Workshop

# Kubernetes

## For Software Developers


Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com

Mail rainer@timecockpit.com

Twitter @rstropek


time cockpit

Saves the day.

# Your Host


## Rainer Stropek

Developer, Entrepreneur
Azure MVP, MS Regional Director
IT-Visions

## Contact

software architects gmbh
rainer@timecockpit.com
Twitter: @rstropek


# What's Kubernetes?

# Feature View

## Platform for containerized workloads and services

Open Source (open-sourced in 2014 by Google)

## Portable cloud platform

Microservices
Tries to combine best of PaaS and IaaS

## Current state → desired state

## Container-centric

OS-level virtualization instead of HW-virtualization
Here: Focus on Docker

https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

# Component View

## Master (aka Control Plane)

API Server
etcd
Scheduler
Controller Manager (Node-, Replication-, Endpoints-, and Service Accounts-Controller)

## Nodes

Kubelet (primary "node agent" ensuring that pods are running and healthy)
Network Proxy
Container Runtime (in our case *Docker*)
Addons
DNS
Dashboard (web UI)
Monitoring and Logging

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

# API

## Web API

Documented with OpenAPI

Versioned

## *kubectl*

CLI for interacting with API

## Base for declarative configuration schema

https://kubernetes.io/docs/concepts/overview/kubernetes-api/

```
GET http://127.0.0.1:8001/api/v1/nodes
Accept: application/json

###
GET http://127.0.0.1:8001/api/v1/namespaces/default/pods

###
POST http://127.0.0.1:8001/api/v1/namespaces/default/pods
Content-Type: application/json

{
    "apiVersion": "v1",
    "kind": "Pod",
    "metadata": {
        "name": "api-demo-pod",
        "labels": {
            "app": "api-demo"
        }
    },
    "spec": {
        "containers": [
            {
                "name": "api-demo-pod",
                "image": "nginx:alpine",
                "ports": [
                    { "containerPort": 80 }
                ]
            }
        ]
    }
}

###
GET http://127.0.0.1:8001/api/v1/namespaces/default/pods?
      labelSelector=app%20in%20%28api-demo%29

###
DELETE http://127.0.0.1:8001/api/v1/namespaces/default/pods/api-demo-pod
```

# Demo: API

API Docs

- Read Nodes
- Create Pod
- Delete Pod

Label Selectors

https://www.json2yaml.com

Tip: Run *kubectl proxy* for authenticated access from localhost

https://github.com/rstropek/DockerVS2015Intro/blob/master/dockerDemos/13-kube-intro/api-demo.http

# Kubernetes 101

## Pods

Typically one, sometimes multiple containers (tightly-coupled)
Common Pod environment (e.g. IP address, *localhost*)
Atomic unit (all-or-nothing)
Tip: Don't create pods directly, use controllers (e.g. deployments)

## Service

Each services gets its own stable IP address, DNS name, and port
Uses *labels* to dynamically associate with Pods
Load-balances requests across Pods
Service type *LoadBalancer* → make service available outside the cluster

# Kubernetes 101

## ReplicaSet (aka ReplicationController)

Number of replicas for a Pod
*Desired State* and *Current State*
Tip: Don't create manually, use deployments

## Deployments

Replication controller + *rolling updates*

## Other Workload (not covered in detail here)

CronJob
DaemonSet (run copy of a pod on every node)
Jobs
StatefulSets (like deployments, but optimized for stateful applications)

# Labels & Selectors

Key/value pairs for K8s objects

Organize and select objects

Used to define relationships of objects
E.g. service → pod

Label Selectors

https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/

# *kubectl* Fundamentals

```
kubectl [command] [TYPE] [NAME] [flags]
```

- Command options
- Resource name
- Resource type (full name or abbreviation)
  List of resource types…
- Operation; e.g. create, get, delete;
  List of operations…

```
Kubectl get all
kubectl get pods

# kubectl configuration

kubectl config get-contexts
kubectl config set-context …

kubectl cluster-info
kubectl get nodes
kubectl describe nodes
kubectl get componentstatuses # or short “cs”
```

# kubectl

Different output formats
E.g. JSON, YAML, tables
Details…

Cheat sheet…

You know Docker?
*kubectl* for Docker users…

https://kubernetes.io/docs/reference/kubectl/overview/

```
kubectl get namespace

kubectl create namespace …

kubectl config set-context --current --namespace=…
```

# kubectl

Different output formats

E.g. JSON, YAML, tables

Details…

Cheat sheet…

You know Docker?

