Universität Stuttgart

Pascal
Hagemann

IPVS

Anwendungssoftware

Evaluating dynamic load balancing of ECM workload pattern employed in cloud environments managed by a Kubernetes/Docker eco-system

Master Thesis
Supervisor: Dipl.-Phys. Cataldo Mega
Examiner: Prof. Dr.-Ing. Bernhard Mitschang

07.10.2021
Universität Stuttgart, IPVS, AS

Overview

Introduction

Goals:

- Dynamic load balancing of a containerized ECM application in Kubernetes (K8s)
- Analysis of dynamic containerized deployment in the cloud
- Feasibility for porting applications to the new approach $\Rightarrow$ Prototype
- Identify challenges and pitfalls

Dynamic Load Balancing Approach

- Managing an elastic deployment in a dynamic cloud environment
- Elasticity based on current system state and workload type/amount
- Automatic system state surveillance and error handling based on heuristics (later aided by ML/AI)
- Maximize efficiency by avoiding under- and over-utilization of CPU, memory, network and I/O
- Elasticity on a component level dependent on the workload type

Properties

- Relies on a platform which supports elasticity and automation
- Can react to topology changes
- Reactive or proactive scheduling approach possible
- Tolerant towards workload pattern changes
- But: Additional ressource usage

$\Rightarrow$ High efficiency with small overhead

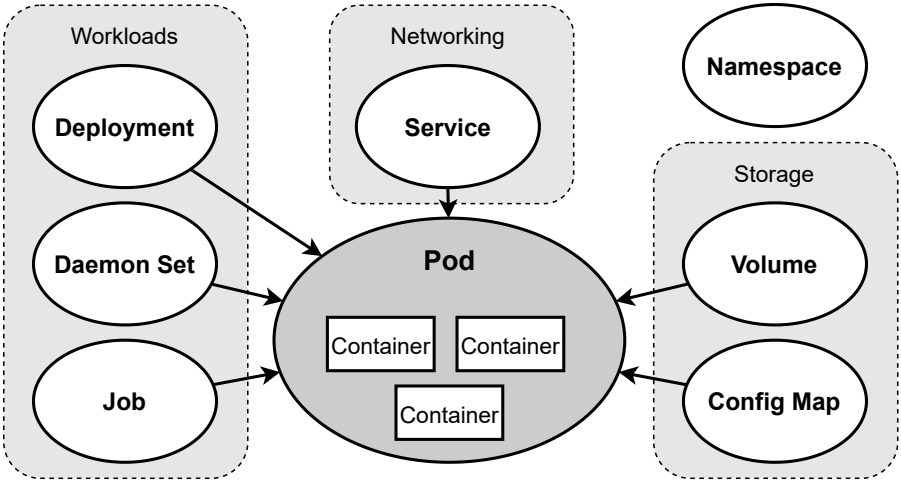
Prototype Component Challenges

1. Using Kubernetes for entity management and load balancing
2. Using Horizontal Pod Autoscaler (HPA) for replication rules/management
3. Adding a monitoring system for Metrics management
4. Fetching metrics from the ECM application components
5. Use a MAPE-Loop implementation to achieve elasticity

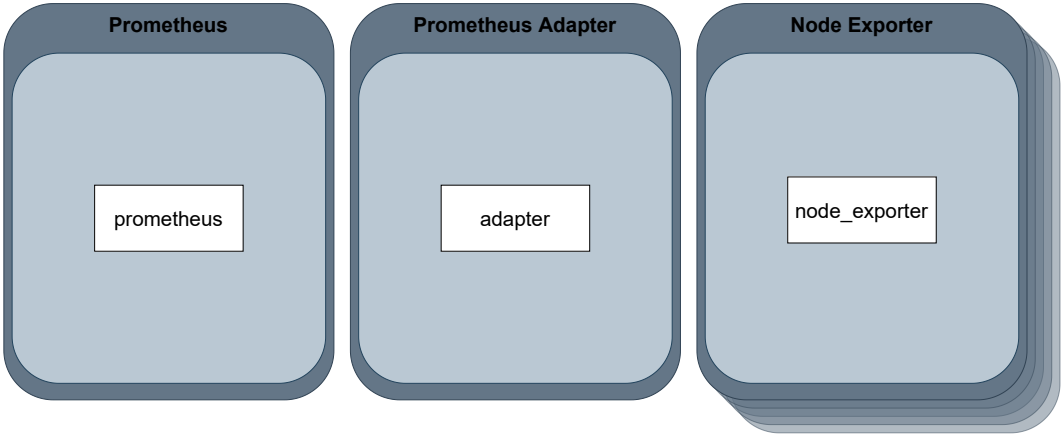
Kubernetes (K8s)

