

Modul CNT Kubernetes

September 2021

Marcel Bernet

Dieses Werk ist lizenziert unter einer
[Creative Commons Namensnennung - Nicht-kommerziell -
Weitergabe unter gleichen Bedingungen 3.0 Schweiz Lizenz](https://creativecommons.org/licenses/by-nc-sa/3.0/de/) /



Lernziele

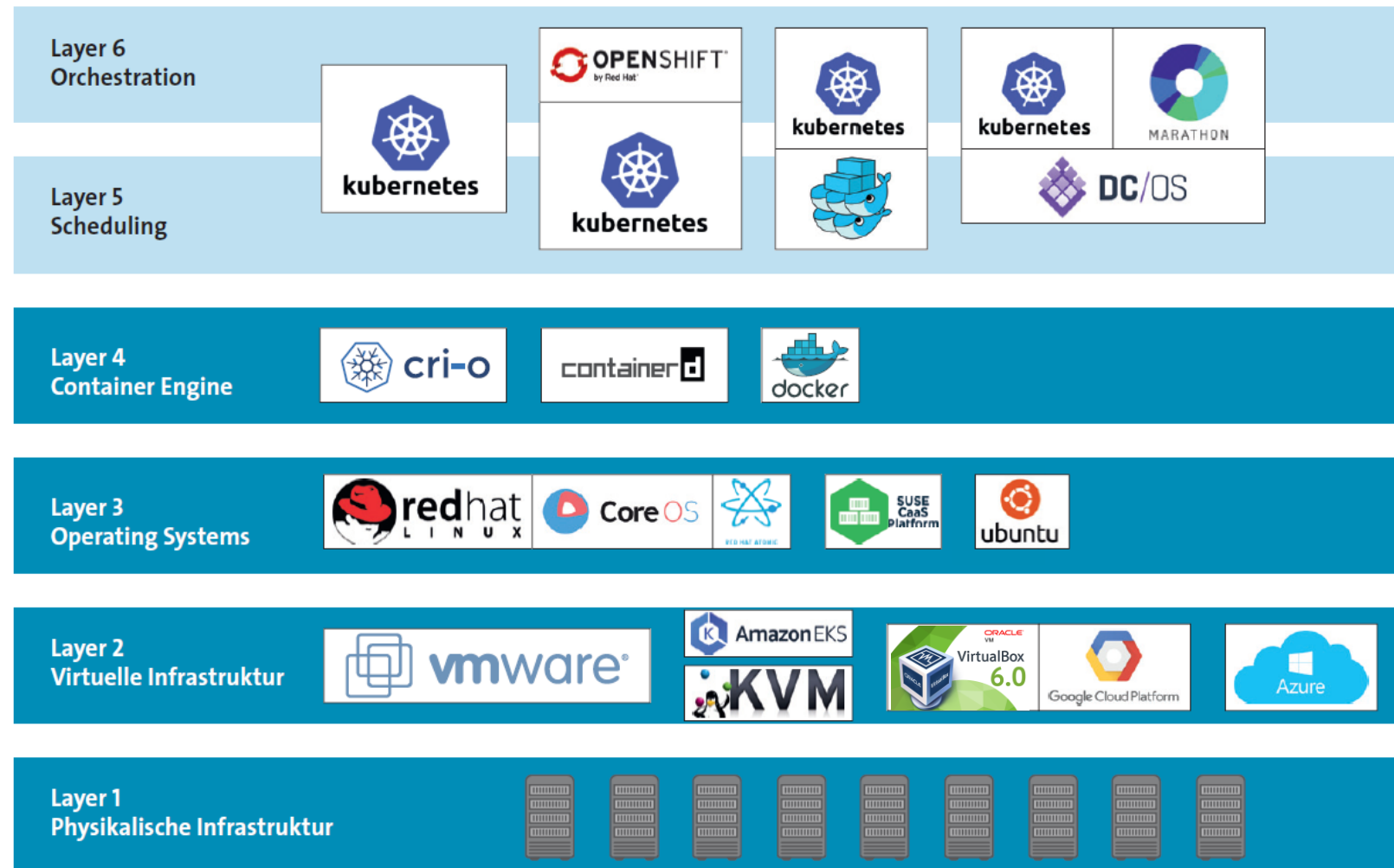
- ★ Sie haben einen ersten Einblick in Kubernetes
- ★ Sie haben einen ersten Überblick über das Container Ökosystem.

Zeitlicher Ablauf

- ★ Die (vereinfachten) »Layer« der Container-Welt, Layer 5 und 6
- ★ Kubernetes (K8s) im Überblick und Merkmale
- ★ K8s-Komponenten
- ★ K8s-Dienste auf den Master und Worker Nodes
- ★ Networking in Kubernetes
- ★ K8s Master- und Worker-Setup
- ★ Übung
- ★ Ökosystem: Cloud Native Landscape und CNCF
- ★ Übung
- ★ Reflexion
- ★ Lernzielkontrolle

Ist Kubernetes ein Produkt? Was macht Kubernetes?

Die (vereinfachten) »Layer« der Container-Welt

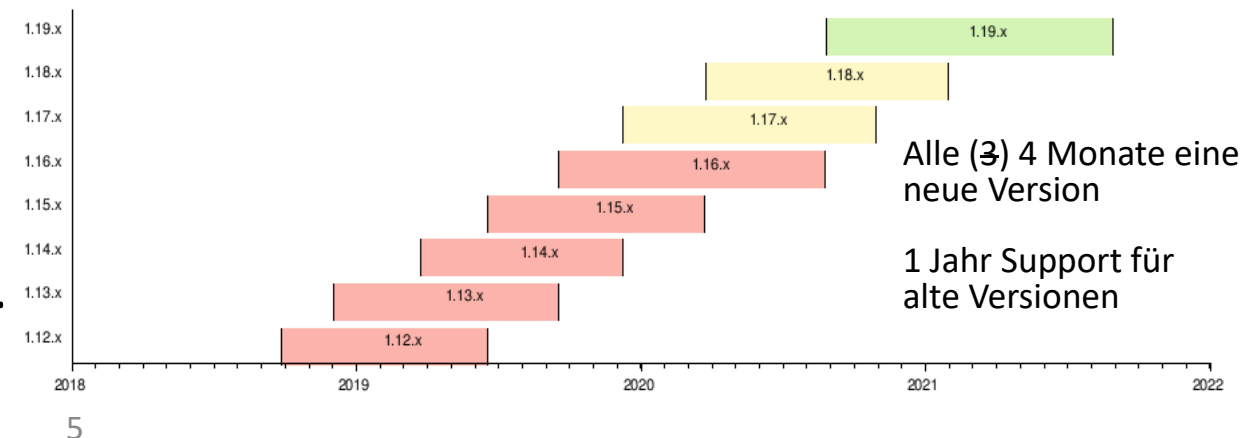


- ★ Die technische Definition von **Orchestrierung** ist die Ausführung eines definierten Workflows: zuerst A, dann B, dann C.
- ★ Kubernetes besteht aus einer Reihe von unabhängigen, komponierbaren Steuerungsprozessen, die den aktuellen Zustand kontinuierlich in Richtung des bereitgestellten Soll-Zustandes vorantreiben.

Kubernetes (K8s) im Überblick



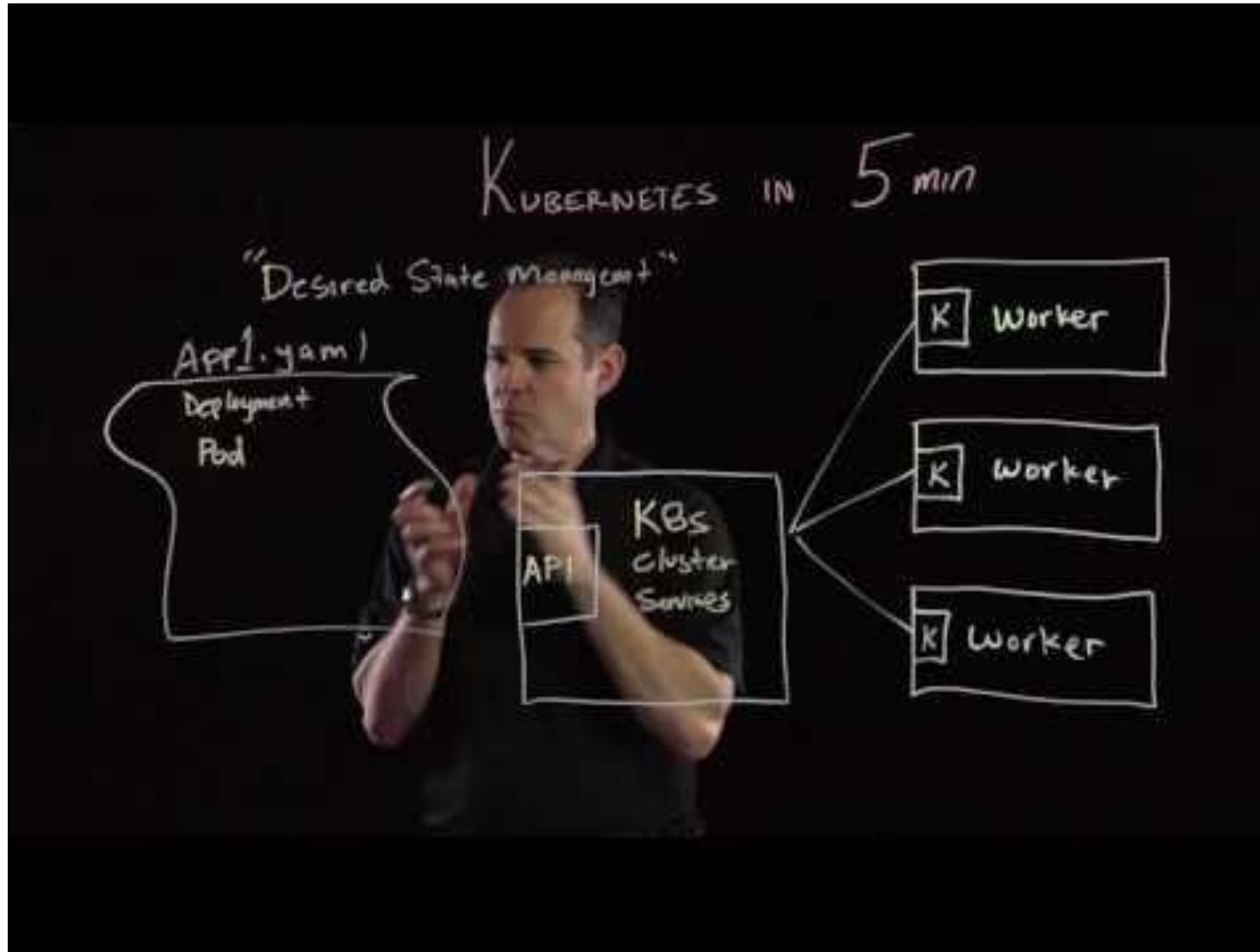
- ★ Das im Juli 2014 gestartete **Kubernetes** (griechisch: Steuermann) stellt die derzeit **populärste** Container-Cluster-/Orchestrierungs-Lösung dar.
- ★ Kubernetes ist mittlerweile bei der **Cloud Native Computing Foundation (CNCF)** gehostet.
- ★ Kubernetes' Hauptaufgabe ist die **Verwaltung und Orchestrierung der Container** innerhalb eines Clusters, der üblicherweise aus mindestens einem Kubernetes Master und multiplen Worker Nodes besteht.
- ★ Kubernetes wurde von Google ursprünglich als Orchestrierungs-Framework unter der Code-Bezeichnung Borg initiiert. **Ziel** war es, Docker- oder Rocket-(CoreOS)-**Container** im **grossem Umfang** zu **deployen**, zu **administrieren** und transparent zu **orchestrieren**.
- ★ Da Google Kubernetes bzw. seine Google-interne Implementierung davon selbst einsetzt, ist relativ sicher, dass K8s **kein kurzlebiges Projekt** sein wird.
- ★ K8s **arbeitete** primär mit Docker-Containern und APIs.