*kubectl* for Docker users…

https://kubernetes.io/docs/reference/kubectl/overview/

# Object Management

## Imperative commands

Manipulate live objects

No source control, no review process → development, not production

## Imperative object configuration

Configuration files (YAML)

Individual files

## Declarative object configuration

Directories with configuration files

```
# Create deployment
kubectl run nginx --image=nginx:alpine --port=80
kubectl get deployments
kubectl get pods

# Interactive Pod
kubectl run my-shell --rm –it --image debian -- bash

# Scale
kubectl scale deployment nginx --replicas=3

# Work with deployment’s labels
kubectl describe deployment nginx
kubectl label deployment nginx env=testing
kubectl describe deployment nginx
kubectl get deployment -l env=testing -o yaml

# Create service
kubectl expose deployment nginx --type=NodePort
kubectl get services
# Open http://127.0.0.1:<port>

# Try Kubernetes Dashboard
kubectl proxy
# Open http://localhost:8001/ui for Dashboard

# Delete deployment and service
kubectl delete deployment nginx
kubectl delete service nginx
```

# Imperative Commands

## Try this exercise in…

…Docker for Windows

…AKS

## Try different service types

NodePort

# Kubernetes Workloads

```
kind: Pod
apiVersion: v1
metadata:
  name: nginx-pod
  labels:
    name: nginx-pod
    app: workshop
spec:
  containers:
  - name: nginx-pod
    image: nginx
    ports:
      - containerPort: 80
        name: http
        protocol: TCP

###

kubectl create -f nginx-pod.yaml
kubectl get pods
kubectl expose pod nginx-pod --type NodePort
kubectl get services
kubectl exec -it nginx-pod -- /bin/bash
```

# Pods

Tipp: Kubernetes Snippets in VSCode

https://kubernetes.io/docs/concepts/overview/object-management-kubectl/imperative-config/

```
kind: Service
apiVersion: v1
metadata:
  name: web-svc
spec:
  selector:
    app: workshop
  ports:
  - port: 8080
    targetPort: 80
  type: NodePort

###

kubectl create -f .\nginx-pod.yaml
kubectl create -f .\nginx-service.yaml
kubectl get services
kubectl delete -f .\nginx-service.yaml
kubectl delete -f .\nginx-pod.yaml
```

# Services

## Service Types

- ClusterIP
- NodePort
- LoadBalancer (in AKS)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web
spec:
  replicas: 5
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
      - name: web
        image: nginx:alpine
        ports:
        - containerPort: 80

###

kubectl apply -f .
```

# Deployment

## Scaling

Change number of replicas and run “apply” again

## Scale cluster in AKS

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: demo-web
spec:
  selector:
    matchLabels:
      app: demo-web
  replicas: 5
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  minReadySeconds: 30
  template:
    metadata:
      labels:
        app: demo-web
    spec:
      containers:
      - name: demo-web
        image: rstropek/docker-image-versioning:1.0
        ports:
          - containerPort: 80
        env:
          - name: PORT
            value: "80"

###

kubectl apply -f .
kubectl get pods
Kubectl describe demo-web

kubectl set image deployment/demo-web demo-web=rstropek/docker-image-versioning:2.0
kubectl rollout status deployment/demo-web
kubectl get deployment demo-web

kubectl rollout history deployment demo-web
kubectl rollout history deployment demo-web --revision=2
kubectl rollout undo deployment demo-web
```

# Rolling Updates

# Storage

# Storage

## Volumes

Volume's lifetime bound to Pod's lifetime
Different types of volumes including cloud (Azure, Google, etc.), path on host, GitHub, etc.
More…

## *PersistentVolume* (PV)

Similar to a Node
Admin provides storage with certain properties (capacity, performance, etc.)
Different types of volumes including cloud (Azure, Google, etc.), path on host, etc.

