petclinic-plain
  - scmm.localhost/scm/repo/argocd/example-apps/pull-requests
- 4. Press CREFRESH in ArgoCD UI Watch GitOps deployment
- 5. Reload app in Browser D Shows message in production 😂 😂
- 6. Have a closer look at the concepts behind this

## GitOps repo structure in GOP



- scmm.localhost/scm/repo/argocd/petclinic-plain/code/sources/main/Jenkinsfile uses
- github.com/cloudogu/gitops-build-lib

cloudogu.com/blog/gitops-repository-patterns-part-6-examples



## **Exercise: Monitoring (2)**



Deploy a Grafana dashboard for an app using GitOps

- 1. Expose metrics
- 2. Create specific Grafana dashboard JSON
- 3. Deploy dashboard via GitOps
- 4. Watch metrics

### 1. Expose metrics

- Enable metrics export on nginx via GitOps
  - scmm.localhost/scm/repo/argocd/example-apps/code/sourceext/edit/main/apps/nginx-helm-umbrella/values.yaml

```
nginx:
    metrics:
        enabled: true
        serviceMonitor:
            enabled: true
# labels:
# release: kube-prometheus-stack
```

- Check if servicemonitor was created:
  - argocd.localhost/applications/example-apps-production/nginx-helm-umbrella

### 2. Create specific Grafana dashboard JSON

- ografana.localhost/dashboard/import
- Paste content from here
   and click
  - github.com/nginxinc/nginx-prometheus-exporter/blob/v1.2.0/grafana/dashboard.json
- Name: nginx-helm-umbrella
- Click Select a Prometheus data source: Prometheus
- Click Import

#### 3. Deploy dashboard via GitOps

- Copy JSON from grafana.localhost/d/MsjffzSZz?editview=dashboard\_json
- to 🍑 scmm.localhost/scm/repo/argocd/example-apps/code/sourceext/create/main/apps/nginx-helm-umbrella

```
apiVersion: v1
kind: ConfigMap
metadata:
   name: nginx-helm-umbrella-dashboard
   labels:
      grafana_dashboard: "1"
data:
   dashboard.json: |-
      # Paste your JSON, INDENTED BY 4 SPACES here
      {...
```

- Path: Add / templates
- Enter Filename: dashboard.yaml, commit message, Click commit
- Click SYNC here: pargocd.localhost/applications/example-apps-production/nginx-helm-umbrella
- Check if == configmap was created

#### 4. Watch metrics

- Follow ingress link to open app in browser
- Generate traffic by reloading
- Enjoy your dashboard
  - grafana.localhost/d/MsjffzSZz 🔯