Kubernetes (K8s) Merkmale

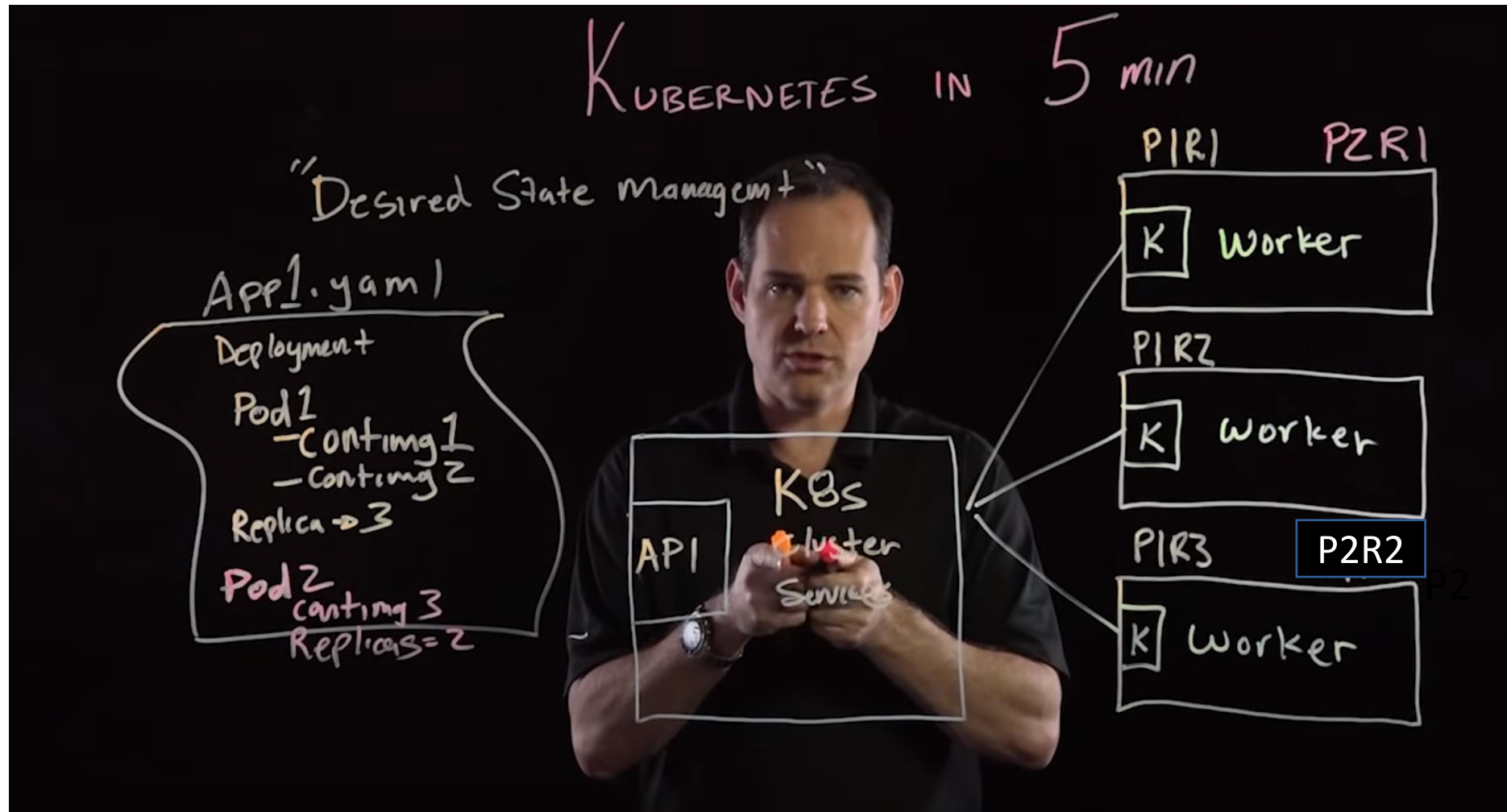
- ★ Pods als Behälter für Container
- ★ Immutable (Unveränderlich) statt Mutable.
- ★ Deklarative statt Imperative (Ausführen von Anweisungen) Konfiguration.
- ★ Selbstheilende Systeme - Neustart bei Absturz.
- ★ Entkoppelte APIs – LoadBalancer / Ingress ([Reverse Proxy](#), URL-based routing).
- ★ Skalieren der Services durch Änderung der Deklaration.
- ★ Anwendungsorientiertes statt Technik (z.B. Route 53 bei AWS) Denken.
- ★ Abstraktion der Infrastruktur statt in Rechnern Denken.

Kubernetes: in 5 Minuten



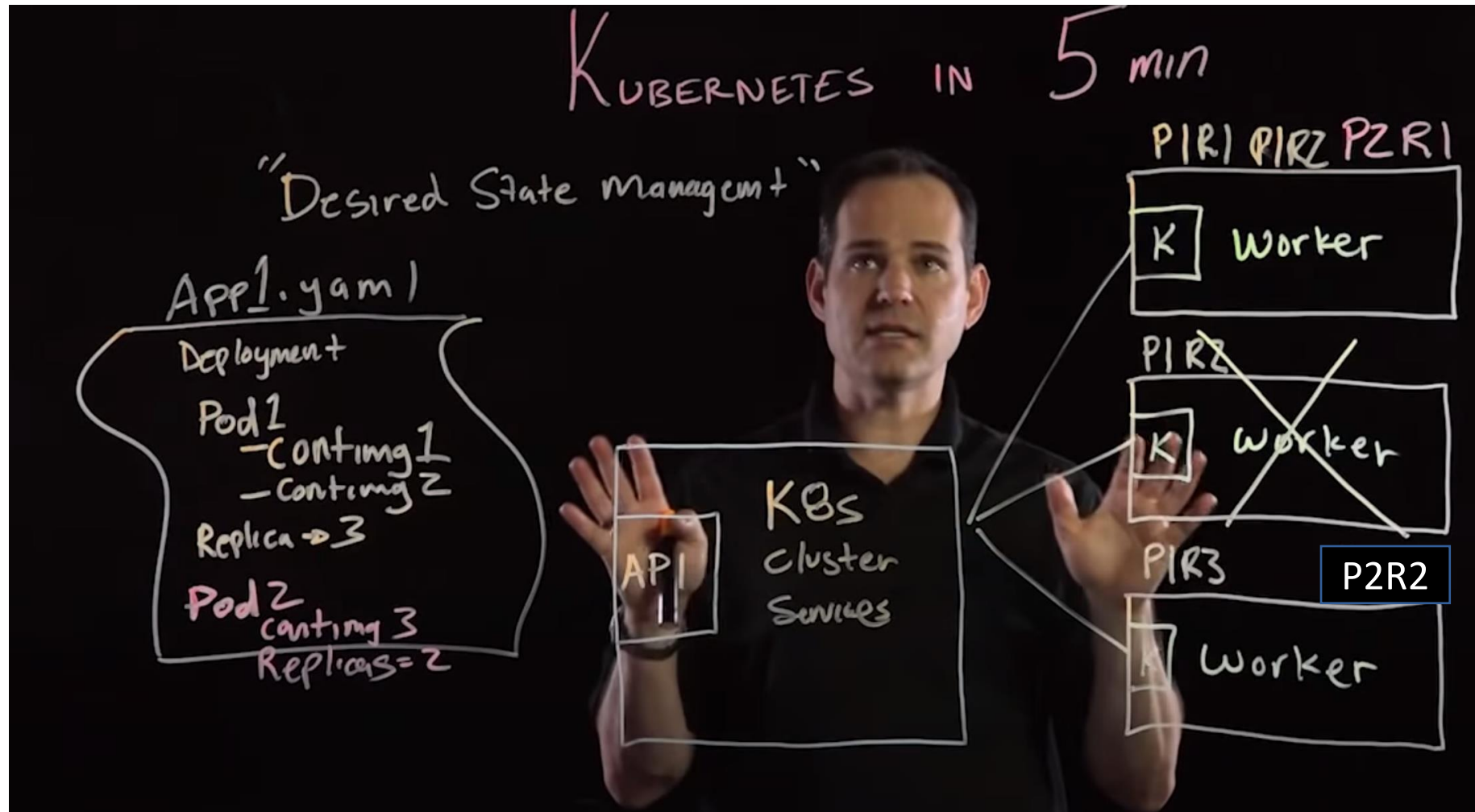
- ★ **Cluster** (vom Englischen für "Rechner-Schwarm", "-Gruppe" oder "-Haufen"), bezeichnet eine Anzahl von vernetzten Computern oder Virtuellen Maschinen.
- ★ Zusammenfassend für zwei unterschiedliche Aufgaben:
 - die Erhöhung der Rechenkapazität
 - die Erhöhung der Verfügbarkeit (HA-Cluster, engl. high available - hochverfügbar).
- ★ Die in einem Kubernetes Cluster befindlichen Einheiten werden als (Master oder Worker) Nodes bezeichnet.

Kubernetes: in 5 Minuten (Deklarativ, API)



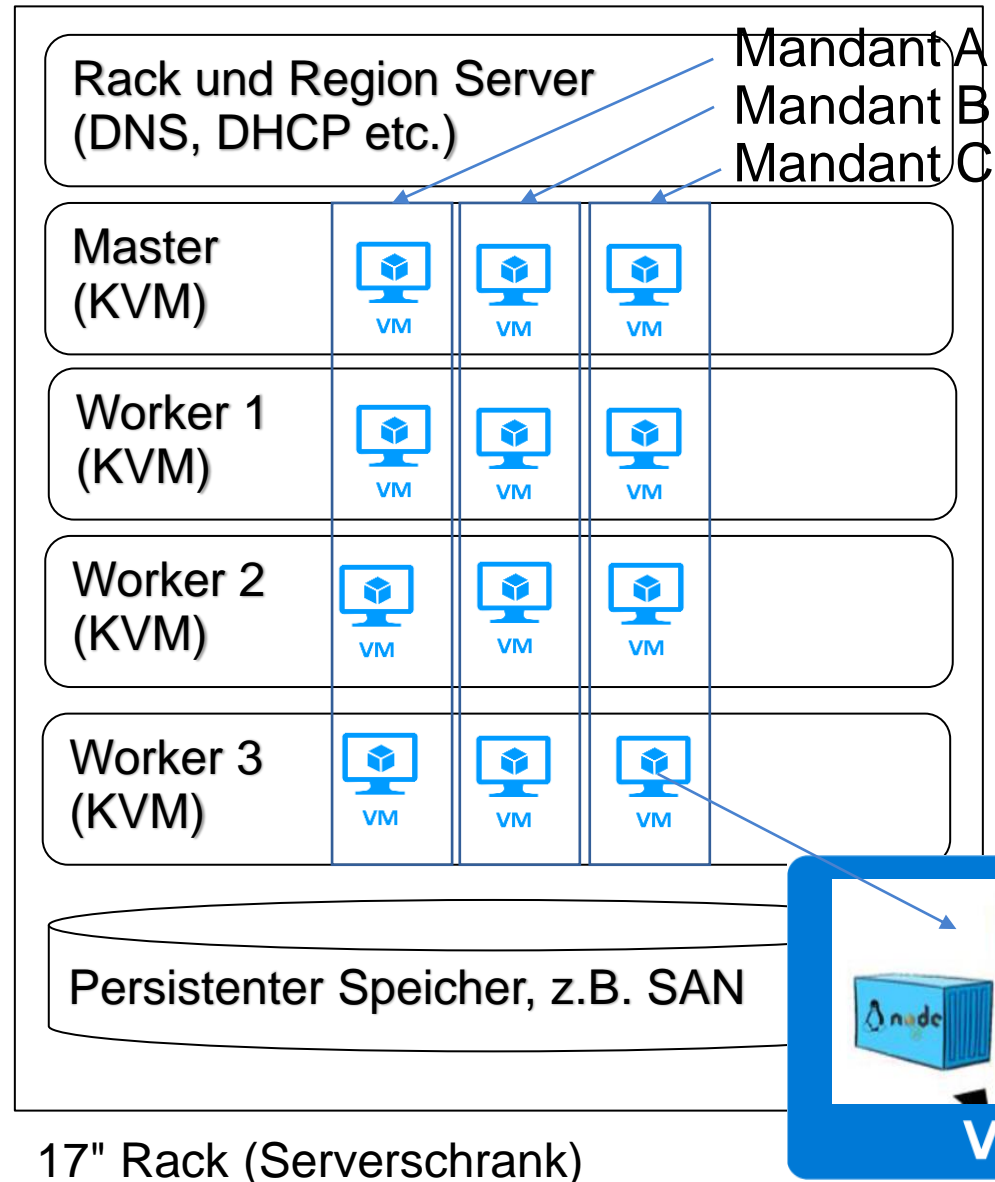
Quelle: <https://www.youtube.com/watch?v=PH-2FfFD2PU>

Kubernetes: in 5 Minuten (HA - Erhöhung der Verfügbarkeit)



Quelle: <https://www.youtube.com/watch?v=PH-2FfFD2PU>

Beispiel: Kubernetes Cluster für 3 Mandanten



- ★ Es werden mindestens 3 Nodes pro Cluster installiert
- ★ Die Nodes werden über die physikalischen Server verteilt.
- ★ Es wird eine Kubernetes Distribution verwendet, welche keine Unterscheidung Master/Worker Nodes macht.
- ★ **Hinweis:** durch Aufteilung von Kubernetes auf mehrere VMs, kann im laufenden Betrieb, ein Update der K8s Nodes durchgeführt werden.

Kann ich mehrere Cluster aufsetzen und z.B. gemeinsam Monitoren?

- ★ Kubernetes ist, in der Vergangenheit, nicht bekannt für seine Mandantenfähigkeit. "Kleinere" Cluster können einen Teil dieses Problems mindern.
- ★ Zalando hat 80 Kubernetes Cluster, in der AWS Cloud, aufgesetzt und nutzt einige Infrastrukturen gemeinsam, z. B. Monitoring (Prometheus) und der Ingress-Proxy ([Skipper](#)).
- ★ Der Ausfallradius ist begrenzt - alles, was in einem Cluster schief geht (Ausfall, Sicherheitsvorfall usw.), wirkt sich nicht unbedingt auf die gesamte Organisation aus.
- ★ Die Kostenaufteilung ist einfacher (jeder Cluster gehört zu einer Kostenstelle).
- ★ Der Cluster (und sein AWS-Konto) dienen als natürliche Vertrauensgrenze für die Zugriffssteuerung.

K8s-Komponenten (teilweise *veraltet*)

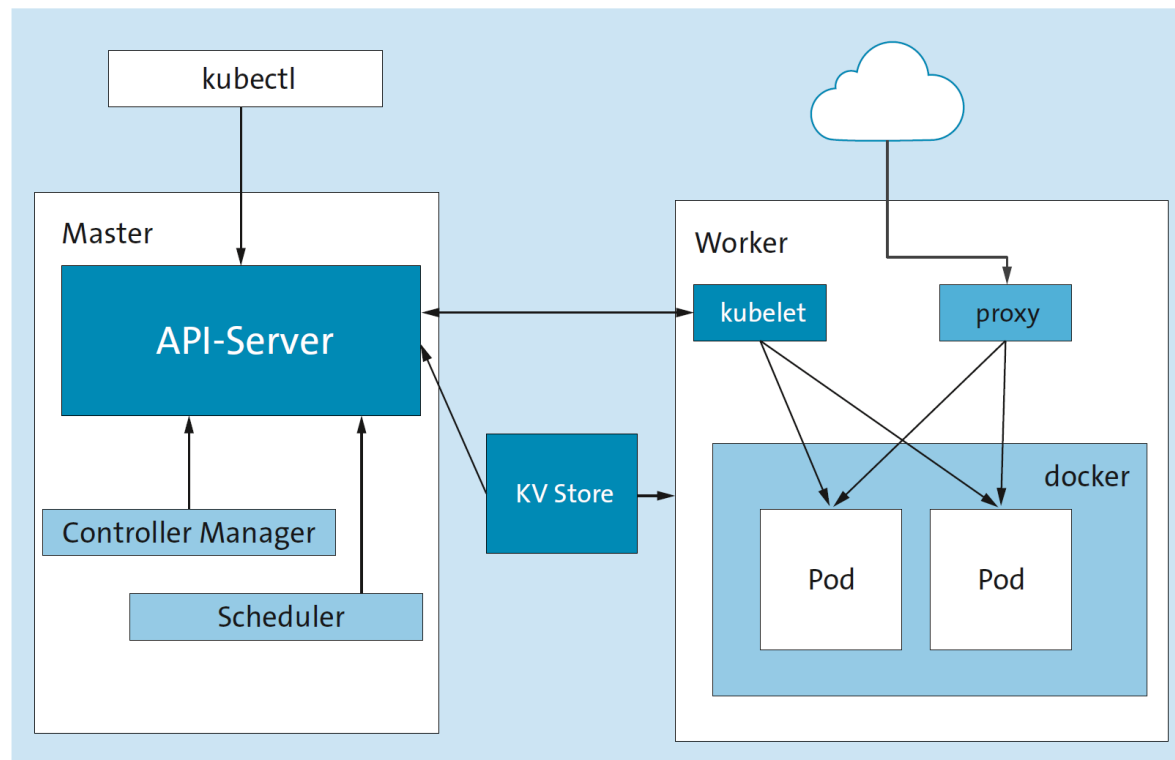


Abbildung 13.1 K8s Master und Worker aus der Vogelperspektive

★ Master

- der zentrale Koordinator in einem K8s Cluster, über den der Cluster gesteuert wird.