## *PersistentVolumeClaim* (PVC)

Similar to Pod
Request for storage (PV) by a user

https://kubernetes.io/docs/concepts/storage/

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: host-path
  labels:
    app: host-path
spec:
  containers:
  - image: nginx
    name: host-path
    volumeMounts:
    - mountPath: /usr/share/nginx/html
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      path: /host_mnt/c/temp/kubernetes-data
      type: Directory

---
apiVersion: v1
kind: Service
metadata:
  name: host-path-svc
spec:
  selector:
    app: host-path
  ports:
  - port: 80
    targetPort: 80
  type: NodePort
```

# Volumes

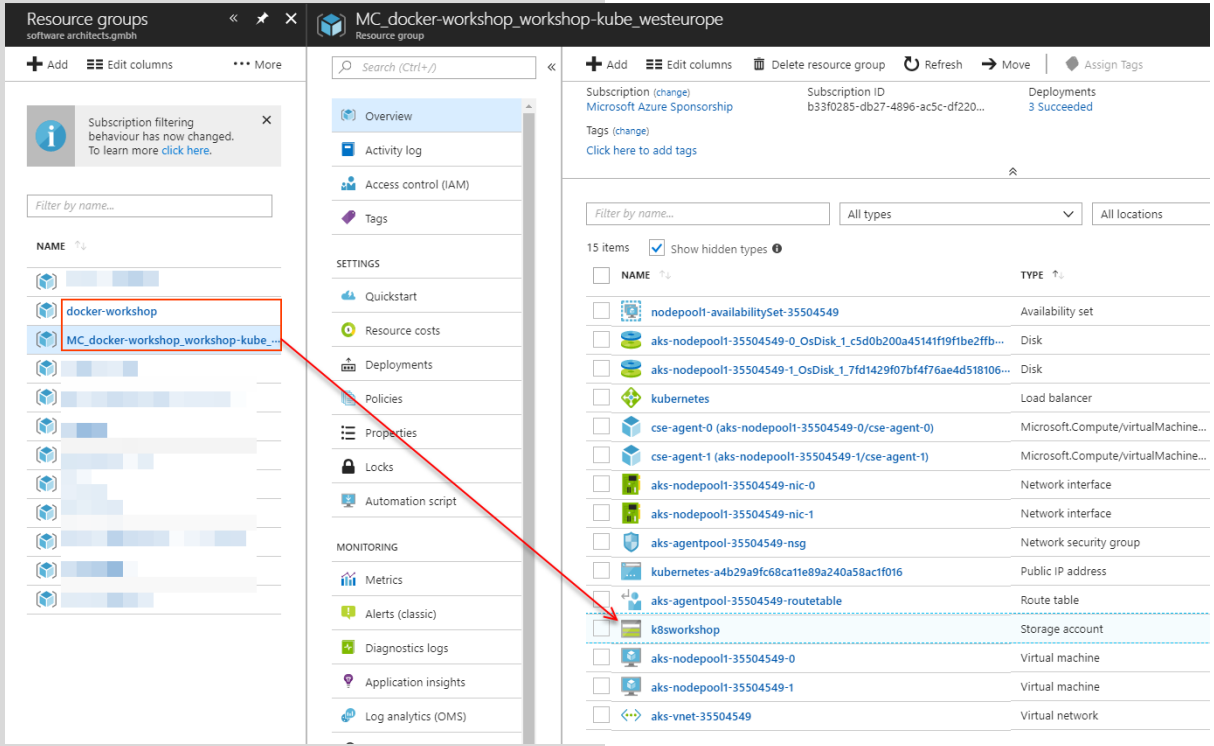
Docker for Windows

## Mount Windows folder

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: k8sworkshop-files
provisioner: kubernetes.io/azure-file
parameters:
  storageAccount: k8sworkshop
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: k8sworkshop-pvc
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: k8sworkshop-files
  resources:
    requests:
      storage: 5Gi
```

# PV, PVC

## With Azure



```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: azure-managed-disk
spec:
  accessModes:
  - ReadWriteOnce
  storageClassName: default
  resources:
    requests:
      storage: 5Gi
---
kind: Pod
apiVersion: v1
metadata:
  name: nginx
spec:
  containers:
    - name: mywebserver
      image: nginx
      volumeMounts:
      - mountPath: "/usr/share/nginx/html"
        name: volume
  volumes:
    - name: volume
      persistentVolumeClaim:
        claimName: azure-managed-disk
```

