ORDINA

Kubernetes Workshop

Competence Center

Cloud Platforms

Tom Verelst

Agenda

- Introduction to Kubernetes
- Deploying to Kubernetes
- Service Discovery and Load Balancing
- Storage
- Exposing services
- Secret and configuration management
- Exercises

Kubernetes Introduction

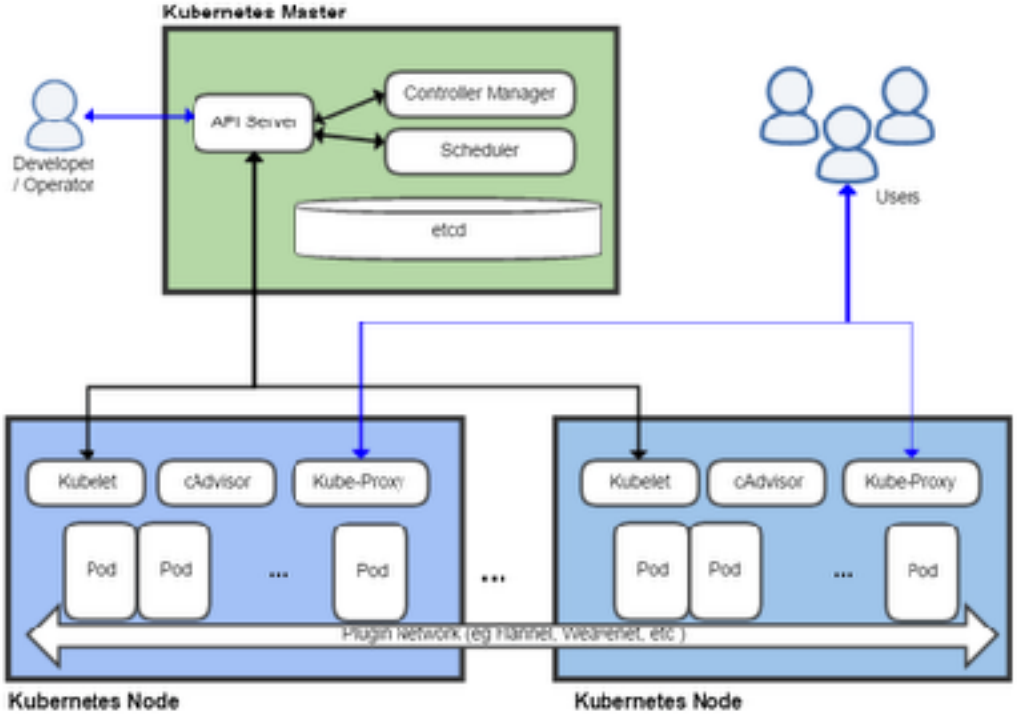
What is Kubernetes?

- Container Orchestration Platform
- Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.
- Part of the Cloud Native Computing Foundation https://www.cncf.io/

Features

- Automatic binpacking
- Horizontal scaling
- Automated rollouts and rollbacks
- Storage orchestration
- Self-healing
- Service discovery and load-balancing
- Secret and configuration management
- Batch execution

