# Autoscaling in Kubernetes

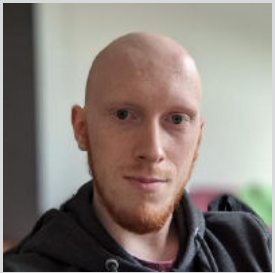
## From Zero to Hero


Lukas Paluch  Tobias Manske

12. März 2023

# Hello

# Wer sind wir



**mail** manske@atix.de
**in** linkedin.com/in/tmanske
**gh** github.com/rad4day

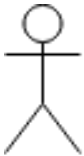
**mail** paluch@atix.de
**in** linkedin.com/in/lukasp
**gh** github.com/fluktuid

# Woher kommen wir

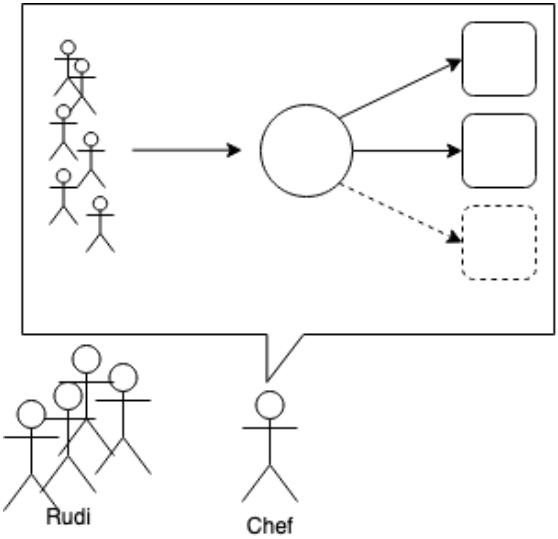
**ATIX AG**

- Transformation und Automatisierung von Rechenzentren
- Fokus auf
  - Lifecycle System Management
  - Container-Technologie

# Whats the Story?



# Whats the Story?



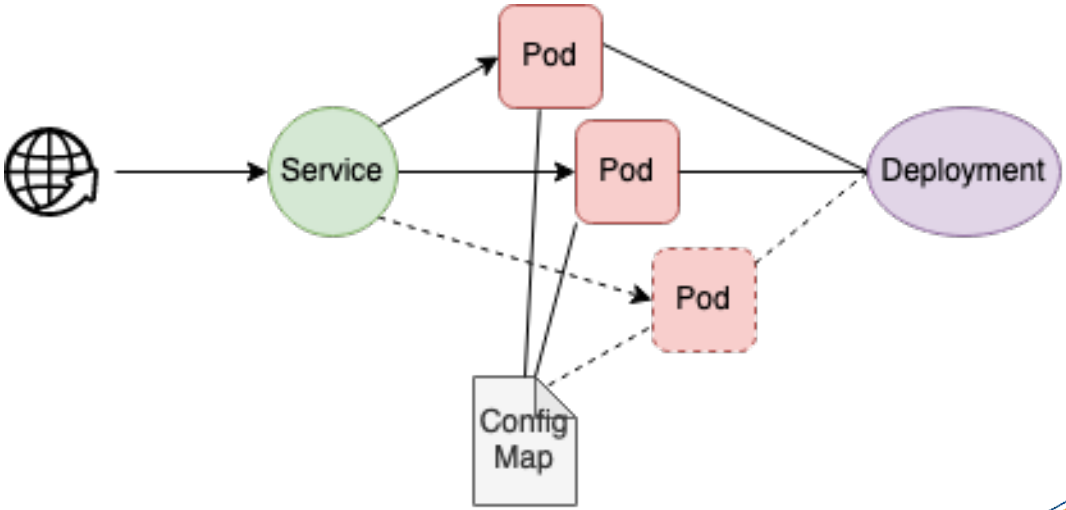
# Agenda

1. Whats Kubernetes
2. Setup
3. Skalierung
   - Skalierung des Clusters
   - Scale to Zero

# Whats Kubernetes

- Verteiltes System
- Betrieb (Orchestrierung) von Containern
- Gut skalierbar
- Möglichkeit zu HA-Setup

