HSB
Hochschule Bremen
City University of Applied Sciences

ALFRED-WEGENER-INSTITUT
HELMHOLTZ-ZENTRUM FÜR POLAR- UND MEERESFORSCHUNG

Analyse der Sicherheit und der automatisierten Bereitstellung eines *On-Premises*-Clusters auf der Grundlage der Container-basierten Virtualisierung: *Kubernetes* im Wissenschaftsbetrieb

Masterarbeit im Studiengang Informatik (KSS)

Fabian Mangels <fabian@mangels.it>

Bremen, 05.10.2020

Gliederung

Einleitung

Herausforderungen

- Arbeits- und Forschungsprozesse basieren immer stärker auf Informations- und Kommunikationstechnologien (IKT)
- Einfluss der Digitalisierung auf die Methoden und das Vorgehen der WissenschaftlerInnen

- Maßnahme: Weiterentwicklung bzw. Neugestaltung der Forschungsinfrastrukturen

- Mitglied der *Helmholtz-Gemeinschaft*
- *Helmholtz Infrastructure for Federated ICT Services* (HIFIS)

Herausforderungen

- Arbeits- und Forschungsprozesse basieren immer stärker auf Informations- und Kommunikationstechnologien (IKT)
- Einfluss der Digitalisierung auf die Methoden und das Vorgehen der WissenschaftlerInnen

- Maßnahme: Weiterentwicklung bzw. Neugestaltung der Forschungsinfrastrukturen

- Mitglied der *Helmholtz-Gemeinschaft*
- *Helmholtz Infrastructure for Federated ICT Services* (HIFIS)

Szenario

- Aufbau von *On-Premises*-Cluster auf der Grundlage der Container-basierten Virtualisierung
- Fokus auf die zugrunde liegende Sicherheit und automatisierte Bereitstellung über mehrere involvierte Interaktionsplattformen
- *Kubernetes* (*K8s*) und *Docker*

Abbildung 1: Vereinfachter Überblick des Szenarios

Szenario

- Aufbau von *On-Premises*-Cluster auf der Grundlage der Container-basierten Virtualisierung
- Fokus auf die zugrunde liegende Sicherheit und automatisierte Bereitstellung über mehrere involvierte Interaktionsplattformen
- *Kubernetes* (*K8s*) und *Docker*

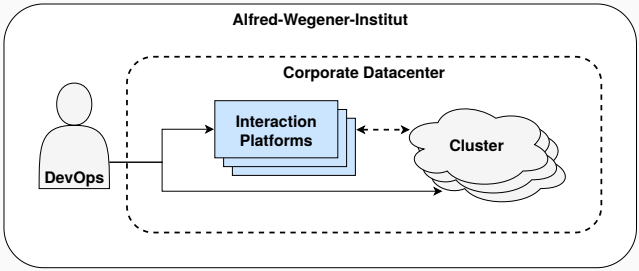
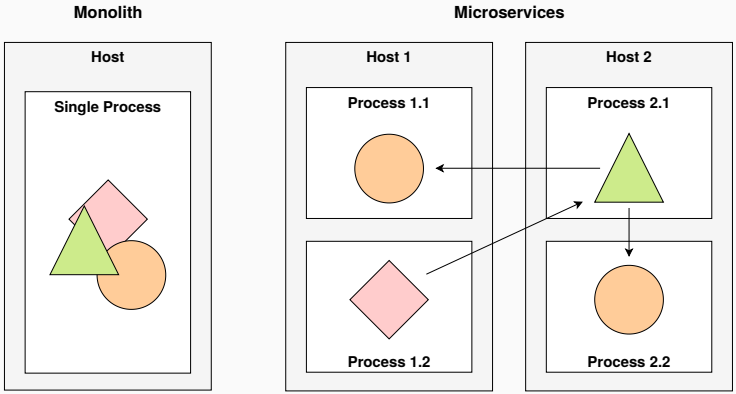


Abbildung 1: Vereinfachter Überblick des Szenarios

Grundlagen



- *Microservice*-Architekturstil
- Entkopplung vom Gesamtsystem
- Einzelne Entwicklung, Bereitstellung, Aktualisierung und Skalierung

Abbildung 2: Gegenüberstellung eines Monolithen und dessen Aufspaltung in *Microservices* [Luk18, vgl. S. 5]

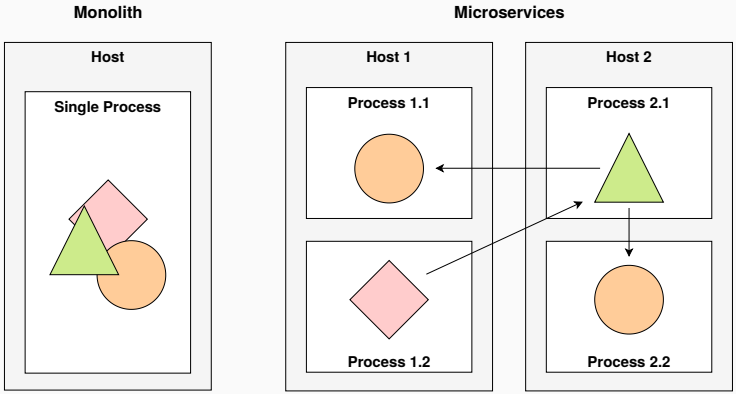


Abbildung 2: Gegenüberstellung eines Monolithen und dessen Aufspaltung in *Microservices* [Luk18, vgl. S. 5]