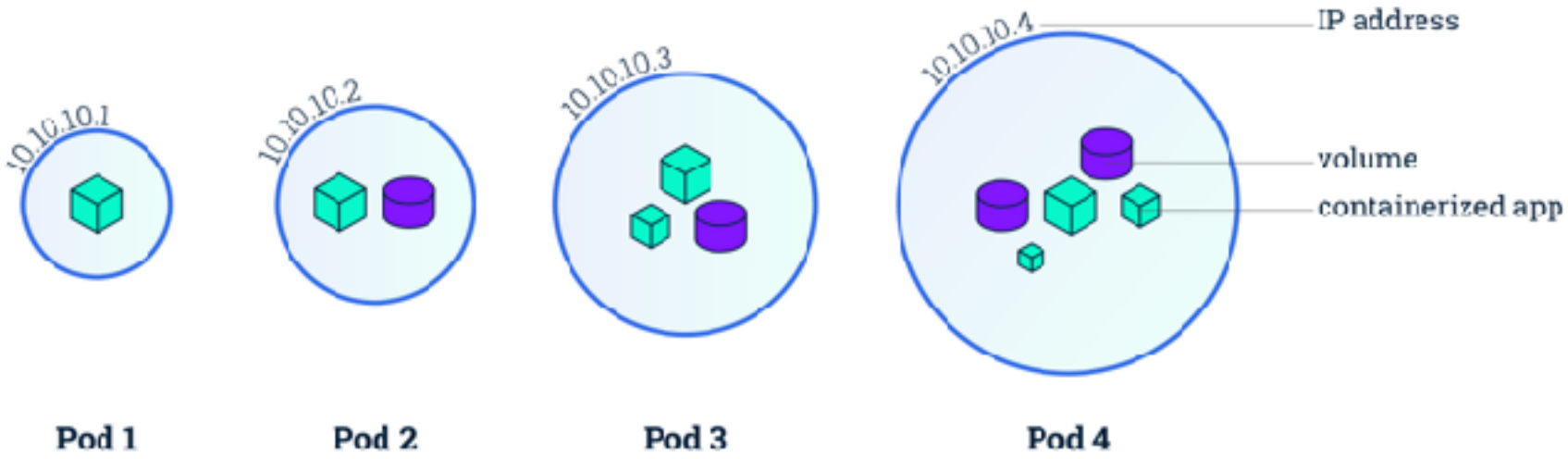
Kubernetes Architecture



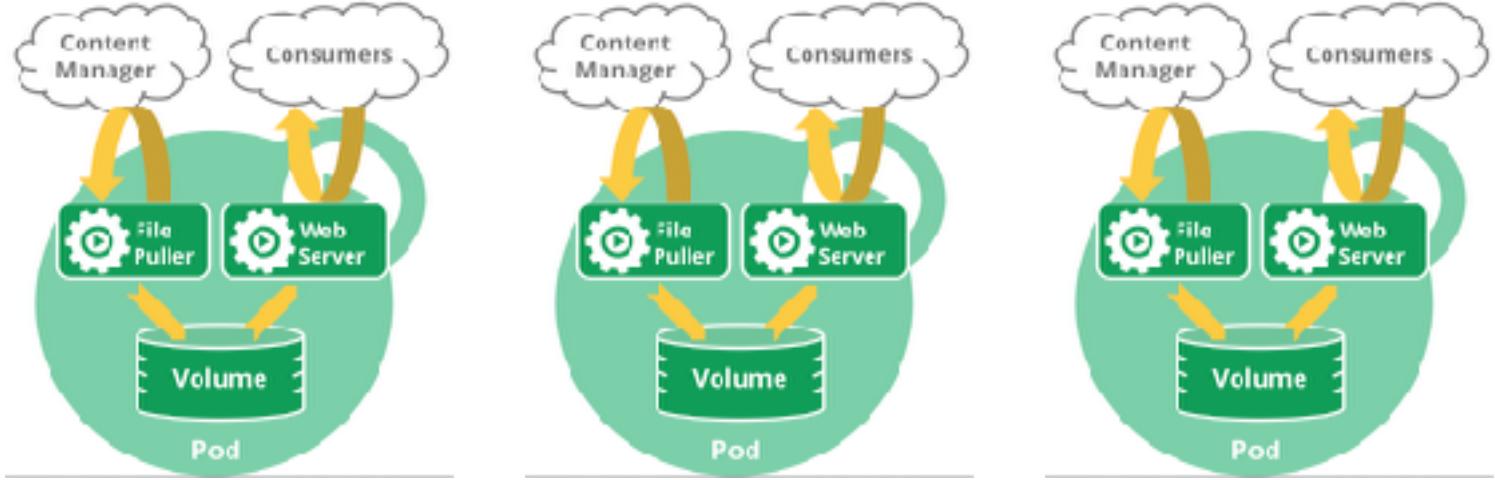
Pods

- Containers within the same pod
- Containers can find each other with `localhost`
- Each pod has an IP address
- Volumes are shared within a pod
- Usually only **one** container
- Multi-container pods only with monitoring/logging agents/ …
- **Do not combine application + database in one pod**

Pods

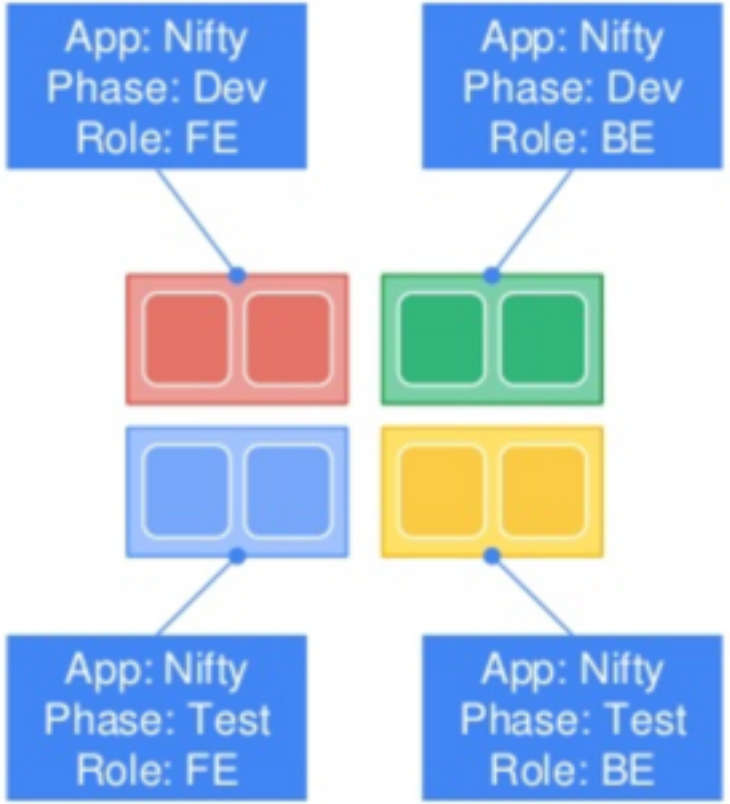

10.10.10.1
10.10.10.2
10.10.10.3
10.10.10.4
IP address
volume
containerized app
Pod 1
Pod 2
Pod 3
Pod 4

Pods


Content Manager
Consumers
File Puller
Web Server
Volume
Pod
Content Manager
Consumers
File Puller
Web Server
Volume
Pod
Content Manager
Consumers
File Puller
Web Server
Volume
Pod

Labels

- Arbitrary **Metadata**
- Attached to any **Resource**
- Represents **Identity**
- Queryable by **Selectors**
- The only **grouping mechanism**



