

# Technische Hochschule Ingolstadt

**Specialist area Computer Science**

**Bachelor's course Computer Science**

## Bachelor's thesis

**Subject:** Conception, Implementation, and Evaluation of a Highly Scalable and Highly Available Kubernetes-Based SaaS Platform on Kubernetes Control Plane (KCP)

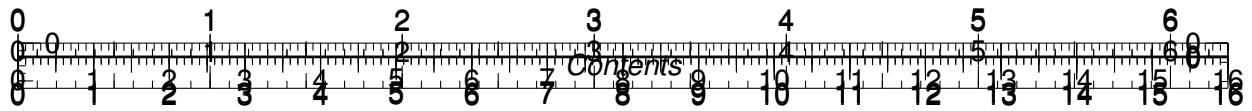
**Name and Surname:** David Linhardt

**Issued on:** TODO: Insert Issue Date

**Submitted on:** TODO: Insert Submit Date

**First examiner:** Prof. Dr. Bernd Hafenrichter

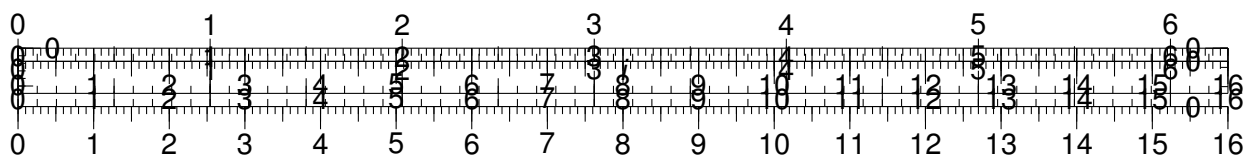
**Second examiner:** Prof. Dr. Ludwig Lausser

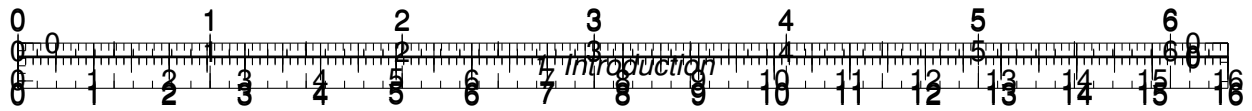


## Abstract

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Problem Statement and Motivation . . . . .	1
1.2	Objectives and Scope . . . . .	1
1.3	Structure of the Thesis . . . . .	1
<b>2</b>	<b>Fundamentals</b>	<b>1</b>
2.1	Kubernetes and Multi-Tenancy . . . . .	1
2.2	Kubernetes Control Plane (KCP) . . . . .	4
2.3	SaaS Architecture and Automation . . . . .	4
<b>3</b>	<b>State of the Art and Related Work</b>	<b>4</b>
3.1	Zero-Downtime Deployment Strategies . . . . .	4
3.2	Kubernetes Scaling Methods . . . . .	4
3.3	Multi-Tenancy Concepts in the Cloud . . . . .	4
<b>4</b>	<b>Conceptual Design</b>	<b>4</b>
4.1	System Requirements . . . . .	4
4.2	Architecture Design with KCP for SaaS . . . . .	4
4.3	Automated Deployment Strategies . . . . .	4
<b>5</b>	<b>Prototypical Implementation</b>	<b>4</b>
5.1	Infrastructure with KCP . . . . .	4
5.2	Tenant Provisioning . . . . .	4
5.3	Scaling Mechanisms . . . . .	4
5.4	Monitoring and Logging . . . . .	4
<b>6</b>	<b>Evaluation</b>	<b>4</b>
6.1	Performance Measurements . . . . .	4
6.2	Scaling Scenarios & Optimizations . . . . .	4
6.3	Discussion of Results . . . . .	4
6.4	Related Work . . . . .	4
<b>7</b>	<b>Conclusion and Outlook</b>	<b>4</b>
7.1	Summary . . . . .	4





7.2 Personal Conclusion . . . . .	4
7.3 Future Outlook . . . . .	4

References . . . . .	4
----------------------	---

List of Figures . . . . .	4
---------------------------	---

## Glossary

## 1 Introduction

### 1.1 Problem Statement and Motivation

### 1.2 Objectives and Scope

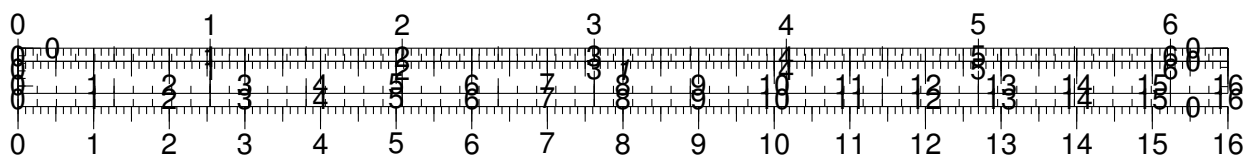
### 1.3 Structure of the Thesis

## 2 Fundamentals

### 2.1 Kubernetes and Multi-Tenancy

**Kubernetes as the Foundation for Cloud-Native Applications** As the de facto standard for deploying and managing *cloud-native applications*, Kubernetes plays a pivotal role in modern cloud architecture . Kubernetes works as an application orchestrator for *containerized, cloud-native microservice* apps, meaning it can deploy applications and dynamically respond to changes . It offers a platform for declarative configuration and automation for containerized workloads, enabling organizations to run distributed applications and services at scale .