# PV, PVC

With Azure

## Mount Windows folder

# AKS

## Azure Container Service – Kubernetes

# AKS

## *Managed* Kubernetes in Azure

No direct access (SSH) to the cluster

## Note preview limitations

At the time of writing, no deployments possible in West Europe

Read more…

https://docs.microsoft.com/en-us/azure/aks/intro-kubernetes

```
# Note: Uses demo app from
# https://github.com/Azure-Samples/azure-voting-app-redis.git

cd ~/GitHub/azure-voting-app-redis
cat cat docker-compose.yaml

docker-compose up -d

# Try app at http://your-ip:8080

docker images
docker ps

docker-compose stop
docker-compose down
```

# Demo

Prepare image

Create images locally

Test application locally

https://docs.microsoft.com/en-us/azure/aks/kubernetes-walkthrough

```
az provider register -n Microsoft.ContainerService

# Create service principal for AKS (or change password)
# This is necessary if you want to use ACR with AKS
az ad sp create-for-rbac --name azurecaas \
  --password P@ssw0rd123
az ad sp credential reset --name azurecaas \
  --password P@ssw0rd123

# Assign “reader” role for ACR to AKS service principal
az acr show --name azurecaas --resource-group azure-caas-demo \
  --query "id"
az role assignment create \
  --assignee ...(app id) --role Reader --scope ...(ACR id)

# Create AKS cluster
az aks create --resource-group azure-caas-demo \
  --name azure-caas-kube --node-count 1 --generate-ssh-keys \
  --location westeurope --client-secret P@ssw0rd123 \
   --service-principal ...(app id)
```

# Demo

Create AKS Cluster

Push demo image to ACR

Create service principal for AKS

Create AKS cluster

https://docs.microsoft.com/en-us/azure/aks/kubernetes-walkthrough

Note: Due to preview limitations, location has to be *eastus*, for details see https://github.com/Azure/AKS/blob/master/preview_regions.md

```
# Connect kubectl with AKS
az aks get-credentials --resource-group azure-caas-demo \
  --name azure-caas-kube

# Check connection to AKS
kubectl config get-contexts
kubectl config use-context …
kubectl config current-context
kubectl get nodes

# Start Kubernetes dashboard (CMD, not bash!)
az aks browse --resource-group azure-caas-demo \
  --name azure-caas-kube

# Scale Kubernetes server
az aks scale --resource-group azure-caas-demo \
  --name azure-caas-kube --node-count 2
```

# Demo

Create AKS Cluster

## Deploy app to AKS

# Workshop

## Deploy a Web API to Kubernetes

## Web API details

Web API implemented with Node.js (sources on GitHub, Dockerfile)
Automated build of image in Docker Hub (rstropek/node-mongo-sample)
Uses MongoDB (Docker Hub image *mongo*) to store data

## Your Job

Deploy MongoDB to Kubernetes
For debugging purposes, add a service to access MongoDB using e.g. *RoboMongo*
Deploy *rstropek/node-mongo-sample* (cluster of two replicas) and connect it to MongoDB
Add a service that makes the Web API available on the network

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: event-db
spec:
  selector:
    matchLabels:
      app: event-db
  template:
    metadata:
      labels:
        app: event-db
    spec:
      containers:
      - name: event-db
        image: mongo
        ports:
        - containerPort: 27017
---
apiVersion: v1
kind: Service
metadata:
  name: event-db-svc
spec:
  selector:
    app: event-db
  ports:
  - port: 27017
    targetPort: 27017
  type: NodePort
```

# Manifest

Mongo

Tip: In real world, use stateful set

https://kubernetes.io/blog/2017/01/running-mongodb-on-kubernetes-with-statefulsets/

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: api
spec:
  selector:
    matchLabels:
      app: api
  replicas: 2
  template:
    metadata:
      labels:
        app: api
    spec:
      containers:
      - name: api
        image: rstropek/node-mongo-sample
        ports:
        - containerPort: 80
        env:
          - name: MONGO_URL
            value: "mongodb://event-db-svc/member-management"
---
apiVersion: v1
kind: Service
metadata:
  name: api-svc
spec:
  selector:
    app: api
  ports:
  - port: 80
    targetPort: 80
  type: NodePort
```

# Manifest

API

# Q&A

Thank you for attending!


## Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com
Mail rainer@timecockpit.com
Twitter @rstropek


time cockpit
Saves the day.