Selectors

- Equity-based selector

```
environment=production,tier=frontend
```

- Set-based selector

```
environment in (production),tier in (frontend)
```

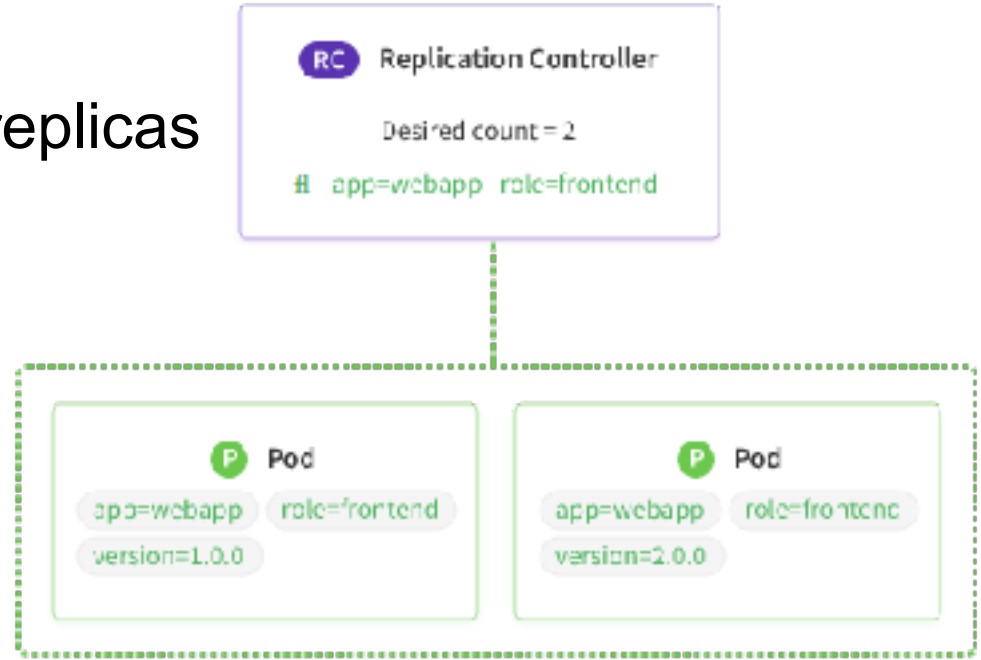
Deploying to Kubernetes

Deploying Pods

- Replication Controller
- Replica Set
- Deployment
- Daemon Sets

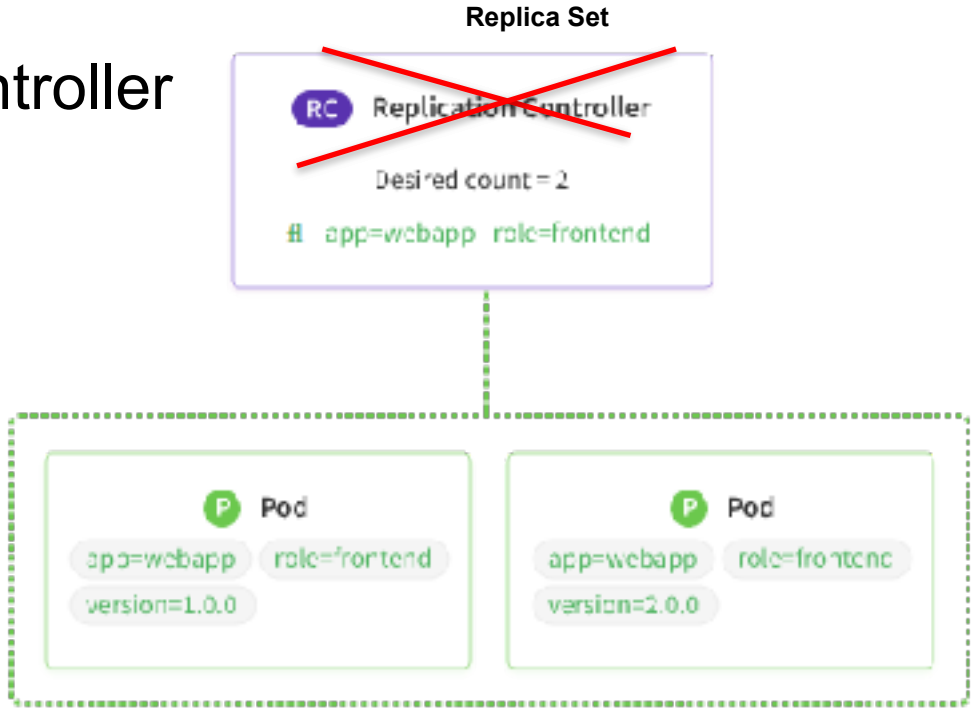
Replication Controller

- Desired state
- Controls the amount of replicas



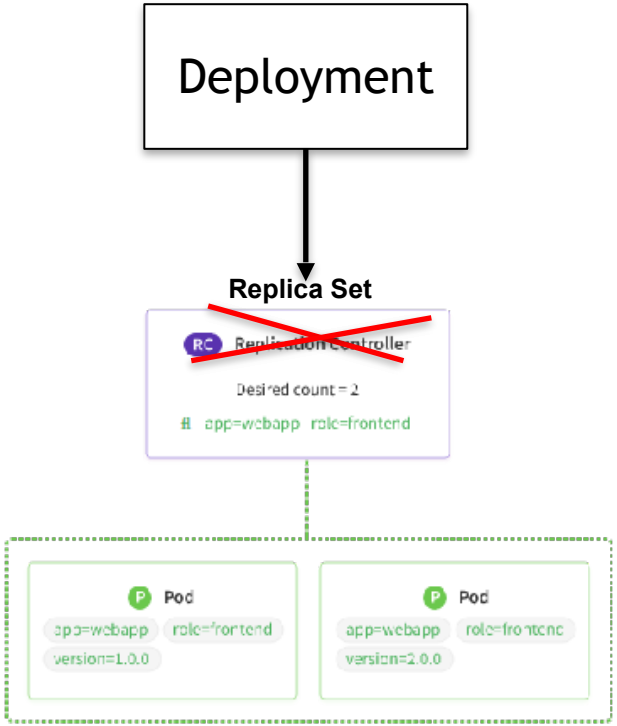
Replica Set

- Same as ReplicationController
- Supports the new **Set-based** selectors

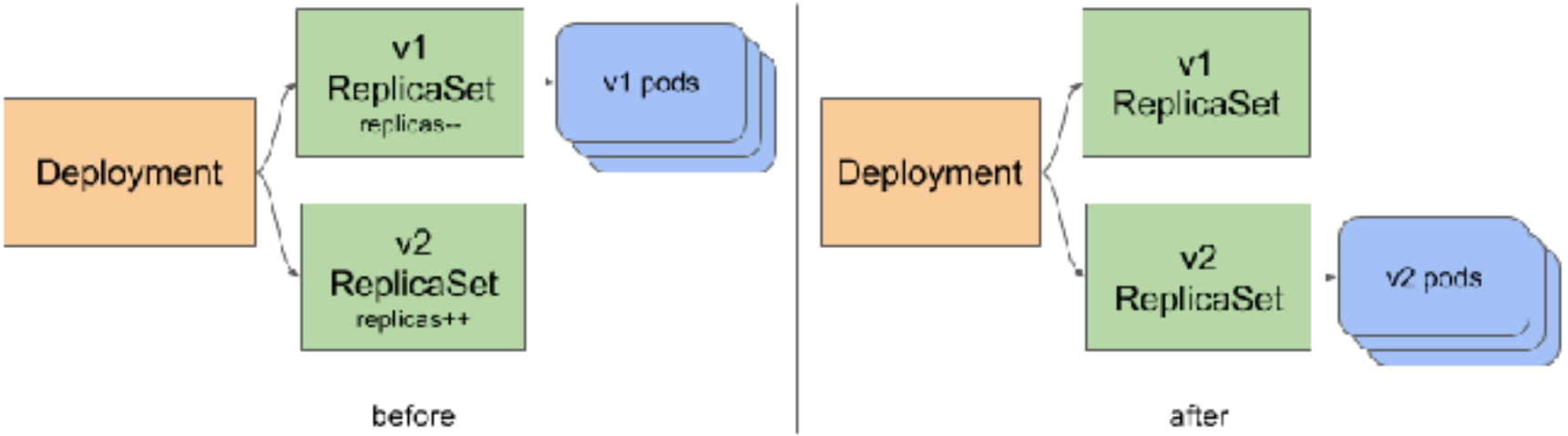


Deployment

- Common way of deploying
- Manages the ReplicaSets for you
- Supports **rolling updates**



Rolling Updates



Daemon Set

- Runs a pod on every node
  - Or a select few of them
- Not a fixed number of replicas
  - Depends on the amount of selected nodes
- Useful for cluster-wide services
  - E.g. logging or monitoring agents

Local Setup

Local Setup

- https://https://github.com/tomverelst/kickstarter-kubernetes
- kubectl
- Minikube (requires a VM like VirtualBox)

minikube

Kubectl

- Command line tool that talks to the Kubernetes API server
  https://kubernetes.io/docs/tasks/tools/install-kubectl/
- macOS

```
$ brew install kubectl
```

- Windows

```
$ choco install kubernetes-cli
```

- Config is in `~/.kube/config`

- A local Kubernetes cluster
- https://github.com/kubernetes/minikube/releases
- Generates kubectl config

```