★ Worker

- Sie sind die Arbeits-Nodes des K8s. Auf ihnen laufen die Container bzw. im K8s-Kontext: Pods

★ Konfigurationsdateien und Keys:

- `/etc/kubernetes/`

Docker unter Windows (nur Kubernetes Worker!)

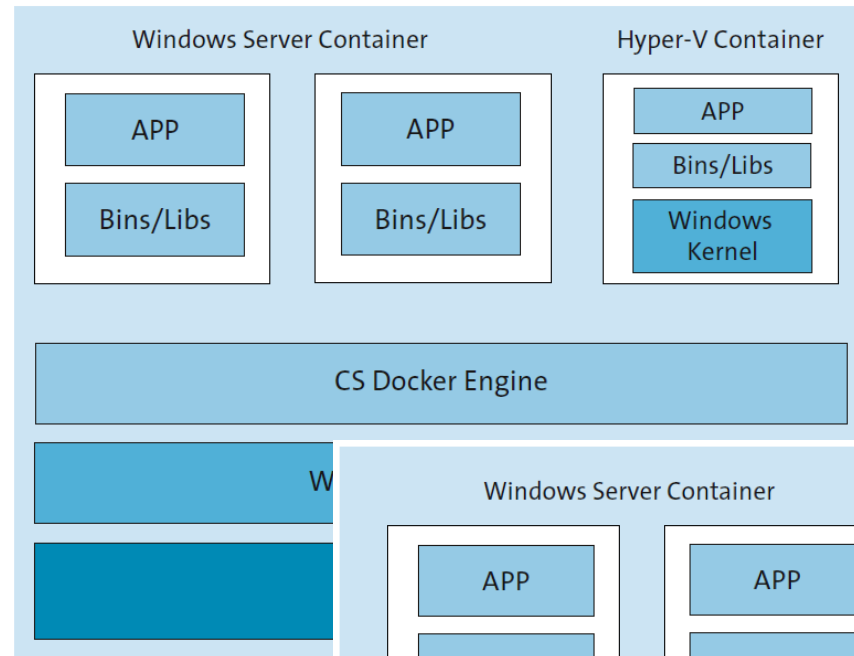


Abbildung 7.1 Docker unter

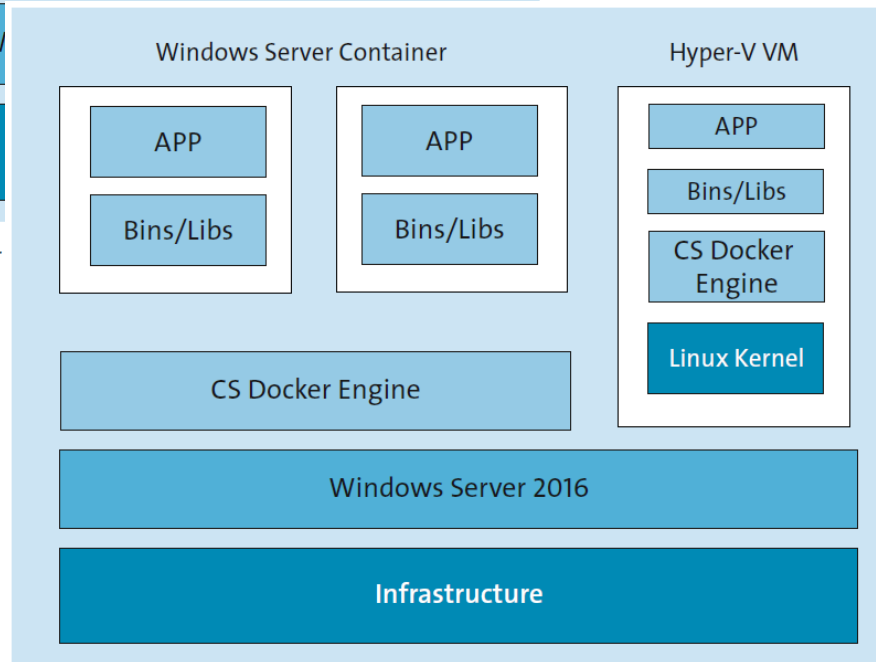


Abbildung 7.2 Normale und Hyper-V-Container unter Windows

- ★ Um Windows-Container ausführen zu können, muss Ihr Kubernetes-Cluster mehrere Betriebssysteme umfassen, wobei die **Master Nodes Linux** ausführen und die Worker entweder Windows oder Linux ausführen.
- ★ **Windows Server 2019 ist** das einzige **unterstützte** Windows-Betriebssystem und ermöglicht Kubernetes Nodes unter Windows (einschliesslich kubelet, container runtime und kube-proxy).
- ★ Quelle: [Intro to Windows support in Kubernetes](#)

K8s-Dienste auf den Master Nodes

```

~01:~$ kubectl get pods --all-namespaces
NAME                                READY
coredns-5644d7b6d9-dsxh7            0/1
coredns-5644d7b6d9-ndwjq            0/1
etcd-master-01                       1/1
kube-apiserver-master-01             1/1
kube-controller-manager-master-01    1/1
kube-proxy-j7fb2                     1/1
kube-scheduler-master-01             1/1

```

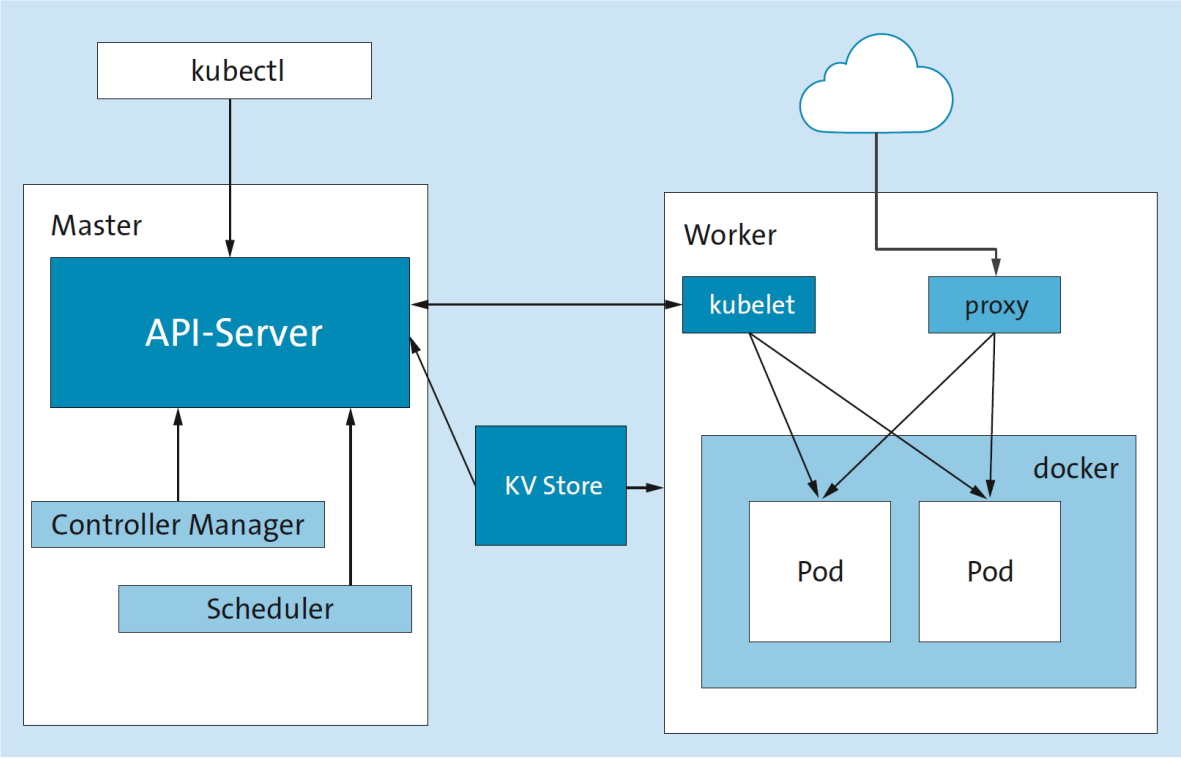


Abbildung 13.1 K8s Master und Worker aus der Vogelperspektive

- ★ **API-Server**
 - zentrale administrative Service-Komponente.
 - Managen aller Ressourcen des K8s Clusters.
- ★ **Controller Manager (neu Replication Controller)**
 - Er ist u.a. verantwortlich für die gewünschte Anzahl von ReplicaSets-Objekten im gesamten Cluster.
- ★ **Scheduler**
 - Der Scheduler-Prozess weist den Worker Nodes ihre Workloads, also die Pods/Container-Instanzen zu. Der Scheduler Prozess (**kube-scheduler**) kennt die Auslastung aller Worker Nodes und kann die Pods so immer bestmöglich platzieren.
- ★ **Etcd**: Key/Value (KV)-Store (No-SQL Datenbank)
- ★ **CoreDNS**: DNS Server
- ★ **Hinweis: ab Version 1.5/1.6 basieren die K8s Dienste meistens auf Pod's.**

K8s-Dienste auf den Workern

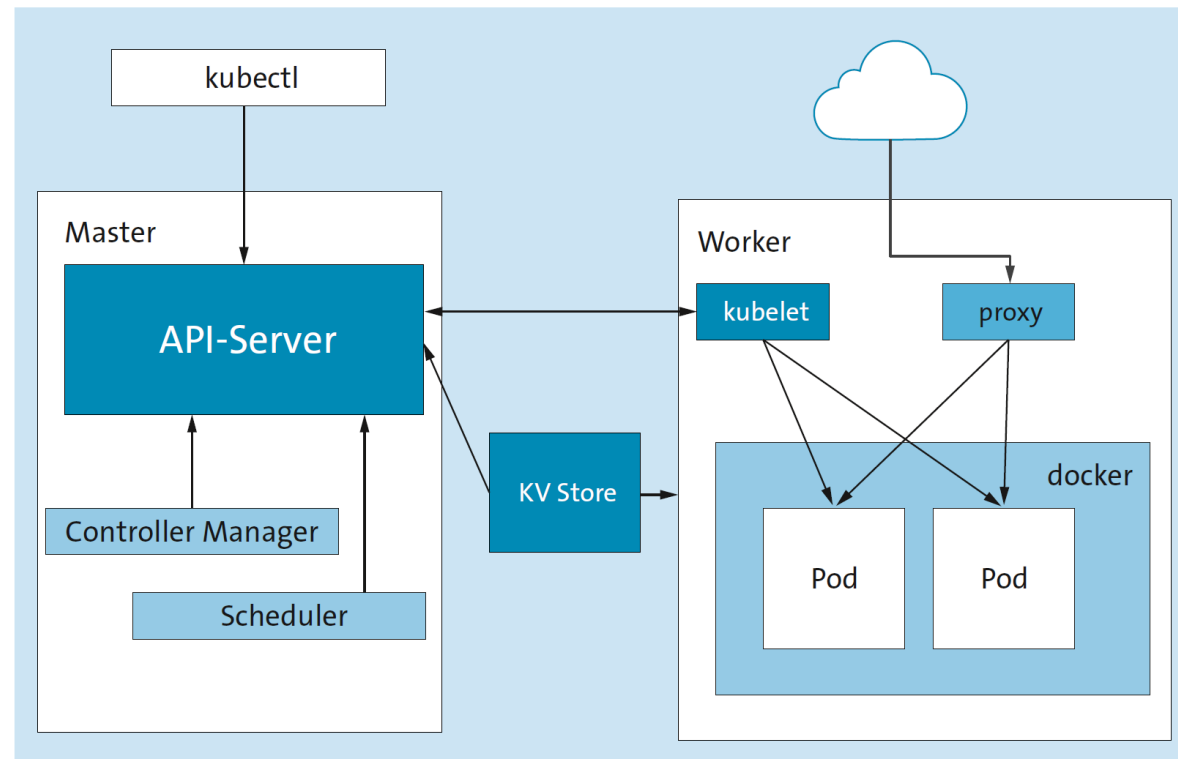


Abbildung 13.1 K8s Master und Worker aus der Vogelperspektive

★ Docker bzw. ein Container Runtime

- Die eigentliche Container-Plattform, mit der K8s arbeitet.

★ Kubelet (Cloud siehe

<https://kubernetes.io/docs/concepts/architecture/cloud-controller/>)

- Der kubelet-Service verbindet die Worker Nodes mit dem Kubernetes Master Node.

★ Kube-Proxy

- Der K8s-Netzwerk-Proxy – er arbeitet auf jedem Worker Node und kümmert sich um die eingehenden Requests und ihr Routing.
- Er kann als primitiver (Layer 4) Loadbalancer fungieren.