- Full container orchestration solution
- Incorporates many services (DNS, **scaling (HPA)**, storage, overlay network, ...)
- Smallest manageable unit: Pod containing 1 or more container
- Configuration through various objects

Kubernetes Objects



Monitoring Components



Prometheus

- Saves metrics in time-series database
- Optimized query language PromQL
- Integrates with Grafana for observation and analysis of data
- Fetches metrics from pods via exporters

Exporter

- Fetches metrics from various sources (process, file, ...)
- For a specific process/application (group)
- Makes the metrics available via a REST-Interface

Metrics Translation

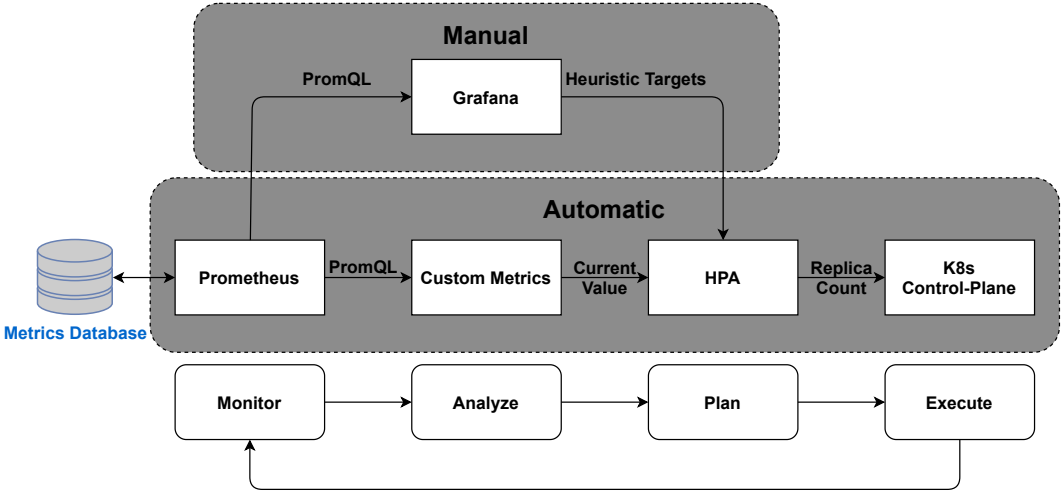
Prometheus Adapter

- Interface between Prometheus and Kubernetes
- Translates metrics from PromQL to "Custom Metrics"
- Can transform and combine several metrics

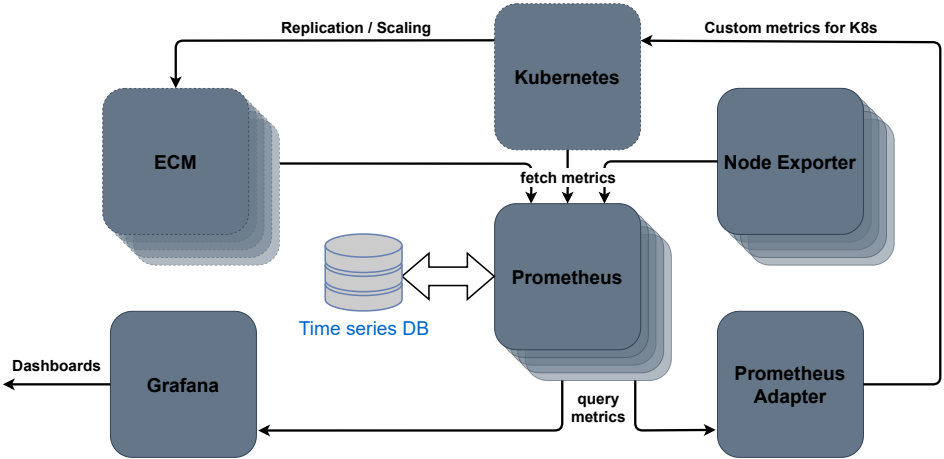
Horizontal Pod Autoscaler (HPA)

- Kubernetes subsystem
- Uses metrics provided by / to Kubernetes
- Scales the amount of pods in a Deployment to match a given target