$ minikube version
minikube version: v0.23.0
$ minikube start --kubernetes-version v1.10.0
```

minikube

New release!

v0.24.0

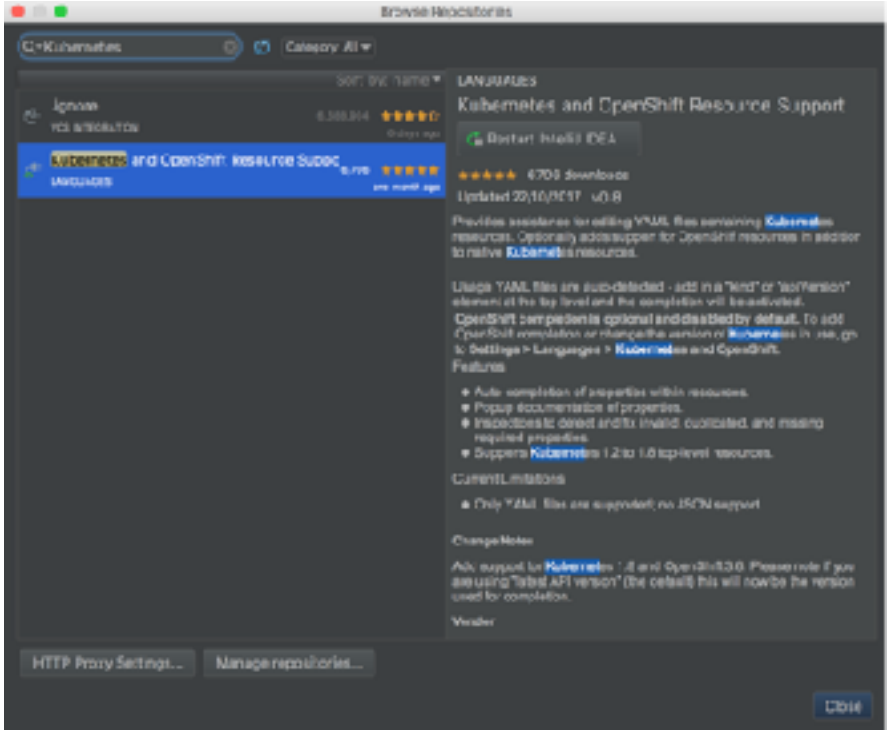
minikube-bot released this 11 hours ago

New Drivers

This release also includes official support for two new drivers, `hyperkit` and `kvm2`. These drivers are intended to replace the out-of-tree drivers `xhyve` and `kvm`, respectively.

minikube

IntelliJ plugin



Minikube commands

```
# Start cluster
$ minikube start --kubernetes-version v1.10.0
  --memory 4096
# Delete cluster
$ minikube delete
# Get IP from the cluster
$ minikube ip
# Open Kubernetes Dashboard
$ minikube dashboard
```

Kubernetes Objects

- Everything is a **Kubernetes Object** (**API Object)**
- Always has a **spec** and a **status**
- Defined in **.yaml** files

Kubectl commands

```
$ kubectl get <resource type>
$ kubectl get pods
$ kubectl get services
$ kubectl get svc
```

Kubectl commands

```
$ kubectl create -f file.yaml
$ kubectl apply -f file.yaml
$ kubectl delete -f file.yaml
$ kubectl delete pod <pod-name>

$ kubectl apply -f afolder
$ kubectl delete -f afolder
```

Kubectl commands

```
$ kubectl logs <pod name>
$ kubectl describe pod <pod name>
$ kubectl describe svc <service name>
```

```
$ kubectl proxy (minikube dashboard)
```

http://localhost:8001/api/v1/proxy/namespaces/kube-system/services/kubernetes-dashboard/#/namespace?namespace=default

- Make a pod accessible locally!