Networking in Kubernetes (1)

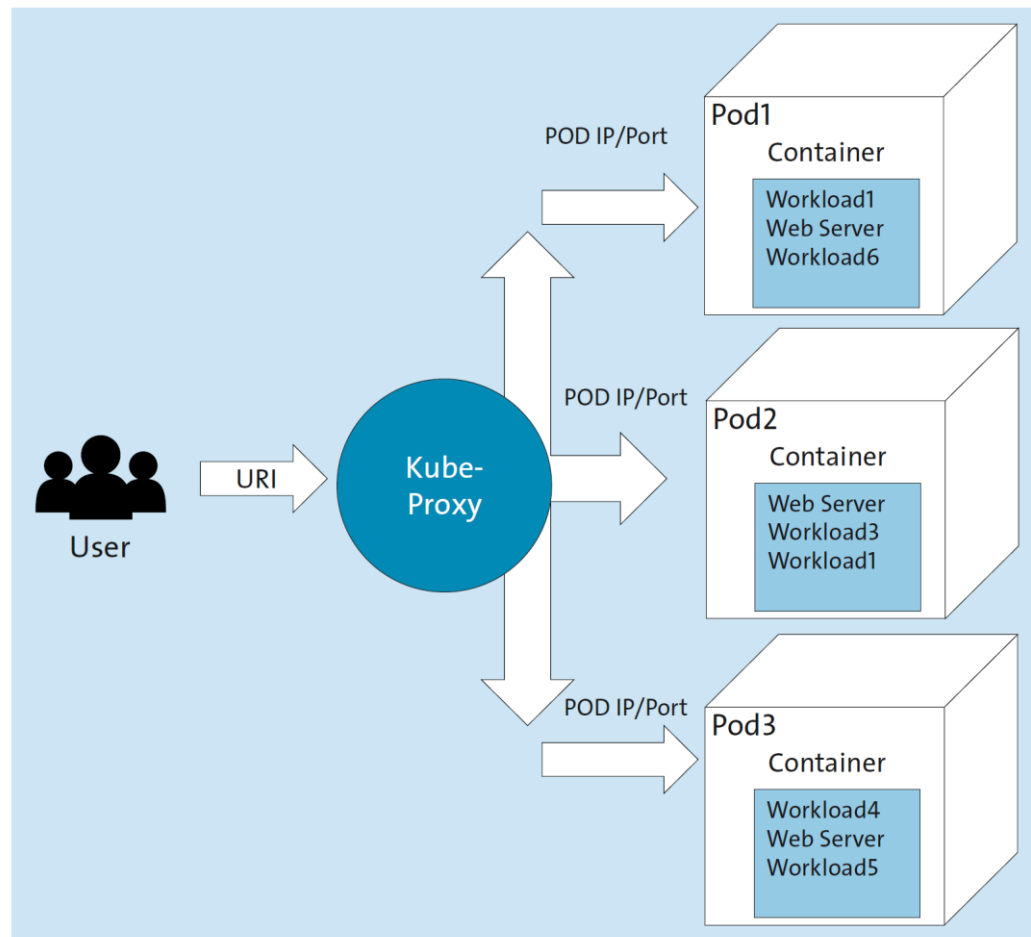


Abbildung 13.2 Arbeitsweise des Kube-Proxy exemplarisch auf einem K8s Worker Node

★ **Kubernetes** verwendet im Unterschied zu Docker eine **flache Netzwerkstruktur**:

- Jeder Container kann mit jedem anderen ohne NAT kommunizieren.
- Alle Kubernetes Nodes können mit allen Containern (und in die andere Richtung) ohne NAT kommunizieren.
- Die IP, die ein Container von sich selbst sieht, ist auch die, die jeder andere Node oder Container im Netz von ihm sieht.

Networking in Kubernetes (2)

```
r-01:~$ kubectl get pods --all-namespaces
NAME                                READY
coredns-5644d7b6d9-dsxh7            1/1
coredns-5644d7b6d9-ndwjg            1/1
etcd-master-01                       1/1
kube-apiserver-master-01             1/1
kube-controller-manager-master-01   1/1
kube-flannel-ds-amd64-pn94s          1/1
kube-proxy-j7fb2                     1/1
kube-scheduler-master-01             1/1
```

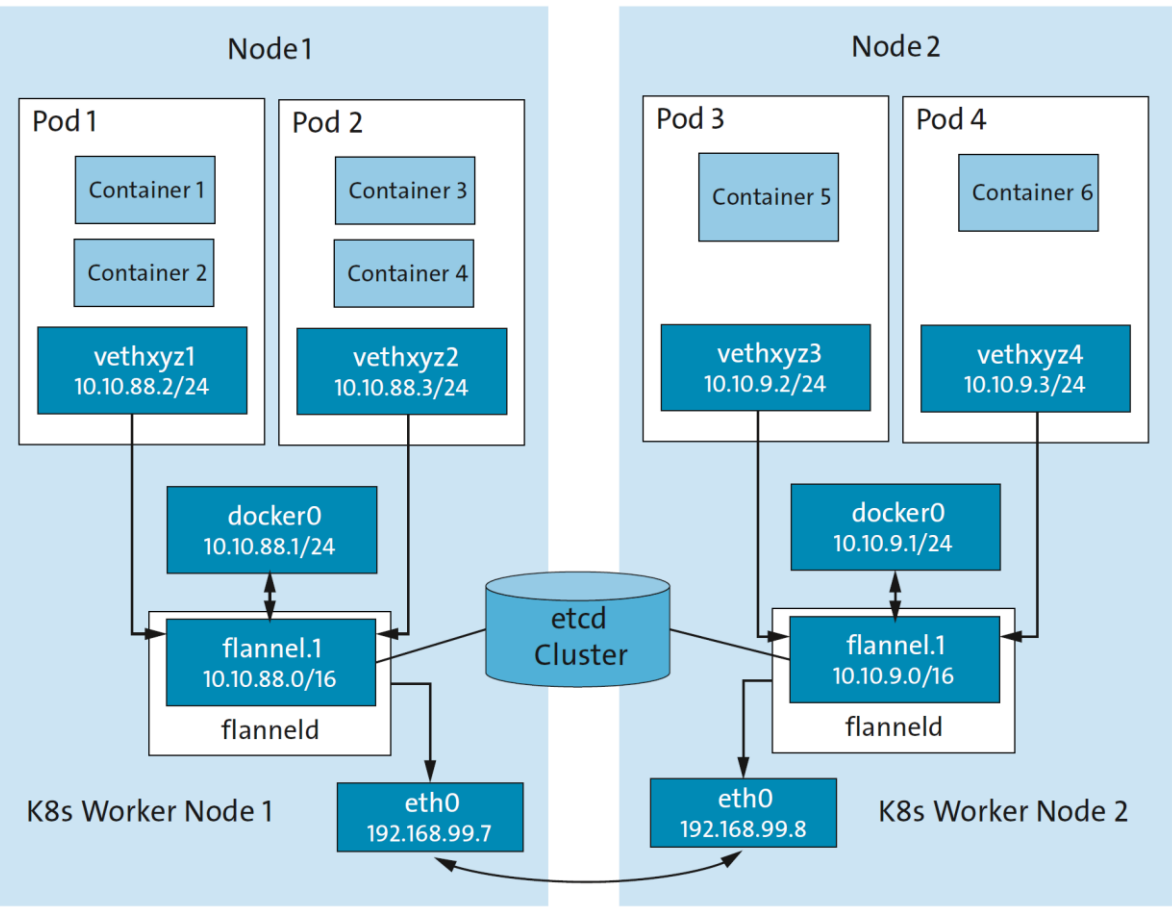


Abbildung 13.4 Schematisches Zusammenspiel der Netzwerkkomponenten

Quelle: Buch Skalierbare Container-Infrastrukturen, Kapitel 10.5

- ★ Was K8s zunächst fehlt, ist ein **Overlay-Netzwerk** welches die **Pods Node-übergreifend** für die **Interkommunikation** nutzen können.
- ★ Das Overlay-Netzwerk wird zwischen jedem K8s Node eingerichtet, sodass sich die Container/Pods auf verschiedenen Nodes direkt sehen bzw. miteinander kommunizieren können.
- ★ **Flannel** ist solch ein Overlay-Netzwerk, stammt aus der CoreOS-Schmiede, und fungiert als speziell für K8s designtes Netzwerk-Overlay.
- ★ Weitere Informationen:
 - [How to implement the Kubernetes networking model](#)
 - [CNI - the Container Network Interface](#)
 - [5G Core Deployment](#)

Networking in Kubernetes (3): Namespaces

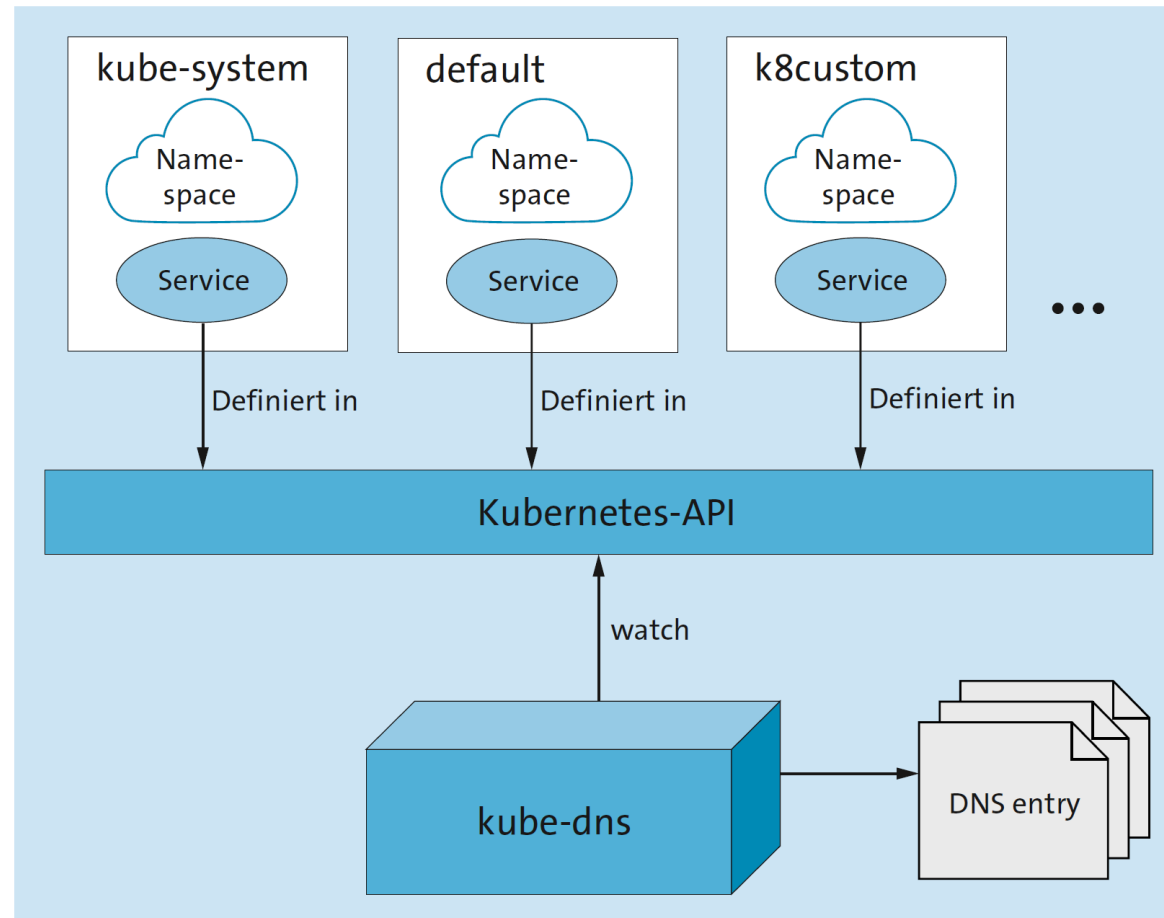


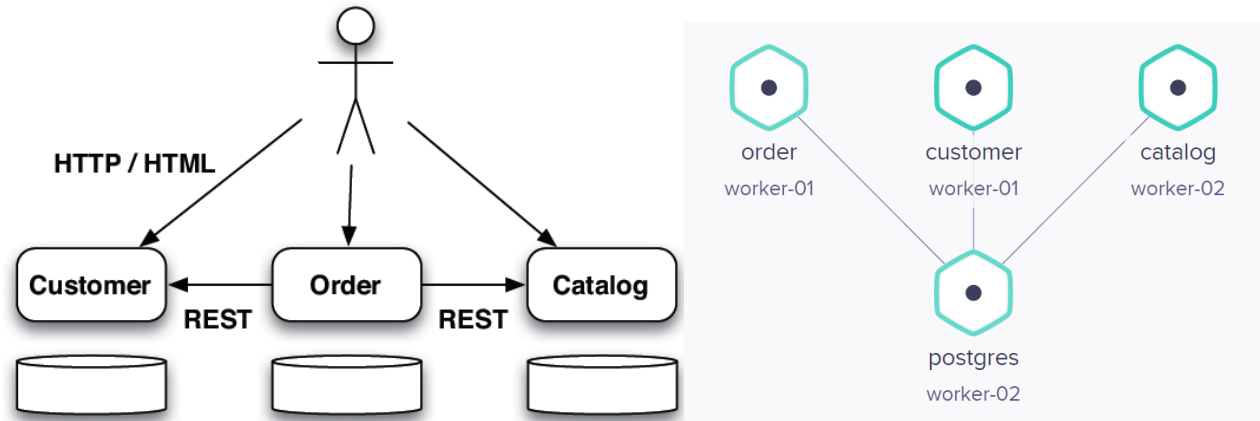
Abbildung 14.6 Zusammenspiel K8s DNS und Namespaces

- ★ Unterteilen den gesamten K8s Cluster in logische Partitionen bzw. Bereiche.
- ★ Standard Namespaces
 - **default:** In diesem Namespace werden alle neuen erzeugt, sofern wir nichts anderes angeben.
 - **kube-system:** Kubernetes-Systemdienste.
- ★ Ressourcen in Namespaces können limitiert werden (Anzahl Pod's, CPU, Memory etc.).
- ★ Der Netzwerktrafik kann, mittels [Network Policy Provider](#), eingegrenzt werden
- ★ **Achtung:** Wird ein Namespace gelöscht, werden damit auch alle in ihm gehosteten Objekte/Ressourcen gelöscht!

