DEEPSHORE

Kubernetes Essentials

The Kubernetes API

Agenda

Introduction

Interaction

Extensions

INTRODUCTION

Why this talk refers to Plain Kubernetes (Vanilla)

- OpenShift: Enterprise **Kubernetes** platform
- Basic principles apply to both OCP and Kubernetes Vanilla
- Kubernetes Vanilla is easier to deploy on a local machine

Minikube

- Setting up local Kubernetes cluster
- Windows, Linux, MacOS
- VMs, containers, bare-metal
- Multiple container runtimes
- Provides Addons
- Support of the latest Kubernetes release

Kubernetes Components

https://kubernetes.io/docs/concepts/overview/components/

Control Plane

- kube-apiserver: enables management of Kubernetes objects by clients
- etcd: stores all cluster data
- kube-controller-manager: runs (built-in) controller processes
- cloud-controller-manager (optional): runs controllers specific to cloud provider
- kube-scheduler: decides on which node pods should run

(Worker) Nodes

- kubelet: runs containers
- kube-proxy: maintains network rules

INTERACTION

Facts about the kube-apiserver

- REST-API
- HTTP
- External and internal requests:
  - kubectl/oc
  - components of the Kubernetes cluster
  - everything that can send HTTP requests, e.g. curl, Browser, …
- TLS support

kubectl

- CLI that „lets you control Kubernetes clusters“
- common way to interact with k8s clusters
- almost interchangeable with oc
- kubeconfig: information how to connect with clusters
- under the hood: it‘s all about generating HTTP request
- Syntax: kubectl [command] [TYPE] [NAME] [flags]
- Examples:
  - kubectl create configmap my-configmap –-from-literal=key=value (creating a configmap)
  - kubectl proxy (runs a proxy to the apiserver)

Kubernetes Resources

Kubernetes Resources can be accessed by Uniform Resource Locators (URL) → HTTP addresses

Resources

- are of a certain type, e.g. namespace, pod, service, …
- are representing objects (instances of a concept on the cluster)
- are either cluster-scoped or namespace-scoped

Resources types are grouped by API groups:

- Core Group: /api/v1
- Named groups: /apis/$GROUP_NAME/$VERSION

Manifests (yaml)

- yaml file including the complete information of a resource (object)
- can be used for creating, editing and deleting objects

Fields:

- apiVersion: API Group and version
- kind: type of an entity
- metadata: data that helps identify the object
- spec: desired state of the object

Example: manifests/pod.yaml

How to create manifests effectively

… by running kubectl commands with the **--dry-run=client** flag

Controllers

Spec field: desired state

Status field: current state, updated by components of the Kubernetes system

Controllers take actions to push the current state towards the desired state

Example: replicas in a Deployment controlled by the Deployment controller

DEEPSHORE

EXTENSIONS

How to extend the API

Steps:

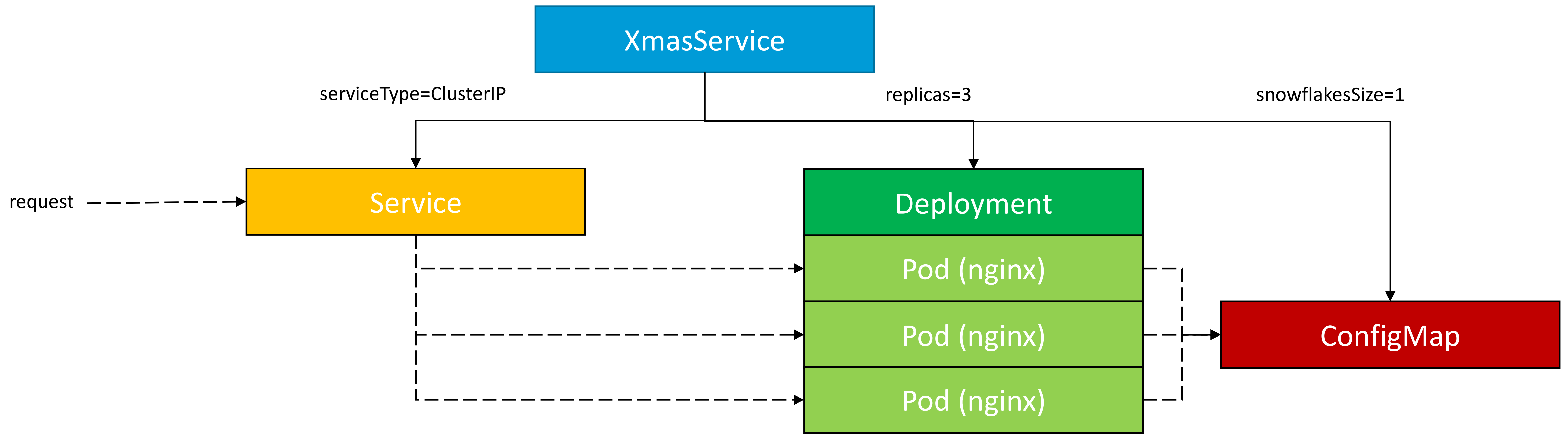
- Create a CustomResourceDefinition (CRD)
- Deploy a custom controller (operator)

→ „Operator Pattern“

Usage of the API Extension:

Create Custom Resources

Extending the API by the XmasService Operator



Common Operators

- Confluent Operator for Kafka
- Strimzi Kafka Operators
- Prometheus: Metrics
- Velero: Backup and Restore
- ArgoCD: Continuous Delivery (GitOps)
- Knative: Serverless Applications
- Istio: Service Meshes
- ...

Operator Frameworks

- Shell-Operator by Flant
- Operator-SDK
- Kubebuilder
- Kopf (Kubernetes Operator Pythonic Framework)

https://kubernetes.io/docs/concepts/extend-kubernetes/operator/# writing-operator

Material related to this talk

- Github: https://github.com/deepshore/kubernetes-essentials
- Kubernetes Docs: https://kubernetes.io/docs/home/
- Operator Pattern on Yt: https://www.youtube.com/watch?v=K_rTn3DaBgO

DEEPSHORE

THANKS

Kubernetes API Terminology

Kubernetes Resources can be accessed by Uniform Resource Locators (URL) → HTTP address

Resource type
name used in the URL (pods, namespaces, services)

Kind
concrete representation (object schema) of a resource type

Object
concrete instance of a concept on the cluster

Resource
instance of a resource type usually representing an object

Collection
list of instances of a resource type