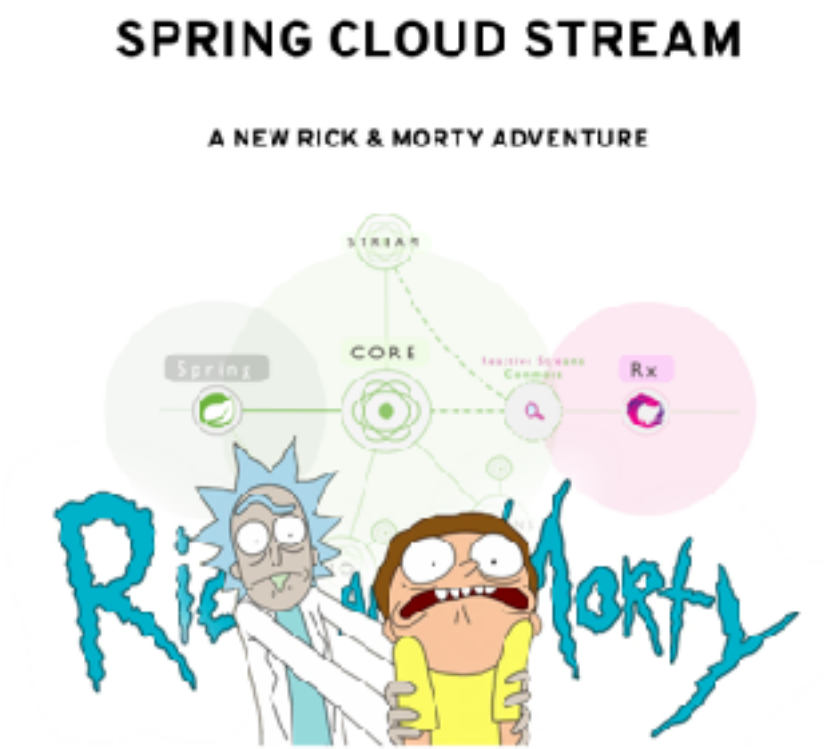
```
$ kubectl port-forward <pod-name> <port>
```

Demo Application

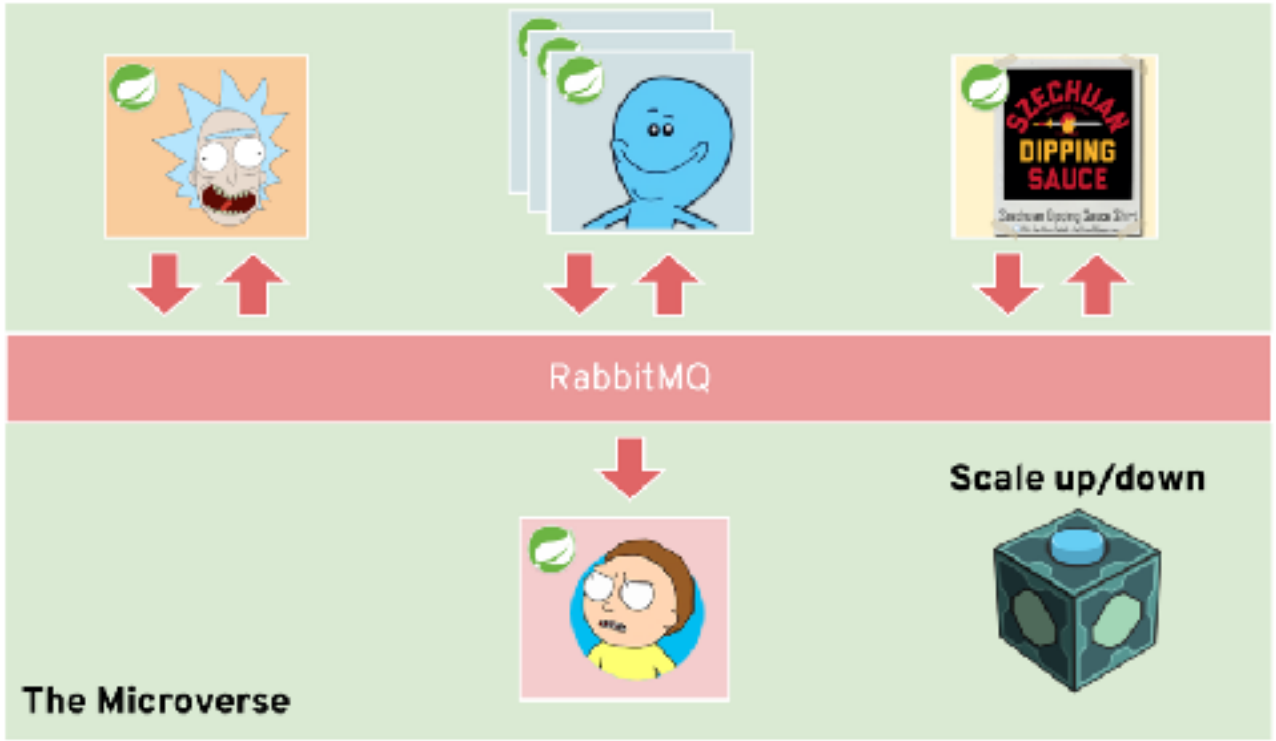
https://github.com/tomverelst/kickstarter-kubernetes

Demo Application

- A Rick & Morty adventure!
- Originally for CloudFoundry
- Ported to Kubernetes
- Credits to **Dieter Hubau**