Networking in Kubernetes (4): Beispiel Microservices

★ Synchrone Microservices ([REST](#))

- Namespace: ms-kubernetes



- **Flache Netzwerkstruktur (auch bei unterschiedlichen Nodes)**

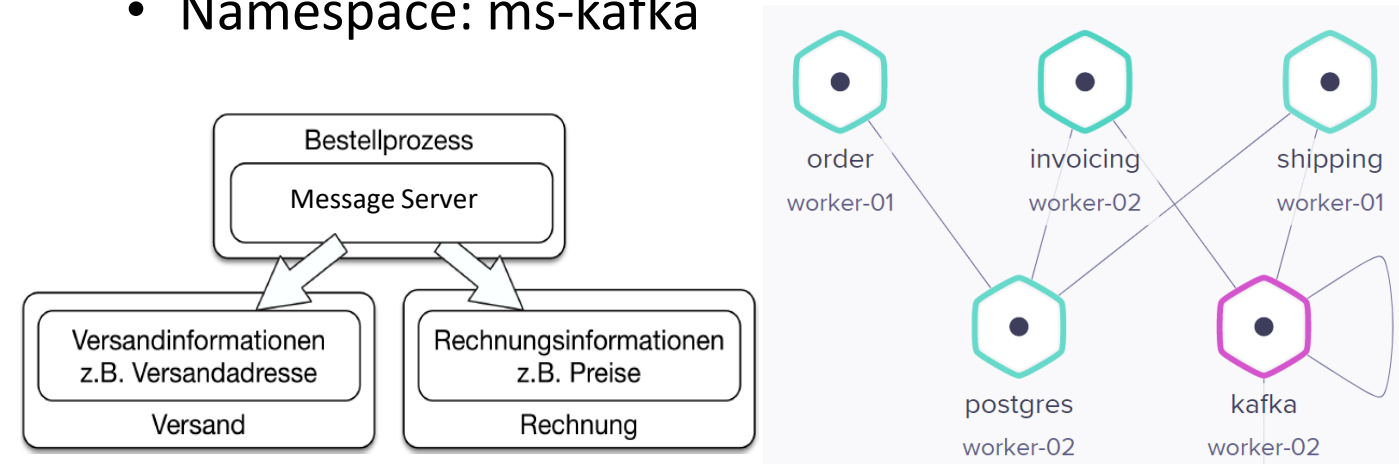
★ `ping order || catalog || customer -> ok`

- **Namespace (anderer Namespace)**

★ `ping invoicing -> schlägt fehl`

★ Asynchrone Microservices ([Messaging](#))

- Namespace: ms-kafka

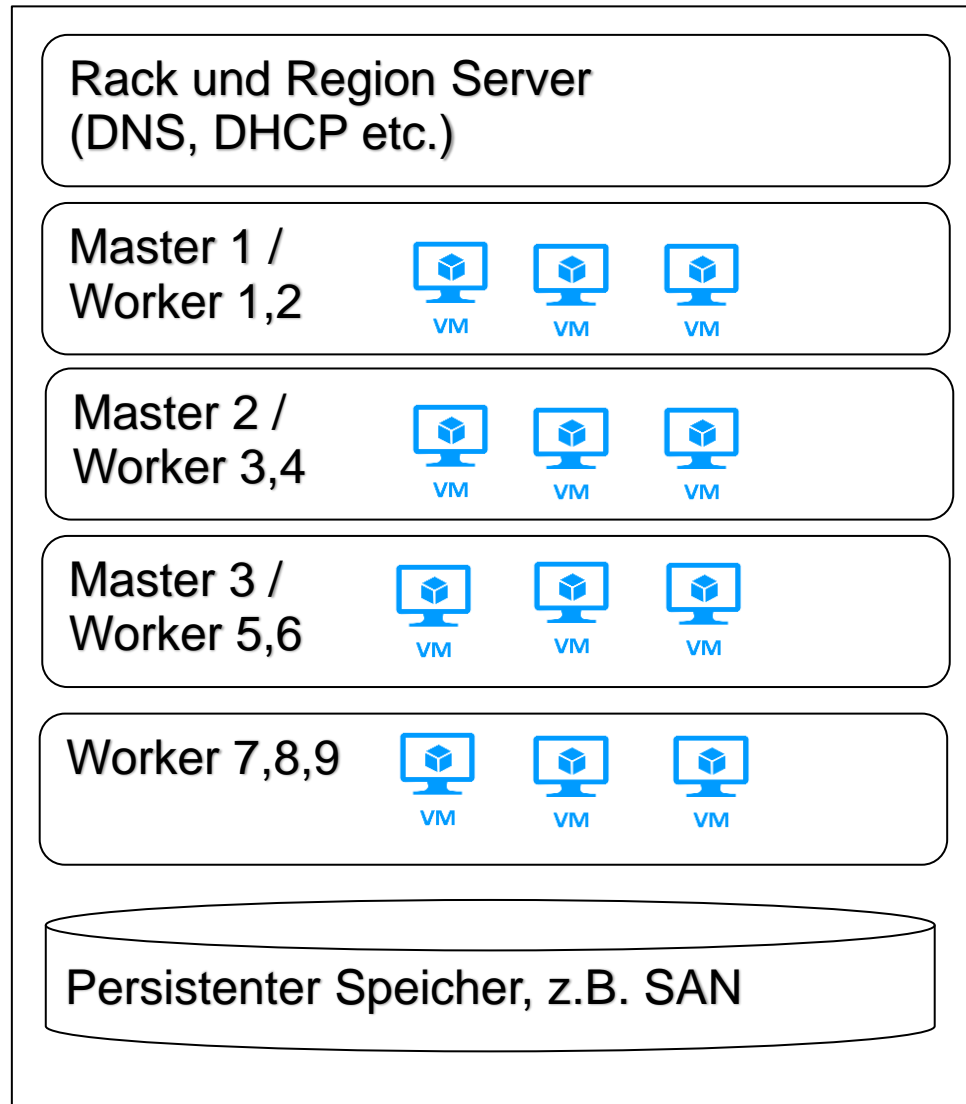


- **Namespace**

★ `ping order -> nur innerhalb Namespace`

★ `ping order.ms-kubernetes -> anderer NS`

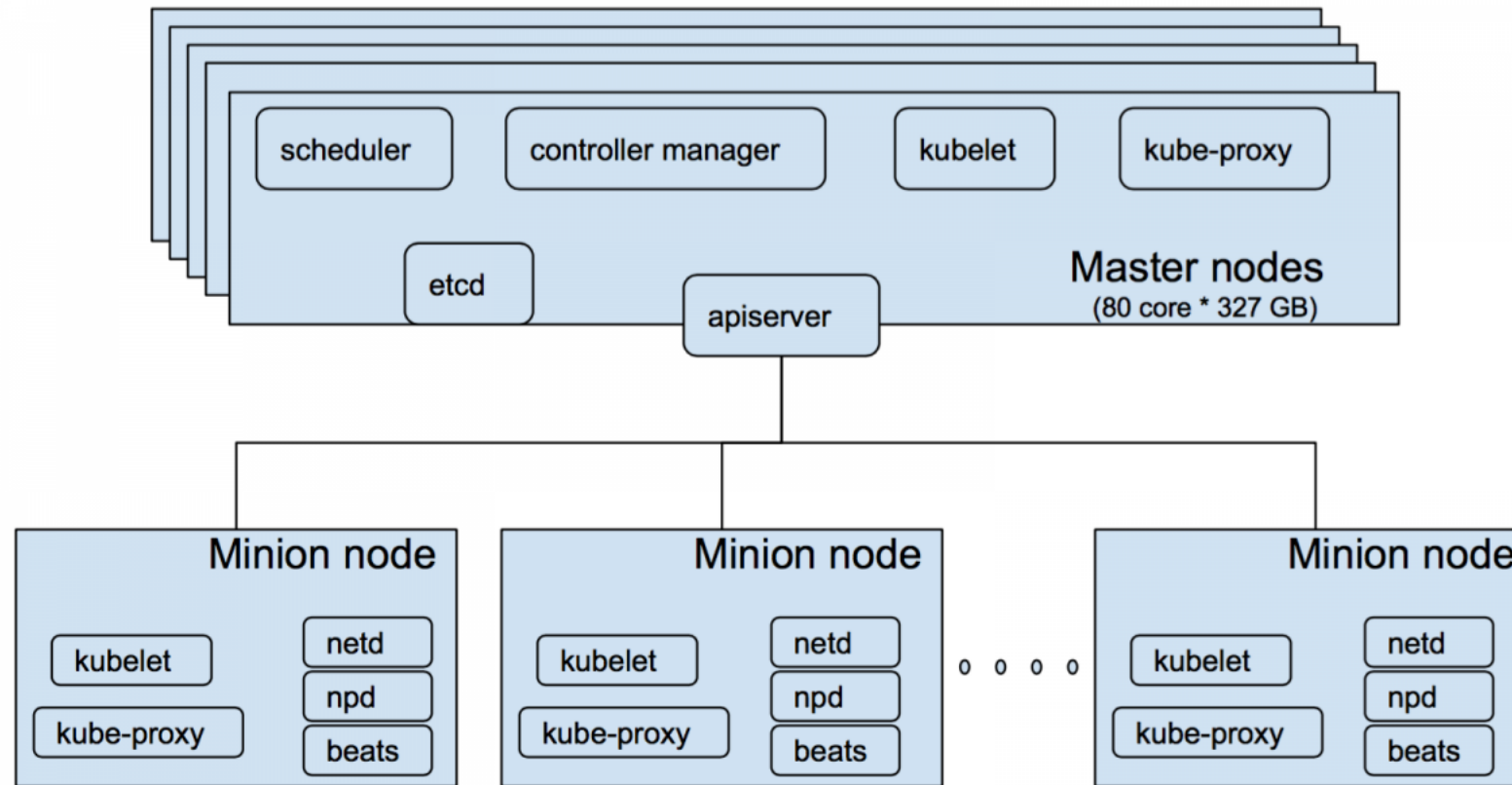
Beispiel: Kubernetes Cluster für 3 Mandanten



17" Rack (Serverschränk)

- ★ Es wird ein grosser Kubernetes Cluster eingerichtet.
- ★ Es werden mehrere Master, verteilt über die physikalischen Maschinen erstellt.
- ★ Die Worker Nodes werden über die physikalischen Maschinen verteilt.
- ★ Die Unterteilung nach Mandanten erfolgt mittels Kubernetes Namespaces.
 - K8s Namespace: MA
 - K8s Namespace: MB
 - K8s Namespace: MC

Wie viele Nodes kann ich zu einem Cluster verbinden?



- ★ Kubernetes behauptet das 5.000 Nodes in einem Cluster möglich sind.
- ★ Der eBay Tess.IO-Cluster bestätigt, nach der Optimierung / Korrektur, diese Behauptung.
- ★ Quelle: <https://tech.ebayinc.com/engineering/scalability-tuning-on-tess-io-cluster/>

Übung: Welche Distributionen und Hosted Kubernetes Umgebungen sind zertifiziert?

★ Öffnet die [CNCF Landscape](#) und sucht alle Kubernetes (K8s) zertifizierten Distributionen und Hosted Umgebungen:

Reset Filters

Grouping

Category

Sort By

Alphabetical (a to z)

Category

Platform, Certified Kub...

CNCF Relation

Any

License

Any

Organization

Any

Headquarters Location

Any

Example filters:

[Open source by age](#)

[Landscape categories](#)

[Open source by stars](#)

[Offerings from China](#)

[Certified K8s/KCSP/KTP](#)

[Sort by MCap/Funding](#)

CNCF Cloud Native Interactive Landscape

CNCF's Cloud Native Trail Map provides a good introduction. You can also view the static landscape and serverless landscapes. Please open the trail map.

You are viewing 135 cards with a total of 20,134 stars, market cap of \$4.56T and funding of \$1.45B.

Platform - Certified Kubernetes - Distribution (37)

Agile Stacks DevOps Automation Platform

Funding: \$2.53M

Alauda EE

Funding: \$15M

Diamanti Converged Container Infrastructure

Funding: \$43M

Docker EE/CE

Funding: \$243M

CNCF Members - Platinum (22)

Alibaba Cloud

Alibaba Cloud

MCap: \$175.00

AWS

Amazon Web Services

MCap: \$1.07

Apple

Apple

MCap: \$2.07

ARM

ARM Holdings

MCap: \$24.00

AT&T

AT&T

MCap: \$24.00

Cisco

Cisco

MCap: \$215.00

Fujitsu

Fujitsu

MCap: \$24.00

Google Cloud

Google

MCap: \$1.07

Huawei

Huawei Technologies

MCap: \$1.07

IBM Cloud

IBM

MCap: \$127.00

Intel

Intel

MCap: \$24.00

JD.COM

JD.com

MCap: \$115.00

Kasten

Kasten

Funding: \$11M

Microsoft Azure

Microsoft

MCap: \$17

NetApp

NetApp

MCap: \$17.00

New Relic

New Relic

MCap: \$4.00

ORACLE

Oracle

MCap: \$215.00

Palo Alto Networks

Palo Alto Networks

MCap: \$135.00

Red Hat

Red Hat

MCap: \$127.00

SAP

SAP

MCap: \$127.00

VMware

VMware

MCap: \$24.00

Volcano Engine

Volcano Engine

MCap: \$1.07

Sumo Logic

Sumo Logic

MCap: \$1.07

SUPER ORBITAL

SuperOrbital

MCap: \$1.07

SVA

SVA

MCap: \$1.07

swisscom

Swisscom

MCap: \$1.07

SYNADIA

Synadia

Funding: \$1M

sysdig

Sysdig

Funding: \$11M

SysEleven

SysEleven

Funding: \$1.07

TALOS SYSTEMS

Talos Systems

Funding: \$1.07

TEAMSUN

Teamsun

MCap: \$1.07

VS.

Docker, Inc.