MAPE-Loop Implementation

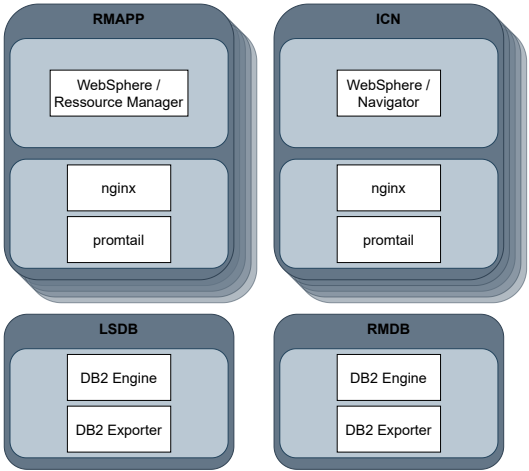


Metrics Dataflow



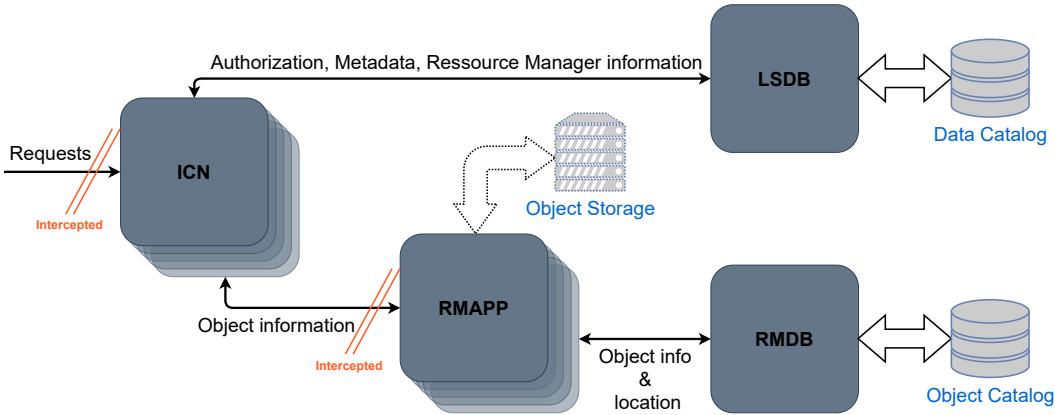
ECM Components

- Extended by NGINX container
- NGINX as agent to intercept requests
- Request metrics written into logfile (response time, traffic, path, ...)
- Parse logfile and publish metrics
- DB2 exporter to publish SQL query results as metrics

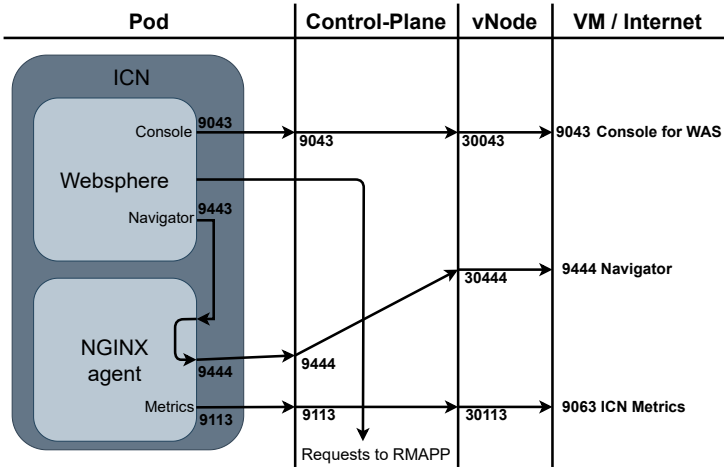


ECM Dataflow

- Requests intercepted by NGINX



Network map


Pod
Control-Plane
vNode
VM / Internet
ICN
Console
9043
Websphere
Navigator
9443
NGINX
agent
9444
Metrics
9113
9043
9444
9113
Requests to RMAPP
30043
30444
30113
9043 Console for WAS
9444 Navigator
9063 ICN Metrics

Test Setup

- Threshold: maximum response time of a request averaged over 1 minute (100ms)
- Workload: Login, Startpage, Search, Open Item, Open Thumbnail, Logout
- 30 concurrent users
- Workload repeated as fast as possible

$$targetInstances = ceil(currentInstances \cdot \frac{currentMetricValue}{threshold})$$

Test Results

1. Threshold reached
2. Scale up
3. Under threshold
4. Scale down after timeout

Challenges

- Generation of useful metrics
- Removing dependencies between components to enable (efficient) scaling
- Complex configuration options of Kubernetes and Prometheus
- Scaling of stateful applications like databases

Next Steps

- Optimize ECM application components for the cloud
- Include more application specific metrics
- Tests with more realistic workloads
- Proactive scheduling possibly with ML/AI support

Thank you for your attention