Architecture

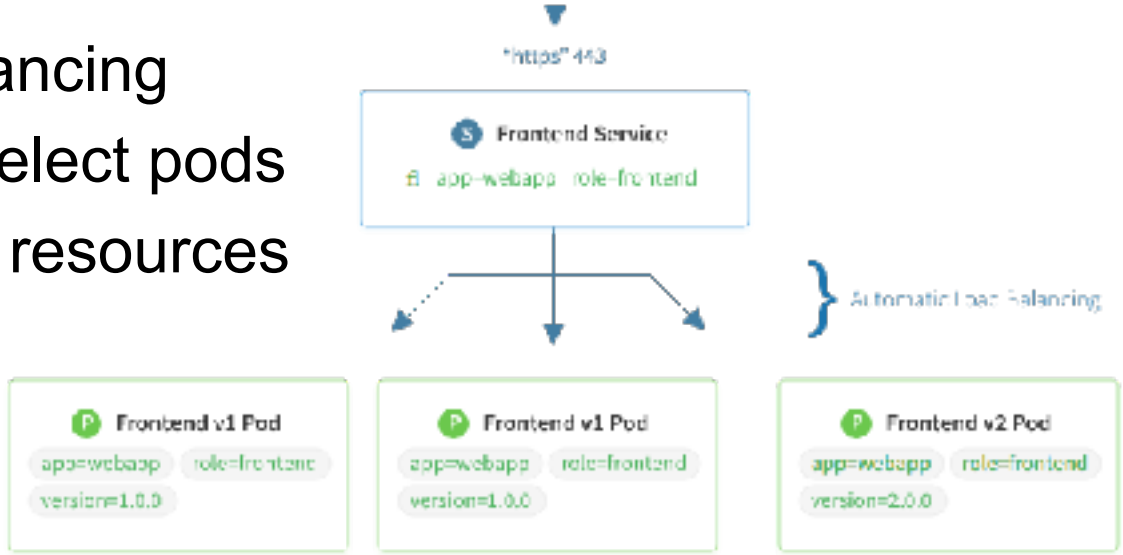


Let’s deploy!

Service Discovery & Load Balancing

Service

- Service discovery
- Automatic load balancing
- Uses selectors to select pods
- Creates “Endpoint” resources



Service Types

- ClusterIP (default)
  - Exposes the service on a cluster-internal IP.
- Load Balancer
  - External load balancer
- NodePort
  - Expose the service on each node with a randomly allocated port
- ExternalName
  - Maps to an external service

Name spaces

- Virtual clusters
- Default namespace is **`default`**
- Divide cluster resources between users/teams

Service Discovery through

- Each service gets assigned a DNS name
- A service named **`foo`** in the **`default`** namespace is discovable through DNS as **`foo`**

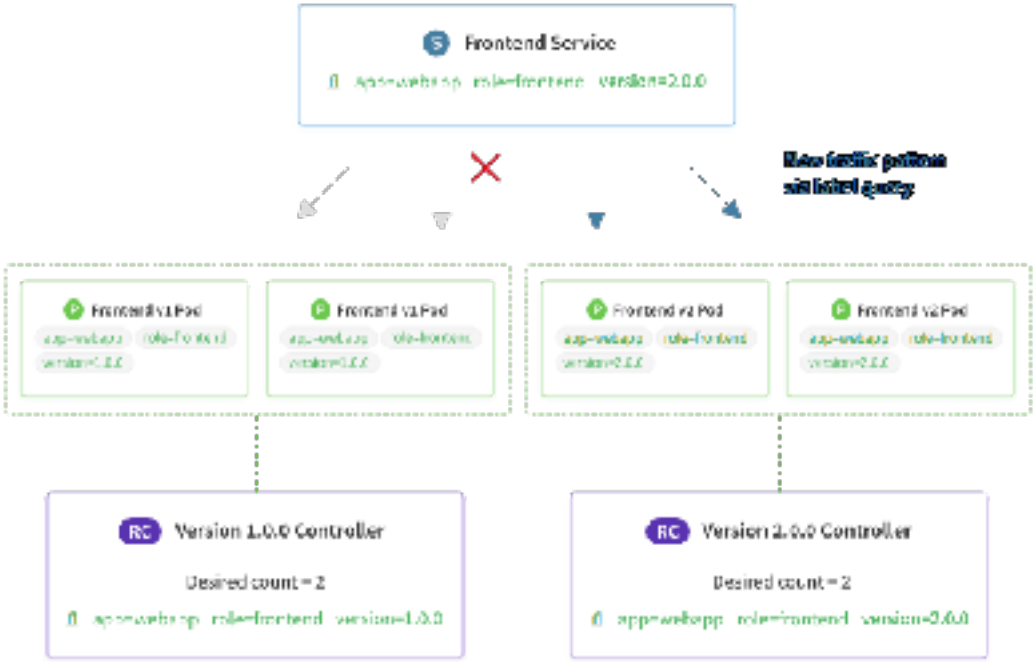
  A Record:
  `foo.default.svc.cluster.local`

Service Discovery through

- A service named **`foo`** in the **`bar`** namespace is discoverable through DNS as **`foo.bar`**

  A Record
  `foo.bar.svc.cluster.local`

Replication + Services



Let’s create our services!