Unternehmen

Aus dem Englischen übersetzt - Docker, Inc. ist ein amerikanisches Technologieunternehmen, das Docker Hub und Docker Desktop entwickelt, Entwicklerproduktivitätstools, die auf Docker basieren, einem Open-Source-Projekt, das die Bereitstellung von Code in Software-Containern automatisiert. [Wikipedia \(Englisch\)](#)

Ursprüngliche Beschreibung aufrufen ▾

Gründung: 2008

Hauptsitz: Palo Alto, Kalifornien, Vereinigte Staaten

Docker CE - Zukunft ungewiss?
Docker EE - verkauft an [Mirantis](#)
Kubernetes is [deprecating Docker](#)
(Ende Unterstützung **31.12.21**)

22

Welche Projekte sind sonst noch bei der CNCF?

CNCF (Kubernetes) Ökosystem Projekte

Graduated


Kubernetes
Orchestration


Prometheus
Monitoring


Envoy
Microservices


CoreDNS
DNS


containerd
Container Engine



Fluentd
Logging


Incubating


Jaeger
Distributed Tracing


OpenTracing
Distributed Tracing API


gRPC
Remote Procedure Call


CNI
Networking API


Notary
Security


TUF
Software Update Spec


Vitess
Storage


NATS
Messaging


Linkerd
Service Mesh


Helm
Package Management


Rook
Storage



Harbor
Registry



etcd
Key/Value Store


<https://www.cncf.io/projects/>


Sandbox Projects


EARLY STAGE



SPIFFE
Identity Spec



SPIRE
Identity



Telepresence
Tooling



OpenMetrics
Metrics Spec



Cortex
Monitoring



Buildpacks
Packaging Spec



Falco
Container Security



Dragonfly
Image Distribution



Virtual Kubelet
Nodeless



KubeEdge
Edge



Brigade
Scripting



Network Service Mesh
Networking



OpenTelemetry
Telemetry Specification


OpenEBS
Storage


Thanos
Monitoring


Flux
GitOps


in-toto
Security

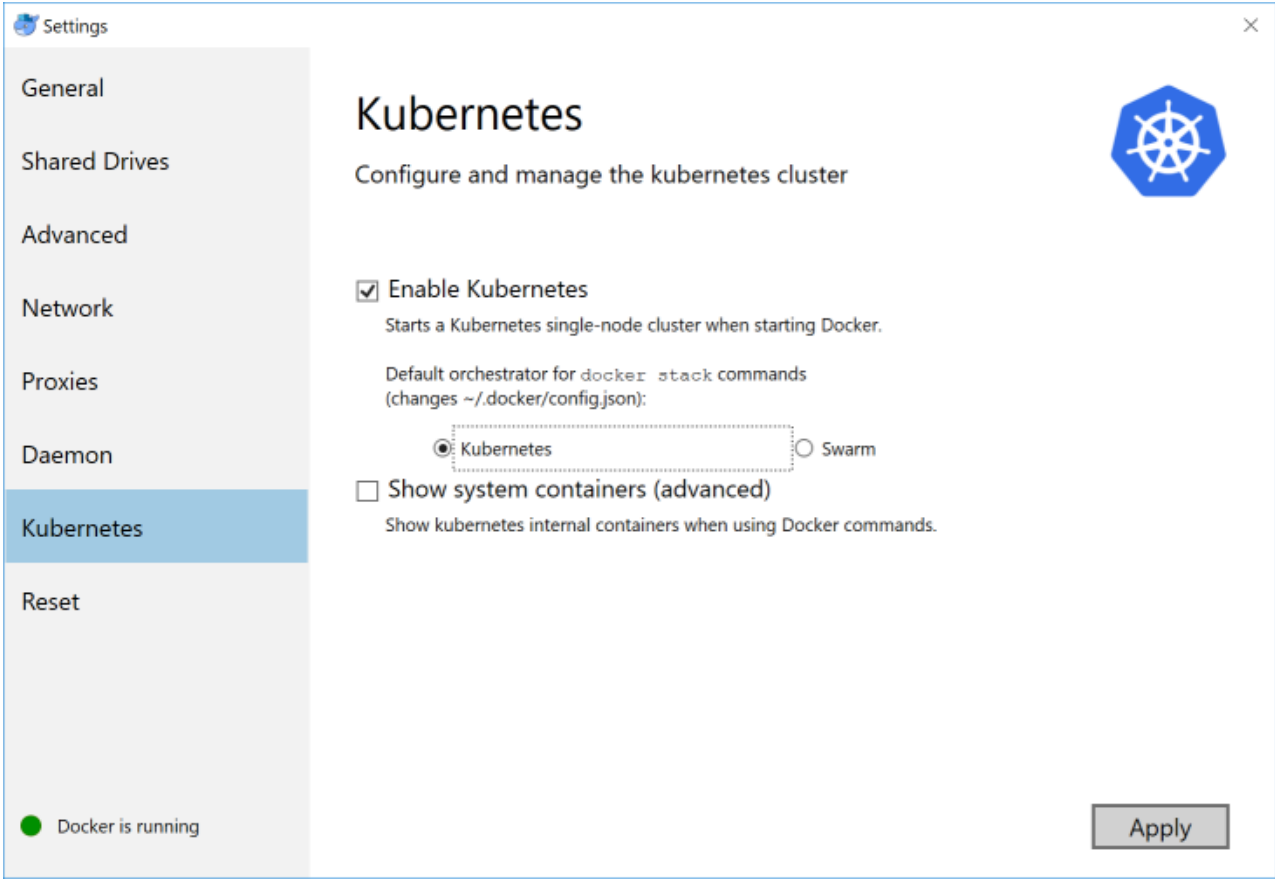

Strimzi
Kafka Operator

Was ist das Ziel der CNCF?

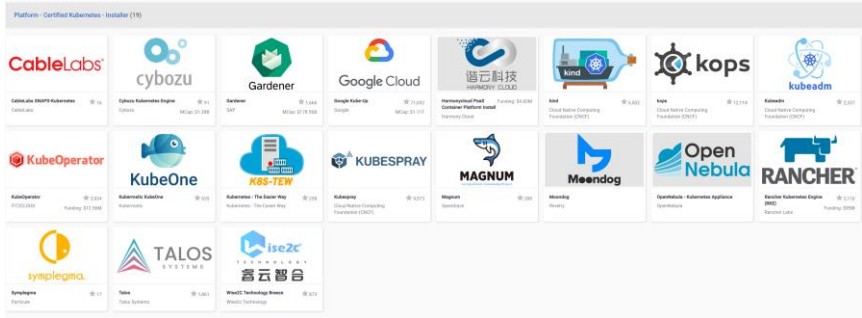
CNCF Cloud Native Definition v1.0

- ★ Cloud native Technologien ermöglichen es Unternehmen, skalierbare Anwendungen in modernen, dynamischen Umgebungen zu implementieren und zu betreiben. Dies können öffentliche, private und Hybrid-Clouds sein. Best-Practises, wie Container, Service-Meshs, Microservices, immutable Infrastruktur und deklarative APIs, unterstützen diesen Ansatz.
- ★ Die zugrundeliegenden Techniken ermöglichen die Umsetzung von entkoppelten Systemen, die belastbar, handhabbar und beobachtbar sind. Kombiniert mit einer robusten Automatisierung können Softwareentwickler mit geringem Aufwand flexibel und schnell auf Änderungen reagieren.
- ★ Die **Cloud Native Computing Foundation** fördert die Akzeptanz dieser Paradigmen durch die Ausgestaltung eines Open Source Ökosystems aus herstellerneutralen Projekten. Wir demokratisieren modernste und innovative Softwareentwicklungs-Patterns, um diese Innovationen für alle zugänglich zu machen.
- ★ Quelle: <https://github.com/cncf/toc/blob/master/DEFINITION.md>
- ★ Willkommen bei Cloud Land! Eine illustrierte Einführung in die [Cloud Native Landscape](#).
- ★ [Cloud-native](#) – was steckt hinter dem Buzzword?

Welche Installer gibt es für Kubernetes (K8s)?



- ★ Minikube (oder MiniShift): zum schnell K8s Luft Schnuppern. Nur Single Node.
- ★ «**Docker for Windows/Mac**»: für Entwickler. Nur Single Node (siehe links)
- ★ MicroK8s (Ubuntu) ist das kleinste, schnellste und vollständig konforme Kubernetes System.
- ★ kubeadm: Single Node bis HA Cluster
- ★ Zertifizierte Kubernetes Installer.



Setup Assistent von «Docker for Windows»
<https://github.com/mc-b/lernkube/tree/master/docker4windows>

Kubernetes Installation + join mittels microk8s

```
systemd(1) +-systemd-journal(383)
| -kube-scheduler(3921)
| -kube-controller(3983)
| -kube-apiserver(11389)
| -kube-proxy(11412)
| -kubenet(11482)
| -containerd(11295)
| -containerd-shim(13570) ---coredns(13600)
| -containerd-shim(12844) ---runsvdir(12872) ---runsv(12920) ---calico-node(12921)
```

★ Kubernetes (Master und Worker)

- `sudo snap install microk8s \`
`--classic --channel=1.19`
- `alias kubectl="microk8s kubectl"`
- `kubectl config view --raw >config`

★ Kubernetes (Master)

- `microk8s enable dns ingress ...`
- `kubectl apply -f persistentvolumes..`
- `micro8ks add-node`
- Ausgabe auf pro Worker Node ausführen

★ Die einzelnen Schritte für Master und Worker sind wie folgt:

- Installation Linux z.B. Ubuntu
- Installation minimale Kubernetes Umgebung (inkl. [containerd](#) und Overlay Netzwerk [calico](#)).
- `config` Datei für Remote Zugriff erstellen.
- Enablen der gewünschten Add-ons.
- Persistent Volumes (Storage) einrichten, z.B. [NFS](#).
- Master mit Worker (Master) Nodes verbinden.
- Ab 3 Nodes haben wir ein HA-Cluster!

Hands-on: Kubernetes installieren



- ★ <https://gitlab.com/ch-tbz-hf/Stud/cnt/-/tree/main/2> Unterrichtsressourcen/K#hands-on

Reflexion

- ★ Das im Juli 2014 gestartete Kubernetes (griechisch: Steuermann) stellt die derzeit **populärste** Container-Cluster-/Orchestrierungs-Lösung dar.
- ★ Von Single Node über Cluster, zu HA-Cluster zu Federated K8s werden praktisch alle Setup's unterstützt.
- ★ Das Container Ökosystem umfasst alle wichtigen Technologiefirmen und wird durch die Open-Source-Software-Stiftung «Cloud Native Computing Foundation» (CNCF) gefördert.
- ★ Es gilt heute, Dank der [CNCF](#), als **Industrie Standard**.

Lernzielkontrolle

- ★ Sie haben einen ersten Einblick in Kubernetes
- ★ Sie haben einen ersten Überblick über das Container Ökosystem.

Sind Geografisch verteilte Cluster möglich?

