Contents

1. Introduction

1.1. Purpose of the Thesis

1.2. Scope of the Thesis

1.3. Structure of the Thesis

In chapter Background and Related Work

2. Background and Related Work

2.1. Containerization

Contenerization is packaging an app along with all necessary runtime stuff like libraries, executables or asets into an object called "container". The main benefits of container are[3]:

– Protable and Flexible – container can be run on bare metal or virtual machine in cloud regardless of operating system. Only a container runtime software like Docker Engine or containerd is required, which allows to interact with host system.

– Lightweight – container is sharing operating system kernel with hostmachine, there is no need to install separate operating system inside

– Isolated – does not depends on host's environment or infrastructure

– Standarized – Open Container Initiative standarize runtime, image and distribution specifications

A container image is set of files and configuration needed to run a container. It is immutable, only new image can be crated with new changes. Consists of layers. The layer contain one modification made a image. All layers are cachable and can be reused when building an image. The mechanism is really usefull when compiling large application components inside one container[2].

2.2. Contianer Orchestration

Container orchestration is coordinated deploying, managing, networking, scaling and monitoring containers process. It automates and manages whole container's lifecycle, there is no need to worrying about of deployed app, orchestration software like Kubernetes will take care of its availability [3].

The Kubernetes Authors says: "The name Kubernetes originates from Greek, meaning helmsman or pilot. K8s as an abbreviation results from counting the eight letters between the

"K" and the "s"". K8s is open-source orchestration platform capable of managing containers. Key functionalities are [1]:

- Automated rollouts and rollbacks – updates or downgrades version of deplotyed containers at controller rate, replacing containers incrementally
- Automatic bin packing – allows to specify exact resources needed by container (CPU, Memory) to fit on appropriate node
- Batch execution – possible to create sets of tasks which can be run without manual intervention
- Designed for extensibility – permits to add feautres using custom resource definitions without changing source code
- Horizontal scaling – scales (replicate) app based of its need for resources
- IPv4/IPv6 dual-stack – allocates IPv4 or IPv6 to pods and services
- Secret and configuration management – allows store, manage and update secrets. Containers do not have to be rebuilt to access updated credentials
- Self-healing – restarts crashed containers or by failure specified by user
- Service discovery and load balancing – advertises a container using DNS name or ip. Loadbalances traffic across all pods in deployment
- Storage orchestration – mounts desired storage like local or shipped by cloud provider

2.3. Kubernetes Architecture

A cluster is set of machines controlled by K8s. Tekst:

1. Tekst
2. Tekst

Listing 2.1: Python function to greet

```
def greet(name):
    print(f"Hello, {name}!")
```

2.3.1. Kuberenetes Basics

2.3.2. Control Plane

2.3.3. Nodes

2.3.4. Objects

2.3.5. Interfaces

TEST

TEKST asdasdasda

2.4. Cluster Networking

Tekst

1. Tekst

2. Tekst

Tekst

2.5. Container Network Interface (CNI)

2.6. Overview of Selected CNI Plugins

2.7. Related Work

DoTekst Na stronie `http://kile.sourceforge.net/screenshots.php` Tekst *Kile*, Tekst

Tekst

– Tekst

Bibliography

[1] The Kubernetes Authors. Overview. https://kubernetes.io/docs/concepts/overview/. Accessed, 07-Dec-2024.

[2] Docker Inc. What is an image? https://docs.docker.com/get-started/docker-concepts/building-images/. Accessed, 07-Dec-2024.

[3] Redhat Inc. What is containerization? https://www.redhat.com/en/topics/cloud-native-apps/what-is-containerization. Accessed, 07-Dec-2024.