Storage

Persisting Data

- Persistent Volumes
- Persistent Volume Claims
- Storage Class
- Dynamic Volume Provisioning

Persistent Volume

- Defines a mountable volume
  - Local disk
  - EBS
  - Google Cloud Disk
  - …
- Admin / OPS owned

Access Modes

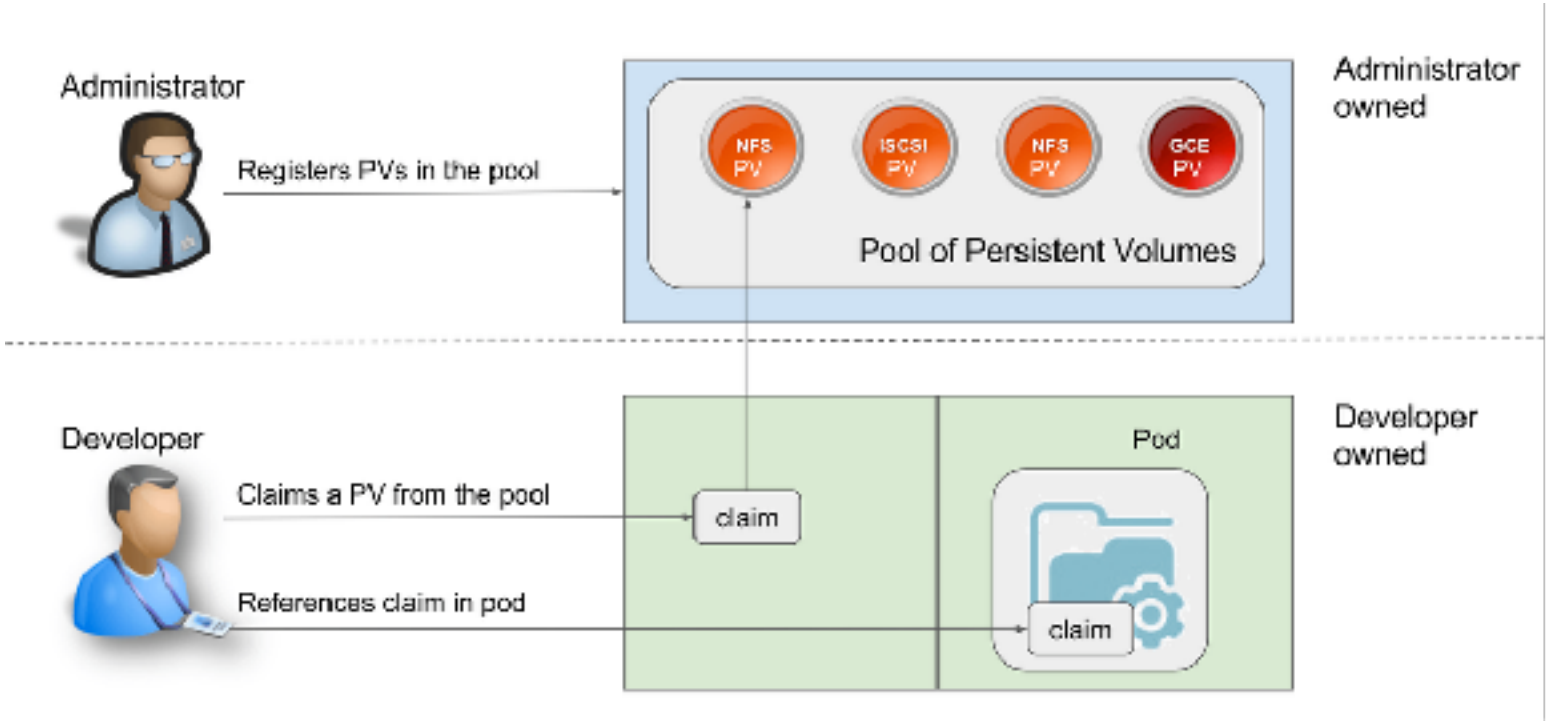
- ReadWriteOnce
  - the volume can be mounted as read-write by a single node
- ReadOnlyMany
  - the volume can be mounted read-only by many nodes
- ReadWriteMany
  - the volume can be mounted as read-write by many nodes

Persistent Volume Claim

- Claims a Persistent Volume using a selector for a pod
- After claiming, the pod can access the volume
- If no PV can be claimed, the PVC will be “pending”

- Developer owned

Claiming a Persistent Volume



Storage Class

- Admins can create profiles of the storage they offer
- Extra abstraction between volumes and claims

Storage Class

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: slow
provisioner: kubernetes.io/aws-ebs
parameters:
  type: io1
  zones: us-east-1d, us-east-1c
  iopsPerGB: "10"
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: claim1
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: fast
  resources:
    requests:
      storage: 30Gi
```

Dynamic Volume

- Allows volumes to be created on-demand
- Requires Storage Classes with a provisioner

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: slow
provisioner: kubernetes.io/gce-pd
parameters:
  type: pd-standard
```

Let’s persist our data!

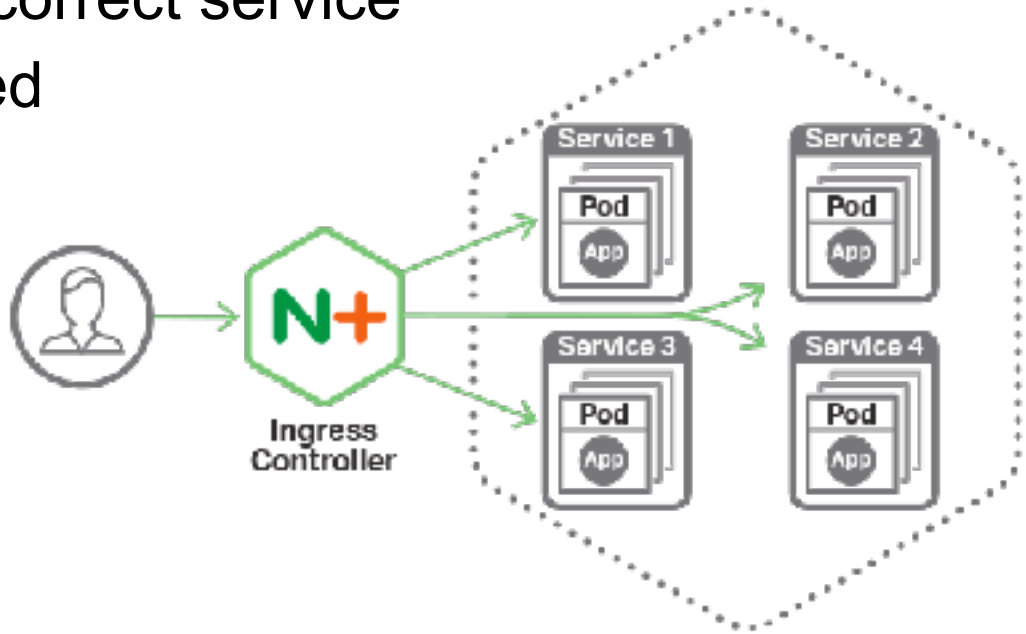
Exposing services

Exposing Services

- Ingress Controller
- Ingresses

Ingress Controller

- Routes traffic to the correct service
- Automatically updated
- nginx, HAProxy, …
- Configure TLS



Let’s expose our services!

