November, 2024

Plone on Kubernetes

Harnessing the power of Kubernetes for modern Plone deployments

Fabiano Weimar [Xiru]
xiru@xiru.org


de meeste mensen zwijgen, een enkeling stelt een daad
Plone


Velocidade, Força e Beleza
Plone
www.plone.org
python brasil [9]
9ª Conferência Brasileira de Python

Plone on Kubernetes

- Plone is an open-source content management system, trusted by governments, universities and businesses all over the world.
- Kubernetes, also known as k8s, is an open-source system for managing containerized applications. It provides mechanisms for deployment and scaling applications.
- Kubernetes is part of the Cloud Native Computing Foundation (CNCF) (graduated project).
    - Other CNCI projects will also be featured along this presentation.
- Plone and Kubernetes combined are a powerful duo.

Kubernetes Benefits

- Scalability: Easily scale applications up or down to meet demand
- Fault Tolerance: Automatically replace failed containers
- Self-Healing: Automatically recover from failures
- Efficient Resource Utilization: Optimize resource allocation
- Declarative Configuration: Define desired state using YAML
- Rich Ecosystem: the "de facto standard" of modern cloud-based infrastructure

kubernetes

Continuous Integration & Delivery

Security & Compliance

Cloud Native Network

Streaming & Messaging

Container Runtime

Coordination & Service Discovery

Container Runtime

Application Definition & Image Build

Service Proxy

Coordination & Service Discovery

Security & Compliance

Observability

Continuous Integration & Delivery

Container Registry

Application Definition & Image Build

Service Mesh

Observability

Scheduling & Orchestration

Automation & Configuration

Scheduling & Orchestration

Service Mesh

Security & Compliance

Observability

Cloud Native Storage

Key Management

Key Management

Security & Compliance

Database

Database

KubeVela

Application Definition & Image Build

KubeVirt

Application Definition & Image Build

Security & Compliance

Chaos Engineering

Cloud Native Storage

Streaming & Messaging

Security & Compliance

Continuous Optimization

Feature Flagging

Continuous Integration & Delivery

Observability

Application Definition & Image Build

Streaming & Messaging

Observability

Scheduling & Orchestration

Kubernetes Basics #1

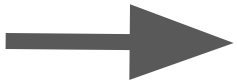
- Pods: The smallest deployable unit of computing, consisting of one or more containers
- Deployments: Manage the replication and updates of Pods
- ReplicaSets: Manage a fixed number of identical Pods
- StatefulSets: Manage stateful applications, like databases
- Jobs: one-off tasks that run to completion and then stop.
- CronJob: perform regular scheduled actions (creates Jobs)
- ConfigMaps: Store configuration data
- Secrets: Store sensitive information (passwords, certificates, etc)
- PersistentVolumeClaims: Request storage
- PersistentVolumes: Provision storage

Kubernetes Basics #2

- Pod Disruption Budgets (PDB): Limit the number of Pods that can be unavailable at any given time (e.g. upgrades, self-healing during disaster recovery, etc)
- Autoscaler: Automatically scale the number of nodes in a cluster based on workload. Enable optimization of resource utilization and cost.
- Ingress: Routing external traffic to services, SSL termination, load balancing, caching and rate limiting.
    - Kubernetes as a project supports and maintains AWS, GCE, and nginx ingress controllers.
- Operator: software extensions to Kubernetes that make use of custom resources to manage applications and their components.

Plone Components as Kubernetes Resources

- Plone backend (Plone "Classic", Zope)
- Plone frontend (volto, Node.js)
- PostgreSQL (relstorage)
- Varnish (cache)
- nginx (reverse proxy, routing)



- plone6-backend (*StatefulSet*)
- plone6-frontend (Deployment)
- plone6-postgresql(StatefulSet)
- plone6-varnish (StatefulSet)
- plone6-nginx (Deployment)
- plone6-nginx (Ingress)
- plone6-zodbpack (CronJob)

+ configmaps, secrets and pdbs

Kubernetes for Devops

- Kubernetes is known for having a **steep** learning curve, but having a working implementation helps a lot.
- Not long ago, installing a Kubernetes cluster was challenging. Today, we have some lightweight/micro k8s implementations (k3s, k3d, microk8s, etc).
- Using Helm charts (the package manager for k8s) simplifies the installation of scalable, fault tolerant and secure Plone clusters.

Security & Compliance

Container Registry

API Gateway

Scheduling & Orchestration

Security & Compliance

Scheduling & Orchestration

Scheduling & Orchestration

Observability

Security & Compliance

Cloud Native Storage

Container Runtime

Observability

Certified Kubernetes - Distribution

Coordination & Service Discovery

Observability

k3d is a lightweight wrapper to run k3s (Rancher Lab’s minimal Kubernetes distribution) in docker.

```
sudo apt install curl docker.io
sudo snap install kubectl --classic
sudo snap install helm --classic
sudo usermod -a -G docker $(whoami)
curl -s https://raw.githubusercontent.com/k3d-io/k3d/main/install.sh | bash
k3d cluster create mycluster
helm repo add plone https://plone.github.io/helm-charts
helm repo update
kubectl create namespace devsandbox
helm install -n devsandbox plone6 plone/plone --set ingress.enabled=false
```

```
git clone git@github.com:plone/helm-charts.git
cd helm-charts
./test.sh
helm install -n devsandbox plone6 ./plone6-volto-pg-nginx-varnish --dry-run
```

Tips and Tricks #1

- Deployment governance can be managed using ArgoCD (Declarative GitOps Continuous Deployment for Kubernetes)
  - You can use helm chart (with –dry-run option) to produce yamls efficiently
  - Do not push k8s secrets to git! You can use sealed secrets instead.
- Modern cloud-based infrastructure operate better with proper observability (metrics, monitoring, distributed tracing and logs).
  - Grafana, Prometheus, Thanos, Fluentd, Loki, OpenTelemetry, Jaeger are all great projects
  - Plone can do better on integrating with some of these technologies
- Use ZODB with PostgreSQL and relstorage
- Varnish Operator (from IBM) looks promising

Tips and Tricks #2

- Run Plone Helm chart "*test.sh*" may use a lot of bandwidth downloading docked images Eventually, docker hub rate limit will impact you.
- k3d supports docker images import (so k8s don't need to download images)

```
$ cat images.txt

plone/plone-backend:latest

plone/plone-frontend:latest

nginx:latest

postgres:16

varnish:stable


for i in $(cat images.txt); do

 docker pull $i

 k3d image import $i -c mycluster

done
```

Demo

Thank you

Fabiano Weimar [Xiru]
xiru@xiru.org