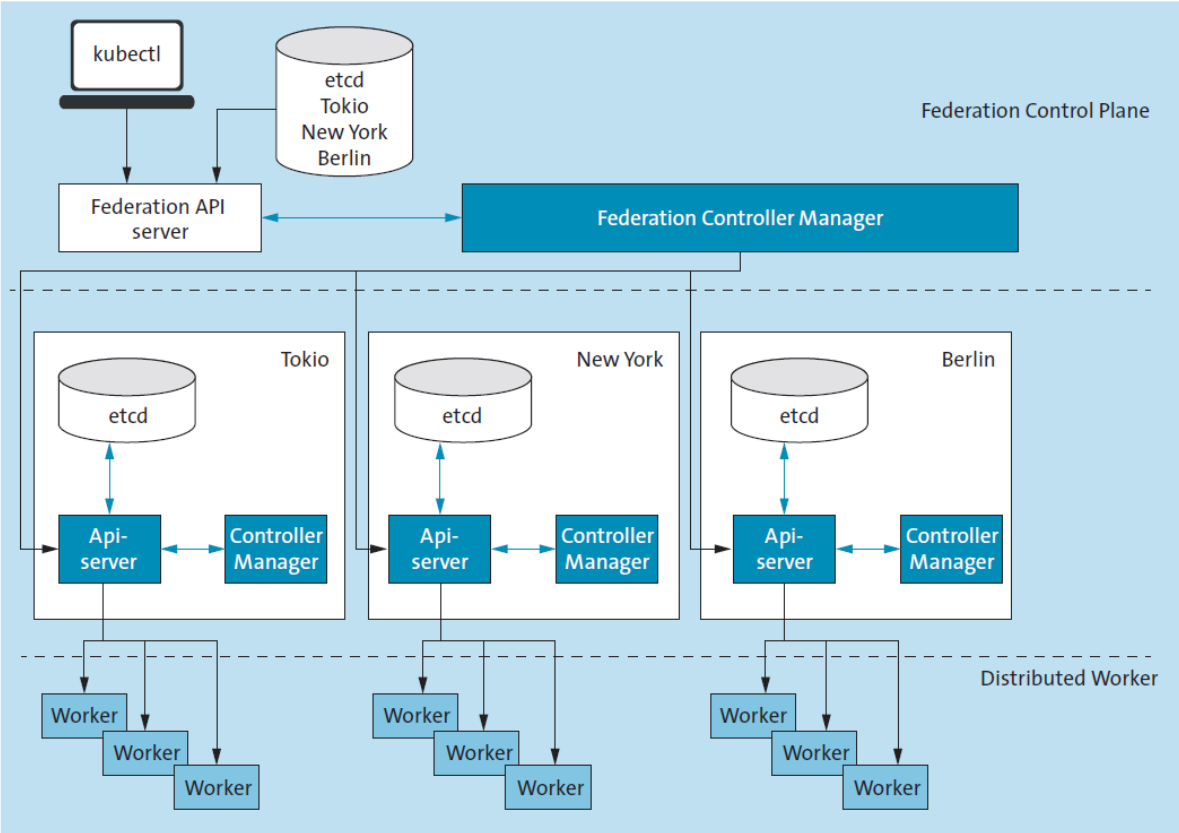


Abbildung 17.1 Federated K8s Cluster

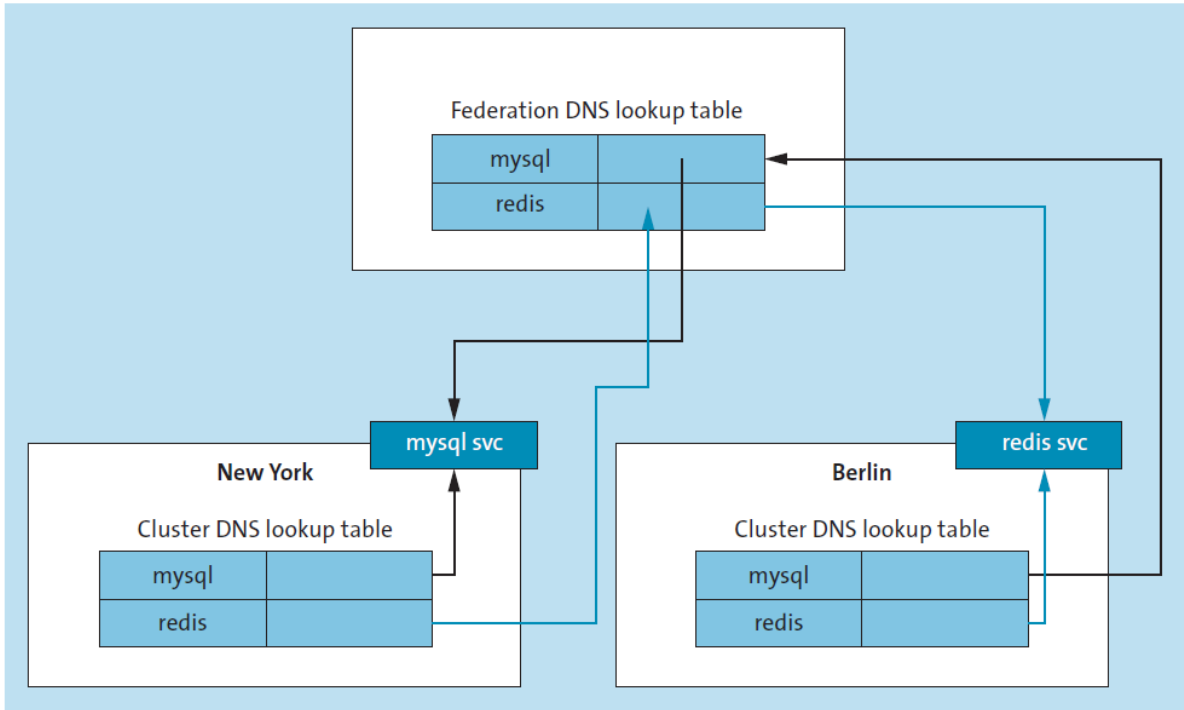
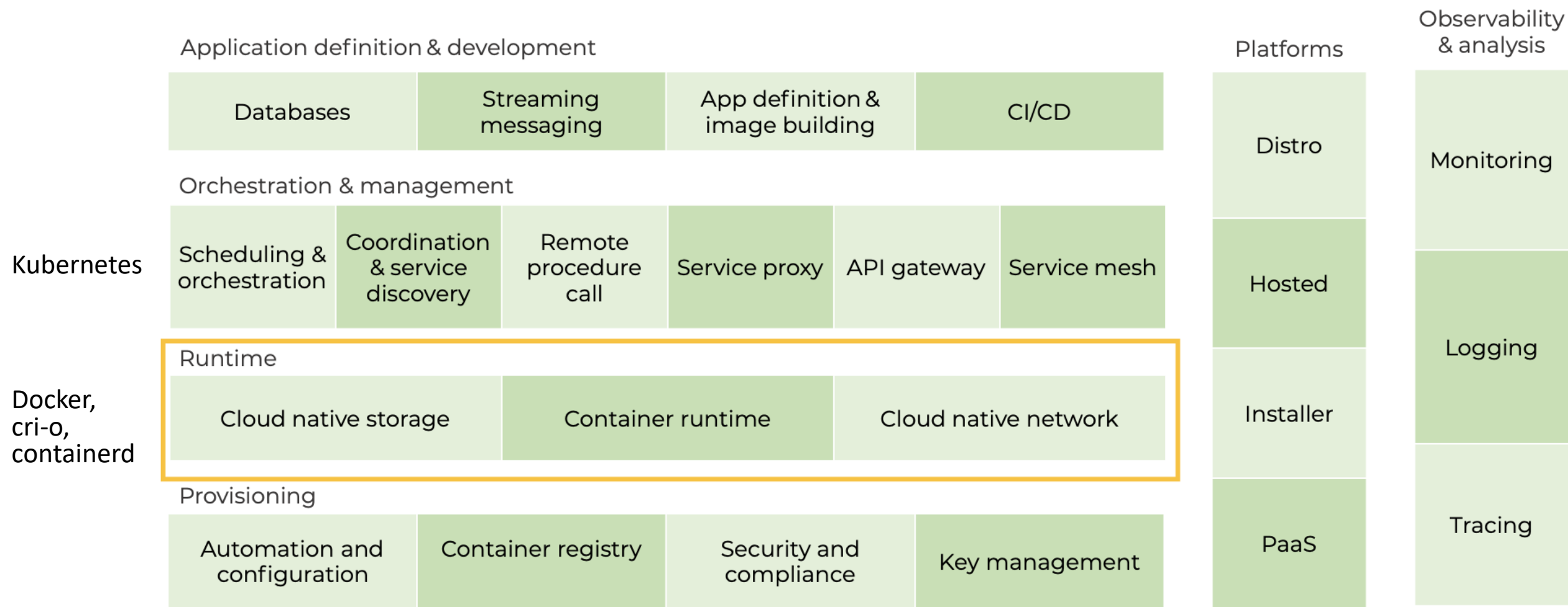


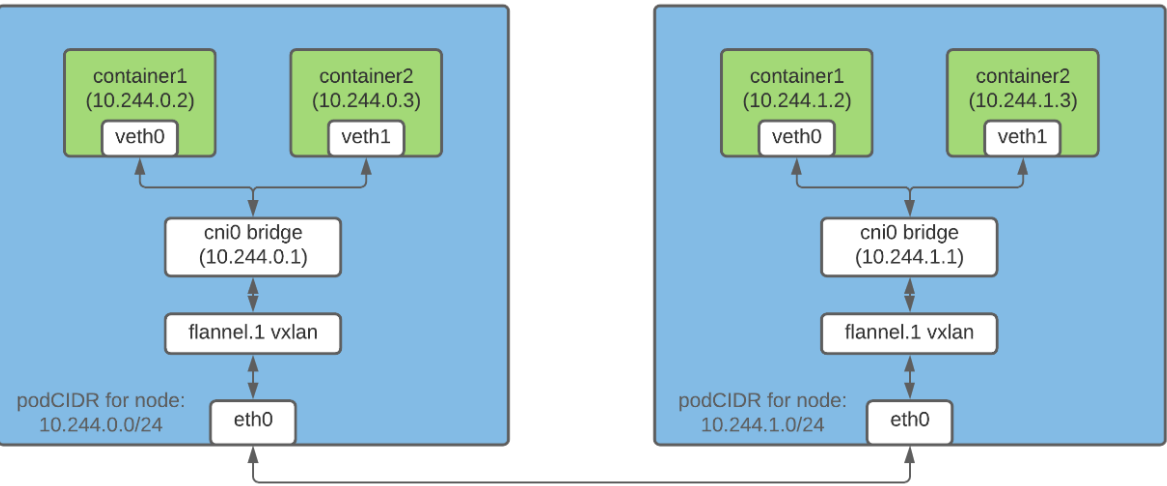
Abbildung 17.2 DNS mit Federated Setup

<https://kubernetes.io/docs/tasks/federation/federation-service-discovery/>

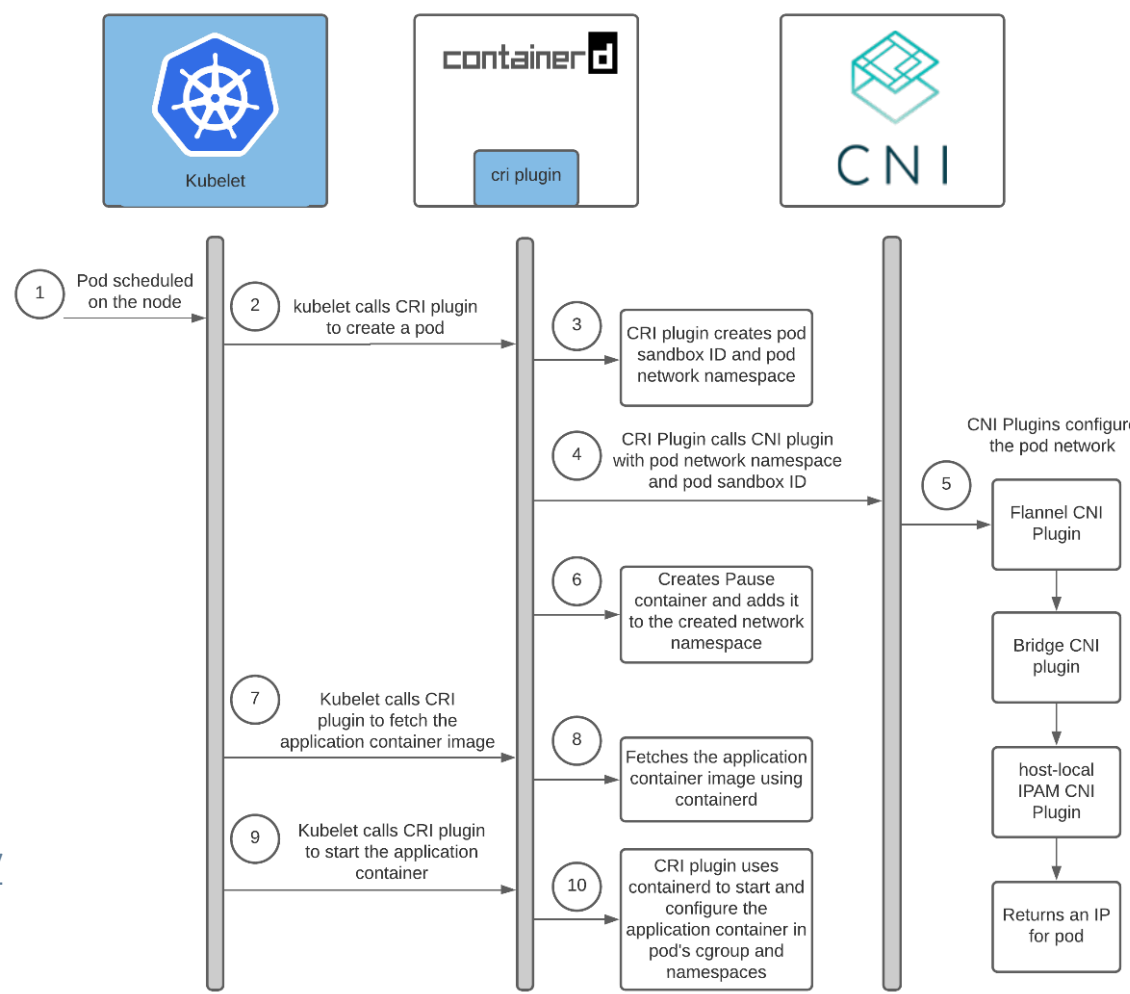
Container Runtime Layer



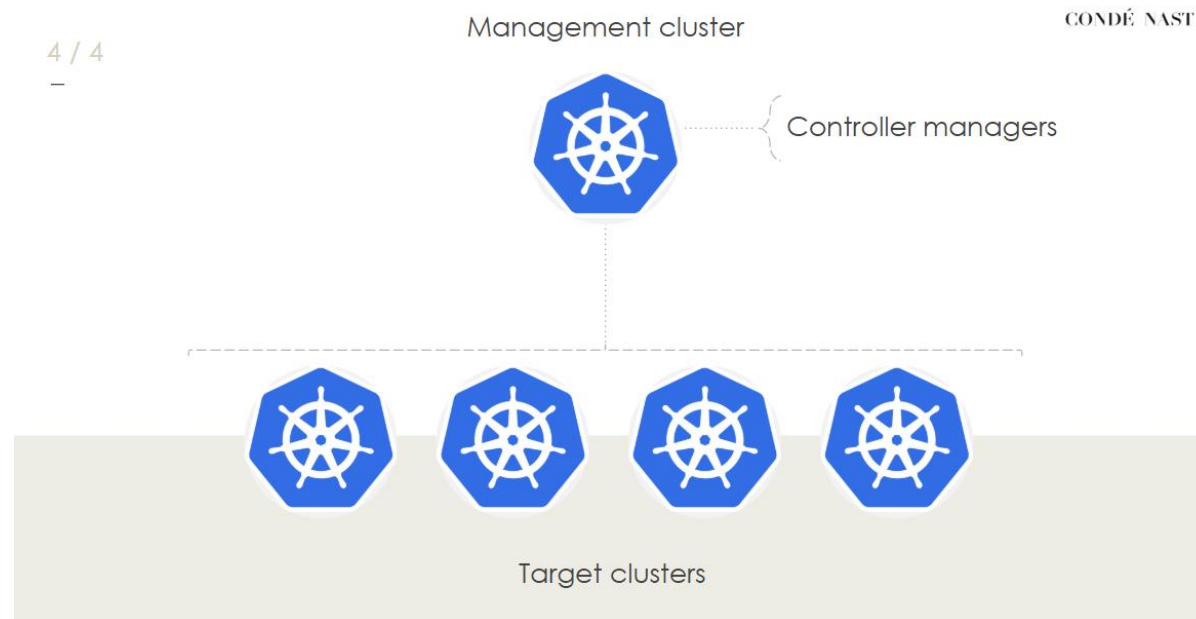
Networking in Kubernetes (Details)



Container im gleichen Worker: [vethX](#) auf [Linux Bridge](#).
Container auf verschiedenen Workern: z.B. Flannel vxlan und Routing Table.
<https://ronaknathani.com/blog/2020/08/how-a-kubernetes-pod-gets-an-ip-address/>



Wie Installiere ich einen Multicluster?



- ★ **ClusterAPI** bietet einen deklarativen Satz von APIs zum Erstellen, Konfigurieren, Verwalten und Löschen von Clustern.
- ★ Mit diesem Tool soll eine einheitliche und nachhaltige Schnittstelle für die Cluster-Initialisierung vor Ort und unterstützten Cloud-Anbietern bereitgestellt werden.
- ★ ClusterAPI ist derzeit in der Version **v1alpha2** und lässt sich mit 12 großen Infrastrukturanbietern integrieren .
- ★ <https://cluster-api.sigs.k8s.io/>

Wie ist die Sicht von Dev vs. Ops auf K8s?