# **Exercise: Secrets Management**





Integrate secrets into app, propagate updates automatically

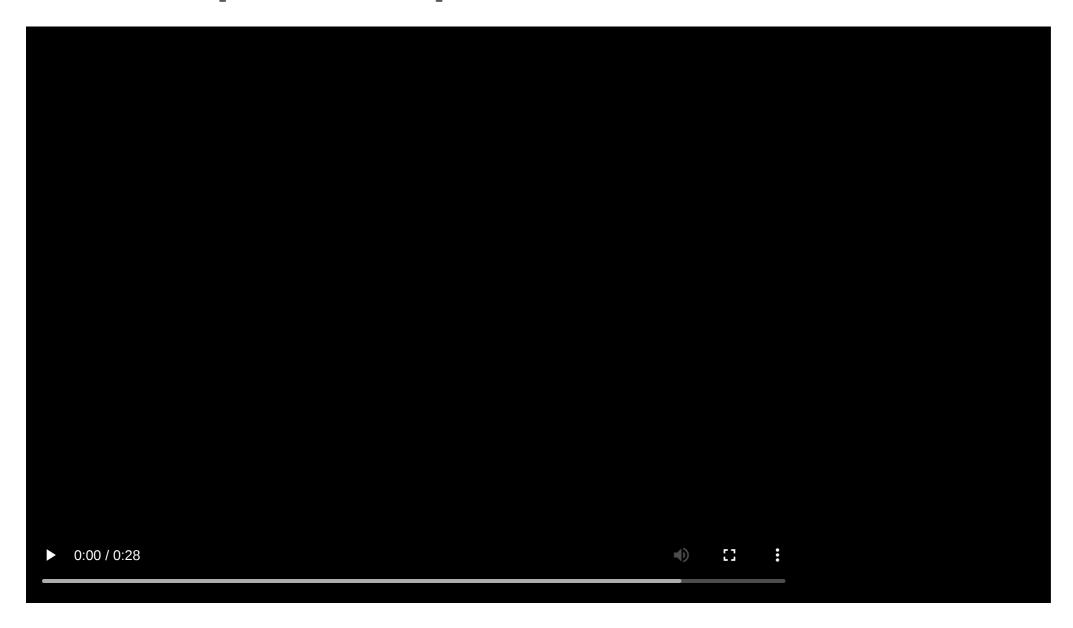
- Warmup
- Mount secret into app

# Warmup 💖

- Secret exposed via HTTP
  - staging.nginx-helm.nginx.localhost/secret
- Change in Vault:
  - vault.localhost/ui/vault/secrets/secret/edit/staging/nginx-helm-jenkins
- Watch it propagate automatically (<2 min)</li>
   Either reload Browser or:

```
while ; do echo -n "$(date '+%Y-%m-%d %H:%M:%S'): " ; \
   curl staging.nginx-helm.nginx.localhost/secret/ ; echo; sleep 1; done
```