**The Importance of Multi-Tenancy in Modern SaaS Platforms** Multi-tenancy plays a fundamental role in modern cloud computing. By allowing multiple tenants to share the same infrastructure through virtualization, it significantly increases resource utilization, reduces operational costs, and enables essential features such as VM mobility and dynamic resource allocation . These benefits are critical for cloud providers, as they make the cloud business model economically viable and scalable. In the context of modern SaaS platforms, multi-tenancy goes even further by enabling unified management, frictionless onboarding, and simplified operational processes that allow providers to add new tenants without introducing incremental complexity or cost .





However, while multi-tenancy is indispensable for achieving efficiency, scalability, and cost-effectiveness, it simultaneously introduces complex security challenges, especially in shared environments where resource isolation is limited. In particular, the potential for cross-tenant access and side-channel attacks makes security in multi-tenant environments a primary concern . As such, understanding and addressing multi-tenancy from both operational and security perspectives is essential when designing and securing modern cloud-native platforms .

## The Challenges of Multi-Tenancy and the Need for Solutions

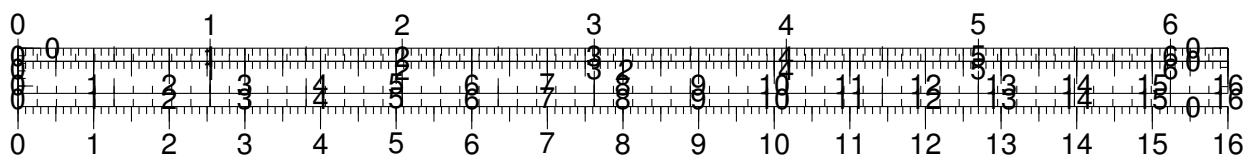
### Kubernetes Control Plane (KCP) as a Promising Approach

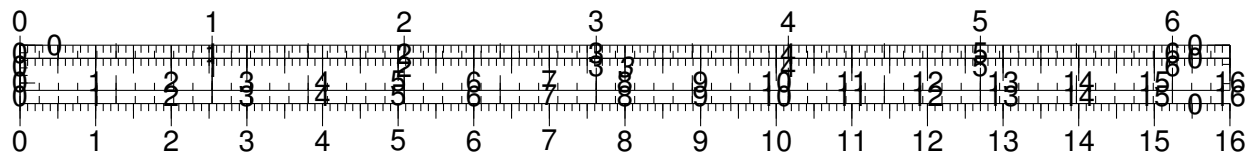
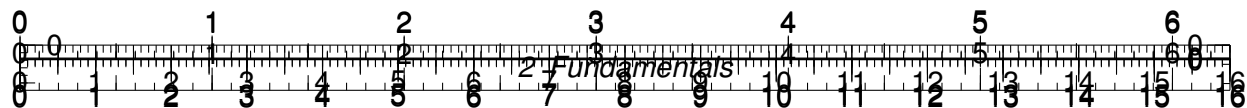
**Background: The Evolution of Kubernetes** Kubernetes was originally developed at Google and released as open source in 2014 .

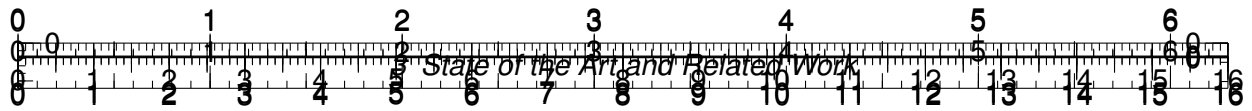
**Background: Containerization as an Enabler of Kubernetes** *Containerization* is a way to bundle an application's code with all its dependencies to run on any infrastructure thus enhancing portability . This comes with additional advantages that can be leveraged by Kubernetes, including vertical and horizontal autoscaling facilitated by quick container boot times, along with self-healing mechanisms and support for distributed, resilient infrastructures .

### Background: Kubernetes Resource Isolation Mechanisms

### Relevance to SaaS and this Thesis







## **2.2 Kubernetes Control Plane (KCP)**

## **2.3 SaaS Architecture and Automation**

# **3 State of the Art and Related Work**

## **3.1 Zero-Downtime Deployment Strategies**

## **3.2 Kubernetes Scaling Methods**

## **3.3 Multi-Tenancy Concepts in the Cloud**

# **4 Conceptual Design**

## **4.1 System Requirements**

## **4.2 Architecture Design with KCP for SaaS**

## **4.3 Automated Deployment Strategies**

# **5 Prototypical Implementation**

## **5.1 Infrastructure with KCP**

## **5.2 Tenant Provisioning (Automation, Multi-Tenancy)**

## **5.3 Scaling Mechanisms (Horizontal Pod Autoscaler)**

## **5.4 Monitoring and Logging (Prometheus, Grafana)**

# **6 Evaluation**

## **6.1 Performance Measurements (Downtime, Latency, Scaling)**

## **6.2 Scaling Scenarios & Optimizations**

## **6.3 Discussion of Results**

## **6.4 Related Work**

# **7 Conclusion and Outlook**

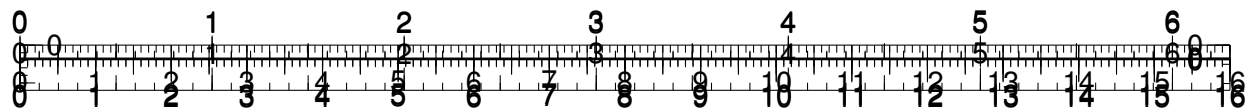
## **7.1 Summary**

## **7.2 Personal Conclusion**

## **7.3 Future Outlook**

# **References**

## **List of Figures**



## Appendix