Setup Ingress Controller

- `$ minikube addons enable ingress`
  - Will create an nginx Ingress controller

Annotations

- Similar to **Labels**
- (Arbitrary) **Metadata**
- Attached to any **Resource**
- **Not queryable by Selectors**
- Intended for configuration, build information, pointers for logging/monitoring agents, etc..

Secret & configuration management

Secret & Configuration

- Secret
- ConfigMap

Secret

- Holds sensitive information like passwords, OAuth tokens, ssh keys, …
- Secrets are **base64** encoded
- Can be **mounted** or consumed through an **environment variables**

Create secret

```
$ kubectl create secret generic db-user-pass --from-file=./username.txt --fron-file=./password.txt
secret "db-user-pass" created
```

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

```
$ kubectl create -f ./secret.yaml
secret "mysecret" created
```

Mount Secret

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - name: mypod
    image: redis
    volumeMounts:
    - name: foo
      mountPath: "/etc/foo"
      readOnly: true
  volumes:
  - name: foo
    secret:
      secretName: mysecret
```

Secrets in Environment

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: secret-env-pod
spec:
  containers:
  - name: mycontainer
    image: redis
    env:
      - name: SECRET_USERNAME
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: username
      - name: SECRET_PASSWORD
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: password
  restartPolicy: Never
```

ConfigMap

- Decouples configuration from images
- Can contain files (`--from-file`)

```
kubectl create configmap game-config --from-file=docs/user-guide/configmap/kubectl
```

- Can contain key value pairs (`--from-literal`)

```
kubectl create configmap special-config --from-literal=special.how=very
```

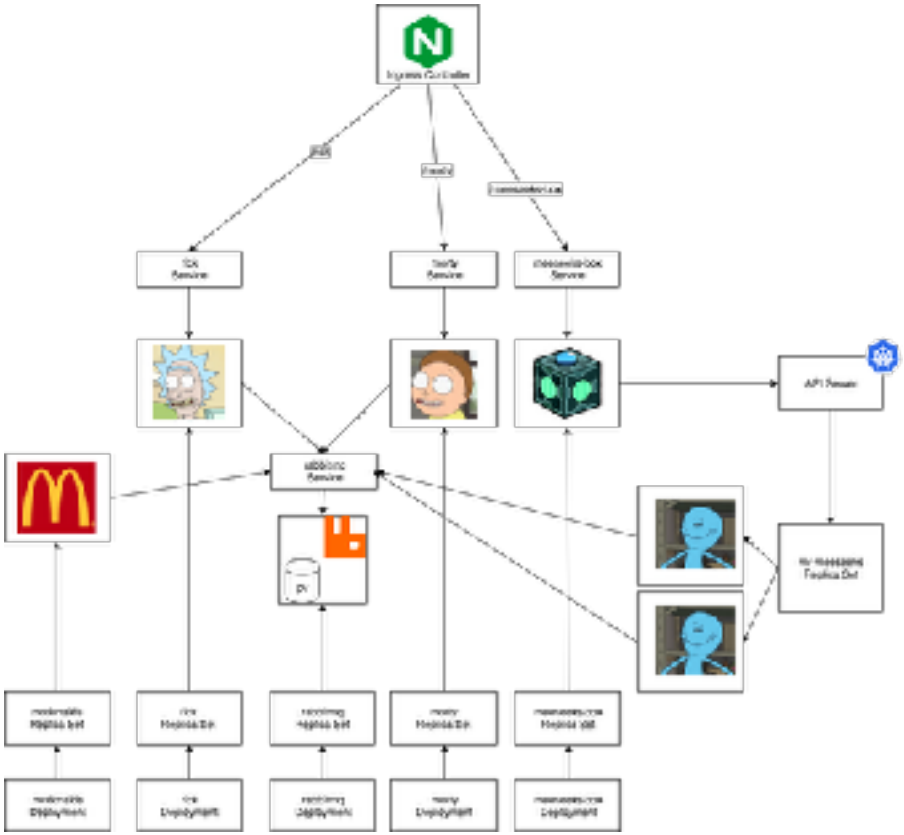
Environment Variables

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dapi-test-pod
spec:
  containers:
    - name: test-container
      image: gcr.io/google_containers/busybox
      command: [ "/bin/sh", "-c", "env" ]
      env:
        - name: SPECIAL_LEVEL_KEY
          valueFrom:
            configMapKeyRef:
              name: special-config
              key: special.how
        - name: LOG_LEVEL
          valueFrom:
            configMapKeyRef:
              name: env-config
              key: log_level
  restartPolicy: Never
```

```yaml
env:
  - name: SPECIAL_LEVEL_KEY
    valueFrom:
      configMapKeyRef:
        name: special-config
        key: special.how
```

Let’s configure!

Final Result



Extra

- Helm - A package manager for Kubernetes
  https://github.com/kubernetes/helm
  https://github.com/kubernetes/charts

- Draft - Containerizes applications for Kubernetes using build packs (à la CloudFoundry), uses helm, in beta
  https://github.com/azure/draft

- kube-lego - Automatically request certificates for Kubernetes Ingress resources from Let's Encrypt
  https://github.com/jetstack/kube-lego

-

Questions?

Thanks!