## **Warmup** in time-lapse



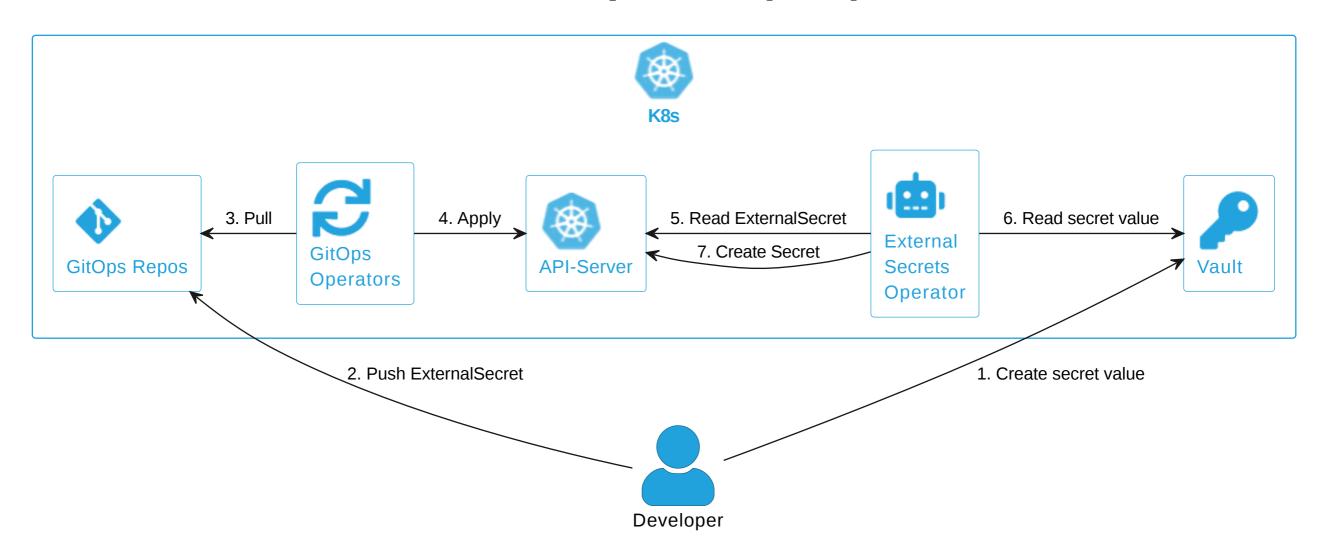
## Mount secret into app 🚀

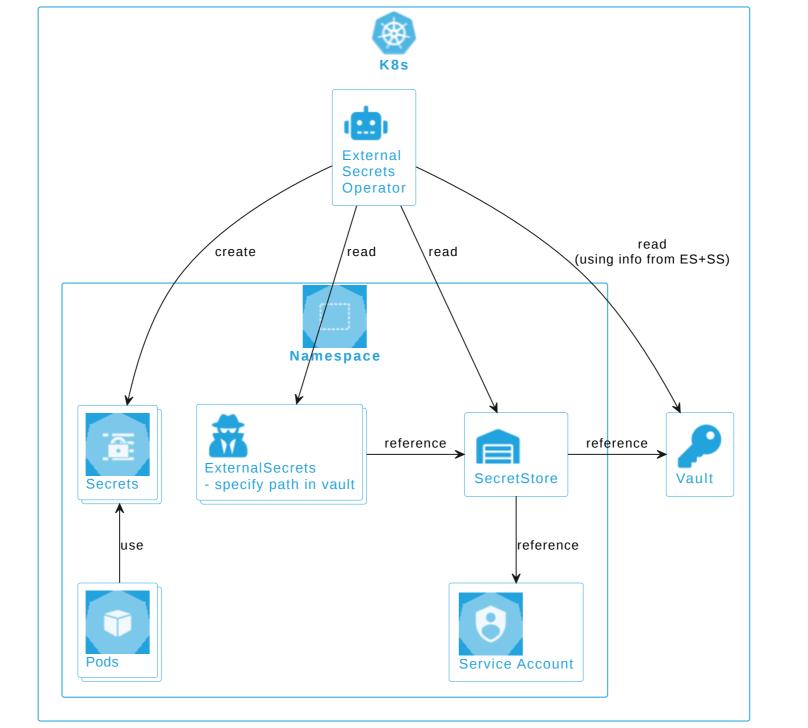
Create a new secret in Vault and mount it into an app

- 1. Create secret in Vault and sync into cluster via ExternalSecret
- 2. Use secret in app

Let's start with some basics

## **External Secrets Operator (ESO) with Vault**





### **ESO+Vault config in GOP**

- SecretStore per Namespace:
  - scmm.localhost/scm/repo/argocd/cluster-resources/code/sources/main/misc/secrets/secret-store-staging.yaml
- Example ExternalSecret:
  - scmm.localhost/scm/repo/argocd/nginx-helm-jenkins/code/sources/main/k8s/staging/external-secret.yaml
- Mounted into app:
  - scmm.localhost/scm/repo/argocd/nginx-helm-jenkins/code/sources/main/k8s/values-shared.yaml

#### Create secret in Vault and sync into cluster via ExternalSecret

- 1. Create secret in Vault
- wault.localhost/ui/vault/secrets/secret
- Click Create secret +
- Path for this secret: production/nginx-helm-umbrella
- key: my-secret, value: choose any
- 2. Deploy ExternalSecret via GitOps ( \* example on previous slide)
  - scmm.localhost/scm/repo/argocd/example-apps/code/sourceext/create/main/apps/nginx-helm-umbrella
- Path: Add / templates
- Filename: secret.yaml
- 3. Check if secret was created:
  - argocd.localhost/applications/example-apps-production/nginx-helm-umbrella

#### Use secret in app

- 4. Mount secret into NGINX ( gexample on previous slide):
  - scmm.localhost/scm/repo/argocd/example-apps/code/sourceext/edit/main/apps/nginx-helm-umbrella/values.yaml
  - Hint: Add one nginx.extraVolumes and one nginx.extraVolumeMounts
- 5. Go to pargocd.localhost/applications/argocd/broken, click
- 6. Follow ingress link to open application in browser, add path / secret 😂
- 7. Optional: Change the secret in Vault and wait for sync as in Warmup 🤓
- Secret in vault is transient, i.e. gone after restart (dev mode)

## Johannes Schnatterer, Cloudogu GmbH



b Join my team: cloudogu.com/join/cloud-engineer







## Legal

- Vault is a registered trademark of Hashicorp
- Docker is a registered trademark of Docker Inc
- Kubernetes is a registered trademark of the Linux Foundation
- Git is a registered trademark of Software Freedom Conservancy
- Grafana is a registered trademark of Grafana Labs The Grafana Labs Marks are trademarks of Grafana Labs, and are used with Grafana Labs' permission. We are not affiliated with, endorsed or sponsored by Grafana Labs or its affiliat