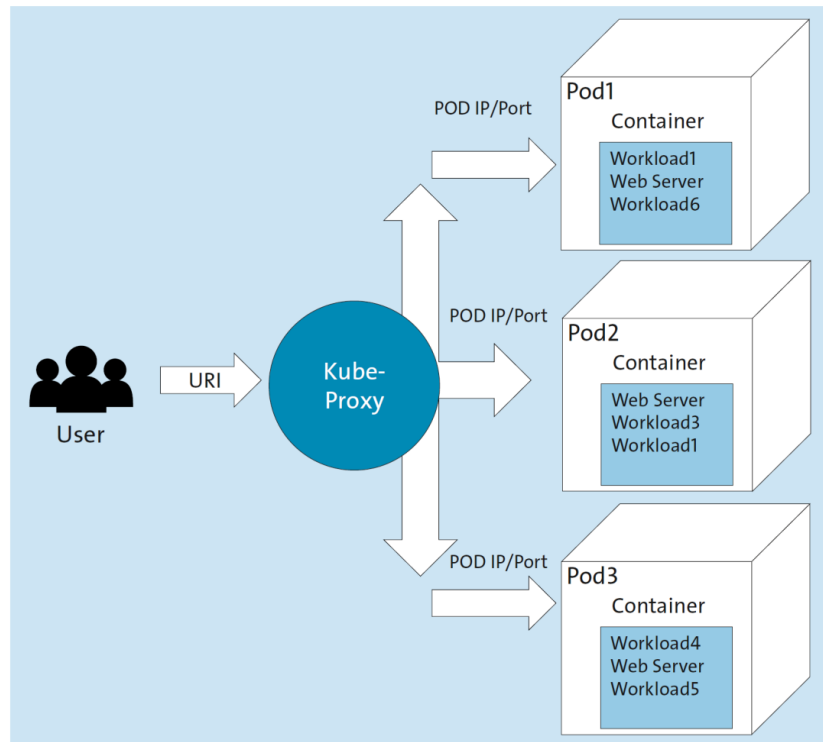
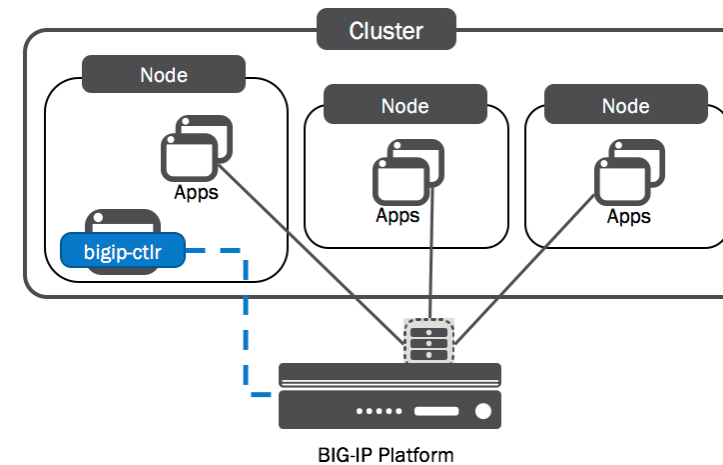


Abbildung 13.2 Arbeitsweise des Kube-Proxy exemplarisch auf einem K8s Worker Node

- ★ Anwendungsorientiertes statt Technisches Denken.
- ★ Abstraktion der Infrastruktur statt in Rechnern Denken.



User Guides

- [BIG-IP and flannel VXLAN Integration](#)
- [Add BIG-IP device to flannel VXLAN](#)
- [Install the BIG-IP Controller: Kubernetes](#)
- [Managed BIG-IP objects](#)
- [Attaching Virtual Servers to Services](#)
- [Using the BIG-IP Controller as an Ingress controller](#)
- [CIS and AS3 Extension integration](#)
- [Manage your BIG-IP virtual servers](#)
- [F5 Resources Explained](#)
- [Deploy iApps with the BIG-IP Controller](#)
- [Troubleshoot Your Kubernetes Deployment](#)

Quelle: <https://clouddocs.f5.com/containers/v2/kubernetes/>

Worker-Node: Scheduling and Eviction

- ★ Kubernetes bietet eine Vielzahl von Möglichkeiten um Pods (Container) auf den Worker Nodes zu platzieren.
- ★ Siehe dazu das Kapitel [Scheduling and Eviction](#) in der Kubernetes Dokumentation.
- ★ Dort finden sich auch Informationen um das Räumen und neu Starten von Pods (Containers), wenn eine Worker Node nicht mehr verfügbar ist, fein zu tunen.
- ★ Meistens fehlt der Eintrag `--pod-eviction-timeout=30s` in der Datei `/etc/kubernetes/manifests/kube-controller-manager.yaml` oder ist auf 300s (10 Minuten) gesetzt.
- ★ Nach Ändern der Datei wird automatisch der Controller Manager frisch gestartet und die Werte übernommen.

Gibt es Fallstudien für Kubernetes?

End User Case Studies

CNCF end users are telling their stories to help elevate the technical conversations to business objectives and challenges. These are real-world use cases about the impact cloud native projects are having on their business. Click on a box to read how that end user has used cloud native technologies and CNCF projects to help their business.

Filter the list of case studies:

All Industries

All Cloud Types

All Projects

All Challenges

All Countries

All Product Types

Reset filter

Displaying 64 case studies

Adform

READ THE PROMETHEUS CASE STUDY

Adform

READ THE KUBERNETES CASE STUDY

adidas

READ THE CASE STUDY

AlphaSense

READ THE CASE STUDY

Amadeus

READ THE CASE STUDY

Ancestry

READ THE CASE STUDY

Ant Financial

READ THE CASE STUDY

CERN

READ THE CASE STUDY

ricardo.ch

READ THE CASE STUDY

AppDirect

BlaBlaCar

BlackRock

Bloomberg

Filter the list of case studies:

All Industries

All Cloud Types

All Projects

All Challenges

Switzerland (2)

Displaying 2 of 64 case studies

Gibt es Schweizer Fallstudien?

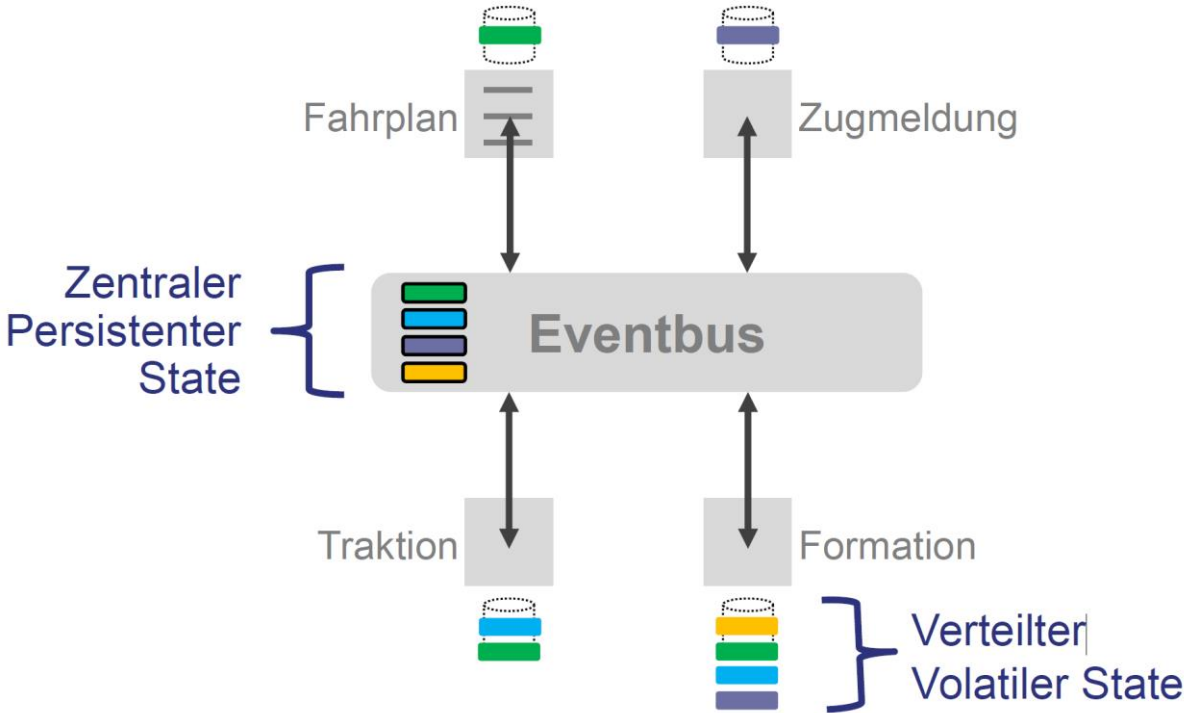
★ CERN

- Seit **Februar 2018** im Einsatz
- Kubernetes für die Orchestrierung, Helm für die Bereitstellung, Prometheus für die Überwachung und CoreDNS für die DNS-Auflösung innerhalb der Cluster
- Hat erlaubt **hybride Cloud-Strategie** einzuführen
- 330 Petabyte an Daten, diese können sich in den nächsten Jahren um das Zehnfache erhöhen
- Die Zeit für die Bereitstellung eines **neuen Clusters** für ein komplexes verteiltes Speichersystem ist von mehr als 3 Stunden auf weniger als **15 Minuten** gesunken ...

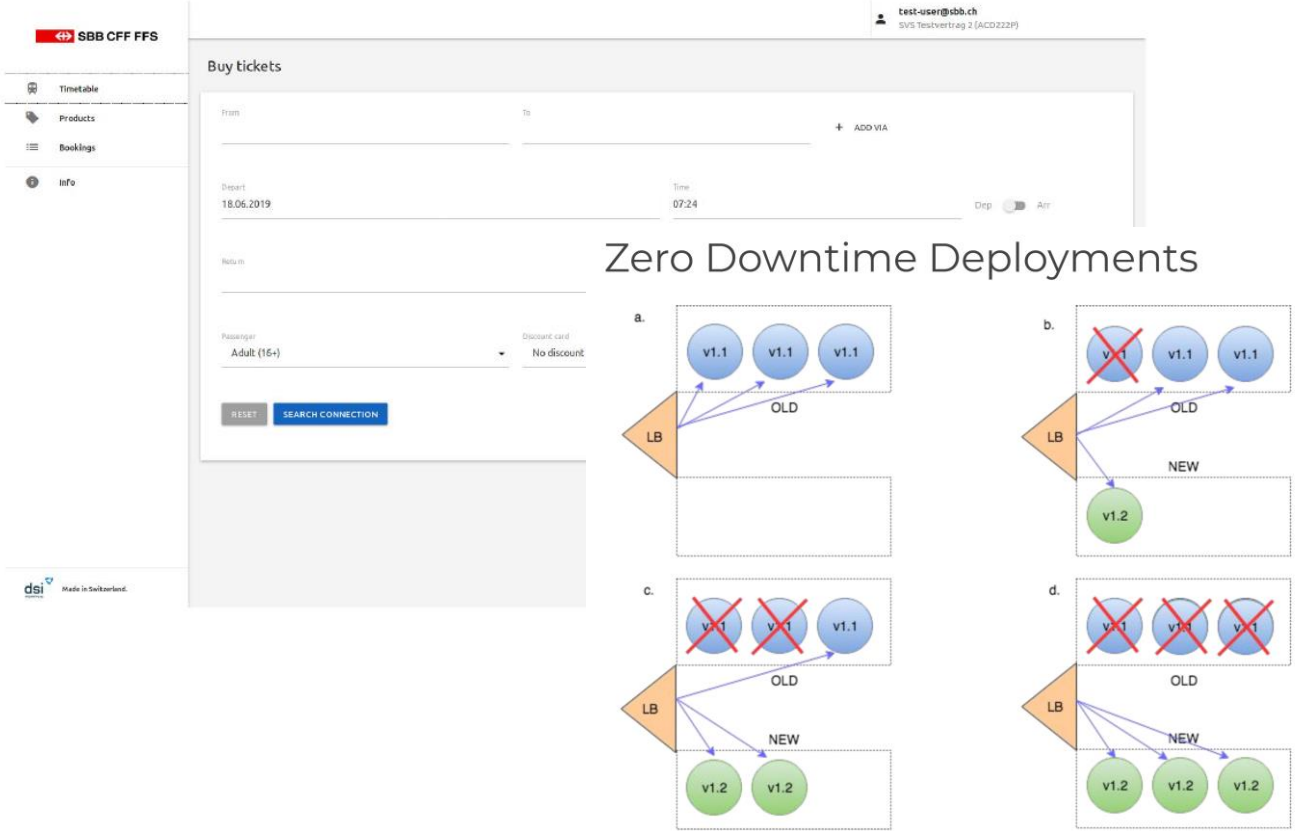
★ Ricardo

- Erster **Cluster im 2016**.
- Die Migration ist zur Hälfte abgeschlossen und das Unternehmen plant, bis Ende 2018 vollständig auf Google Cloud Platform umzustellen.
- Kubernetes für die Clusterverwaltung, Prometheus für die Überwachung und Fluentd für die Protokollierung.
- Die Aufteilung des Monolithen in Microservices ermöglichte eine höhere Geschwindigkeit neue Services bereitzustellen und **Kubernetes** war **entscheidend** bei der Umstellung.
- Das klassische Problem von erstellen, bearbeiten und ausliefern von neuen Versionen, konnte durch **30 - 60 Auslieferungen pro Tag statt 10 pro Woche** gelöst werden.

SBB (Messaging und K8s Worker Updates)



SBB Güterverkehr
Quelle: Event-basierte Architektur mit Apache Kafka
https://www.jug.ch/html/events/2019/architektur_mit_apache_kafka_bs.html



Agenturclient: Allows SBB business customers to sell tickets
Quelle: Real World Kubernetes
https://www.jug.ch/html/events/2019/kubernetes_live_zh.html