# Die Komponenten



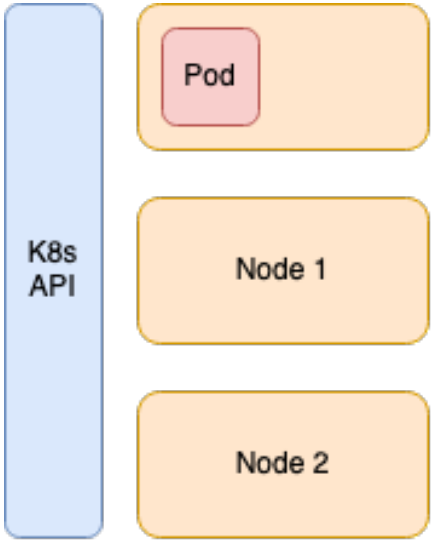
# Agenda

1 Whats Kubernetes

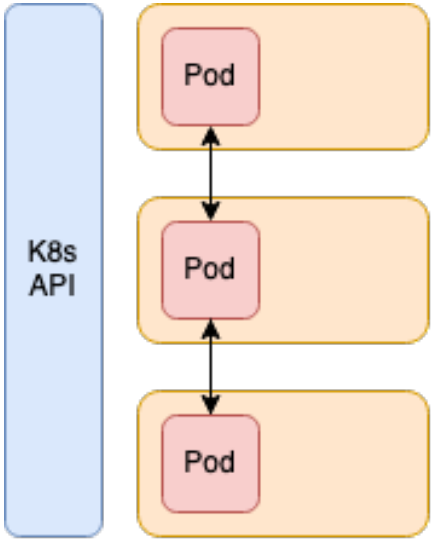
2 Setup

3 Skalierung
- Skalierung des Clusters
- Scale to Zero

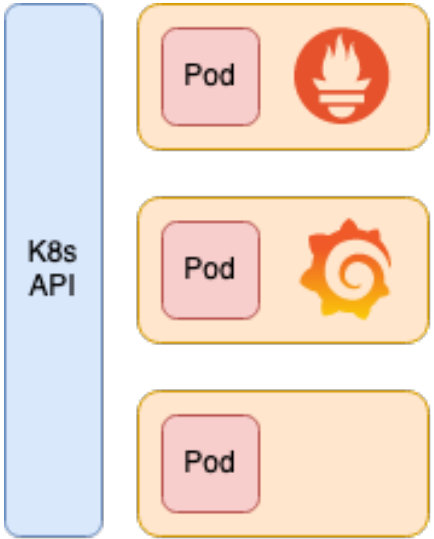
# Setup



# Setup Replizierung + Affinity



# Setup Monitoring



# Agenda

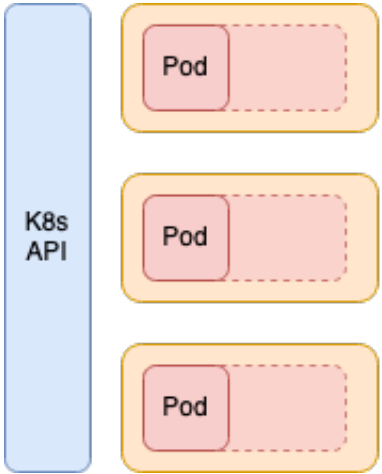
1 Whats Kubernetes

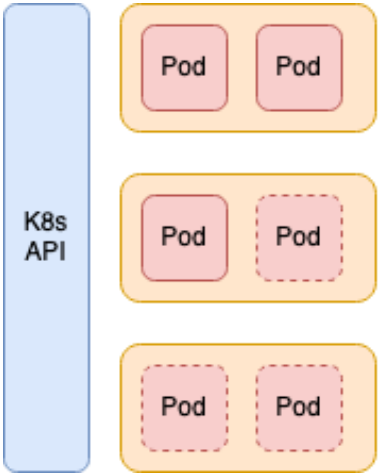
2 Setup

3 Skalierung
- Skalierung des Clusters
- Scale to Zero

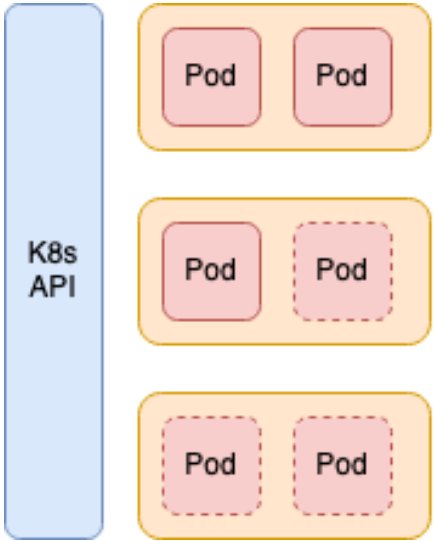
# Skalierung der Anwendung

# PodAutoscaler



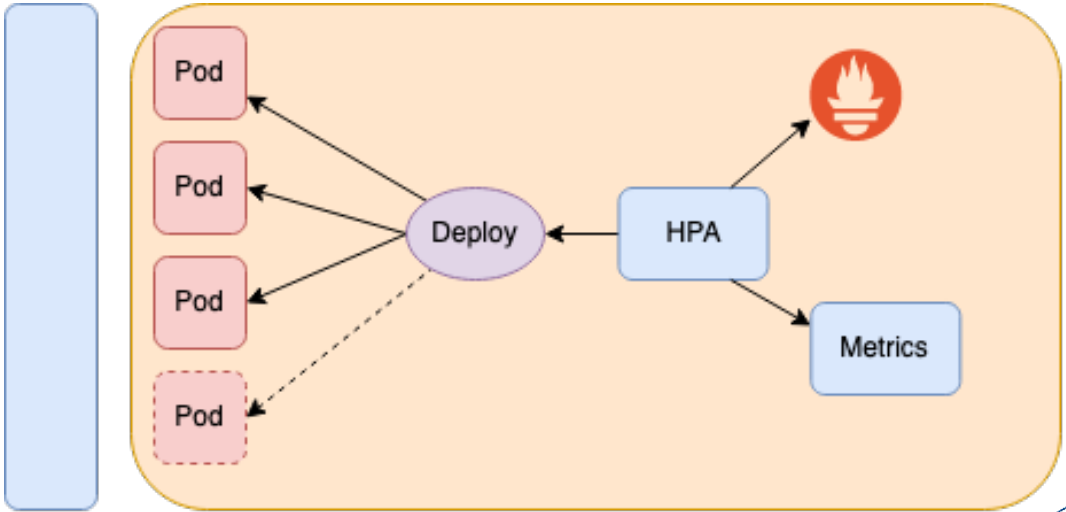
# HorizontalPodAutoscaler




```yaml
kind: HorizontalPodAutoscaler
...
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: server
  minReplicas: 1
  maxReplicas: 100
  metrics:
  - type: Pods
    pods:
      metric:
        name:
            server_requests_per_second

      target:
        type: AverageValue
        averageValue: 16
  ...
```

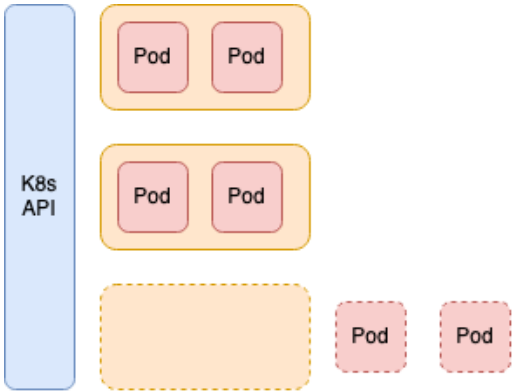
# So HPA



# Agenda

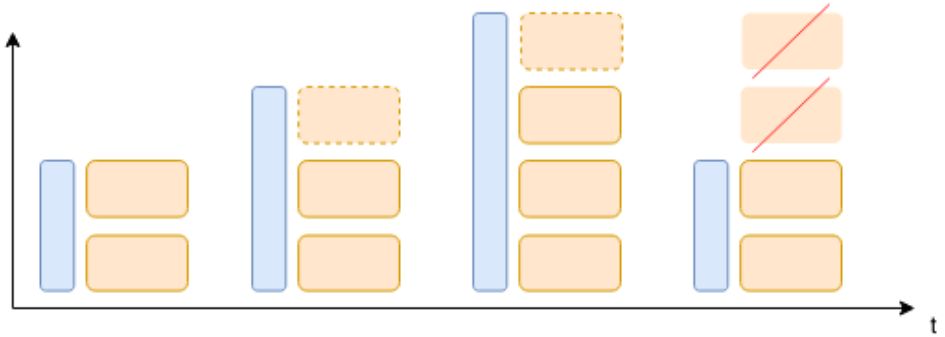
1. Whats Kubernetes
2. Setup
3. Skalierung
   - Skalierung des Clusters
   - Scale to Zero

# Skalierung des Clusters




```yaml
# cluster-autoscaler.yaml
autoscalingGroups:
  - name: "<eks-worker-group>"
    minSize: 1
    maxSize: 20
```

# Cluster Skalieren



# Agenda

1. Whats Kubernetes
2. Setup
3. Skalierung
   - Skalierung des Clusters
   - Scale to Zero

# Scale to Zero

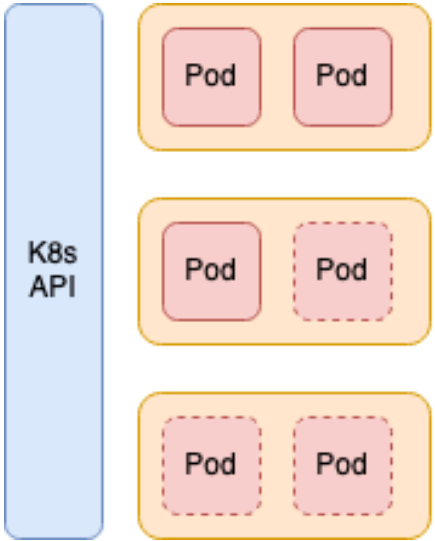
## X auf 0

- Einfach umzusetzen.
- Anwendung »verschwindet« wenn keine Last vorhanden.

## 0 auf X

- Benötigt podunabhängige Metrik.
- Was passiert mit Anfragen wenn kein Pod vorhanden?
- (Knative vs KEDA?)

# Scale to Zero



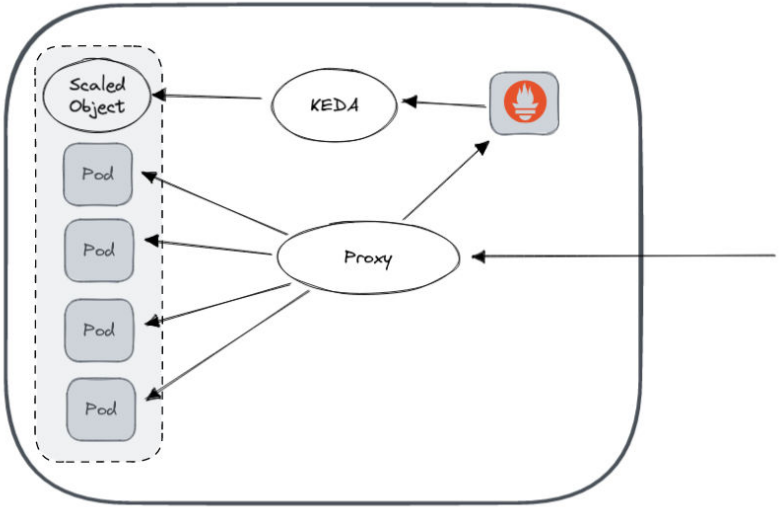

```yaml
apiVersion: keda.sh/v1alpha1
kind: ScaledObject
metadata:
  ...
spec:
  scaleTargetRef:
    name: scaled-deployment
  pollingInterval: 10
  cooldownPeriod:  15
  minReplicaCount: 0  # <--
  maxReplicaCount: 10
  triggers:
  - type: prometheus
    metadata:
      serverAddress: http://...
      metricName: access_frequency
      threshold: '25'
      query: sum(rate(
          node_http_requests_total
          [2m]))
```

# Das Problem

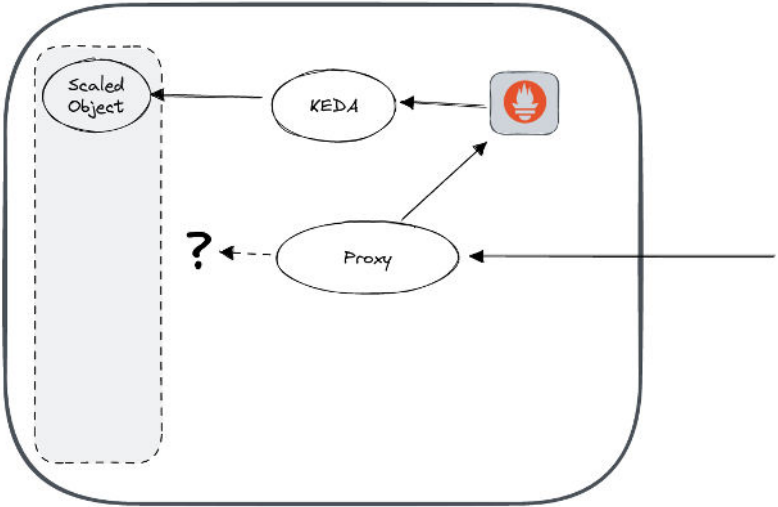
## Aus dem keda.sh Blog

Autoscaling HTTP is often not as straightforward as other event sources. You don't know how much traffic will be coming and, given its synchronous nature, supporting scale-to-zero HTTP applications requires an intelligent intermediate routing layer to "hold" the incoming request(s) until new instances of the backend application are created and running.

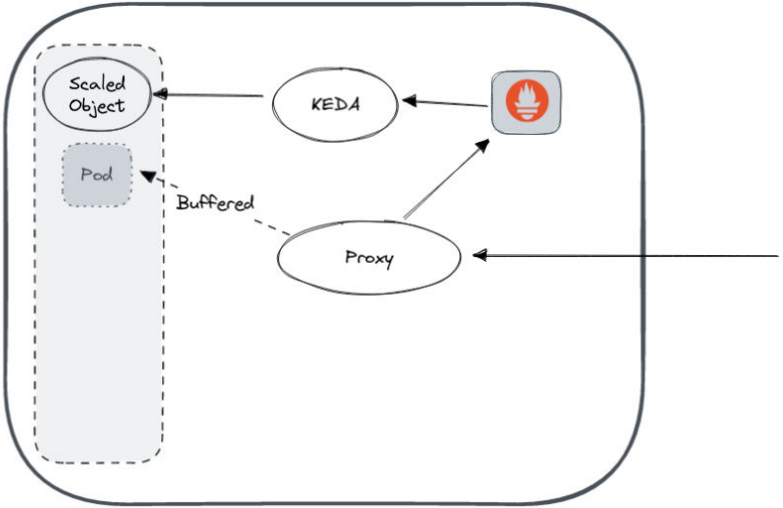
# Beispiel



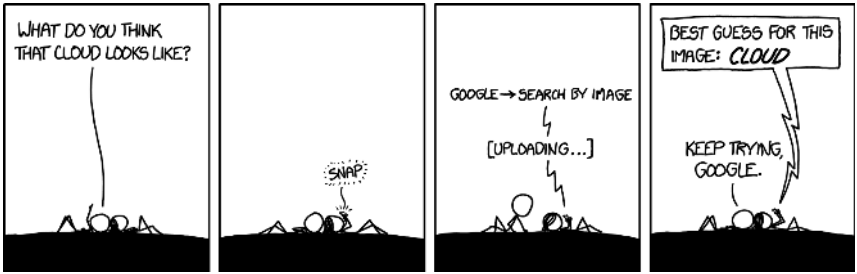
# Beispiel



# Beispiel



# And now there is Time for you




(xkcd.com/1444/)

# Sources

presentation & code
github.com/fluktuid/talk-k8s-skaling