- *Microservice-*Architekturstil
- Entkopplung vom Gesamtsystem
- Einzelne Entwicklung, Bereitstellung, Aktualisierung und Skalierung

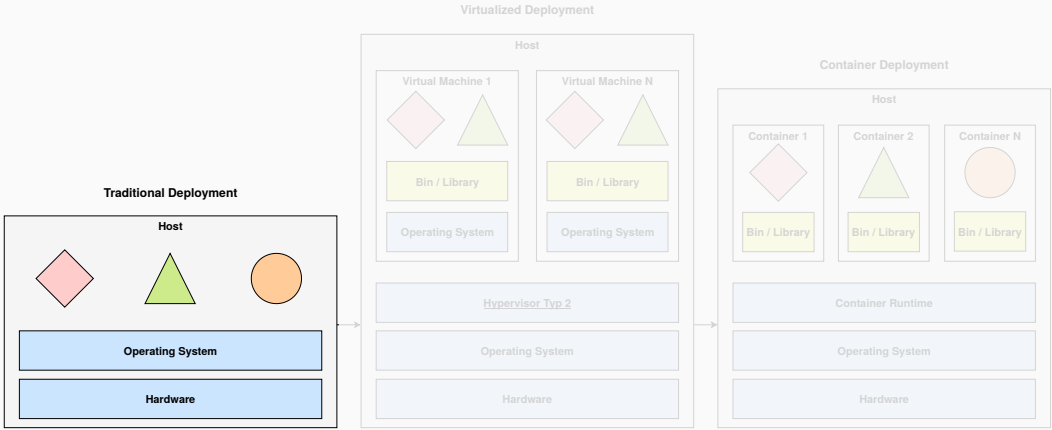


Abbildung 3: Entwicklung der Bereitstellungsarten auf einem einzelnen Host [Kub20f], [WAG+18, vgl. S. 40], [Lie19, vgl. S. 114]

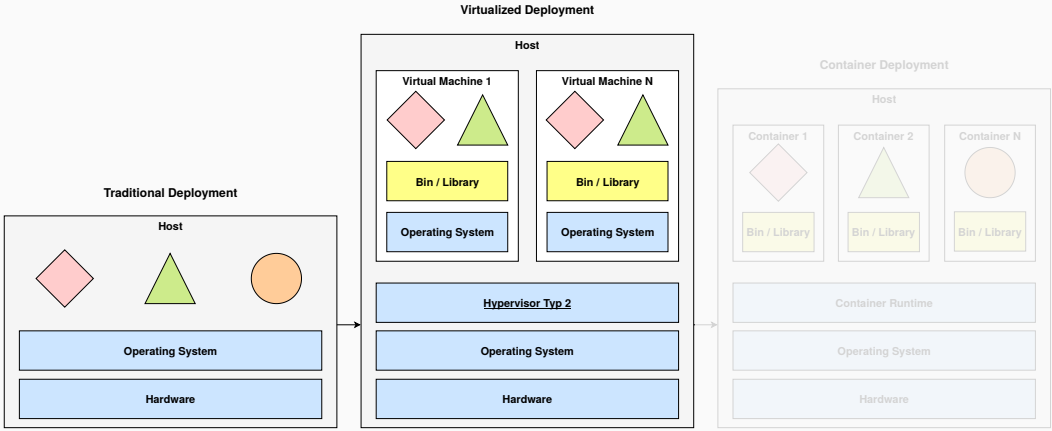


Abbildung 3: Entwicklung der Bereitstellungsarten auf einem einzelnen Host [Kub20f], [WAG+18, vgl. S. 40], [Lie19, vgl. S. 114]

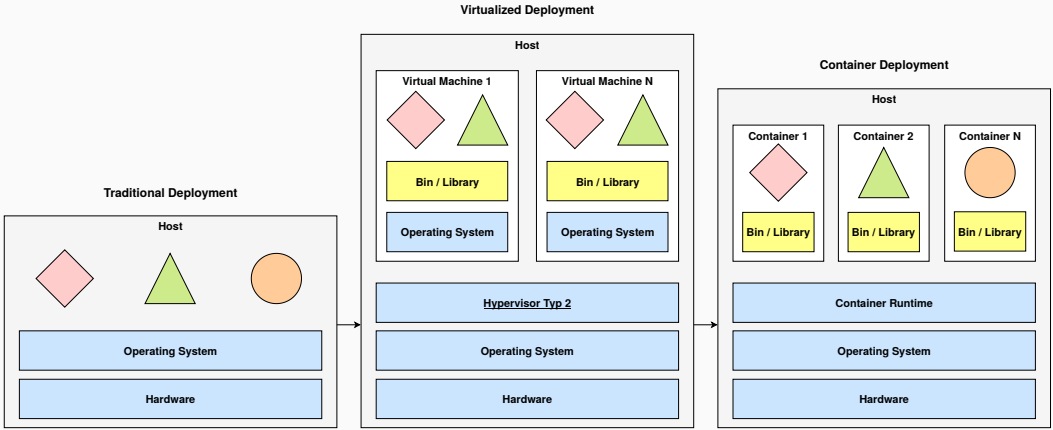


Abbildung 3: Entwicklung der Bereitstellungsarten auf einem einzelnen Host [Kub20f], [WAG+18, vgl. S. 40], [Lie19, vgl. S. 114]

Dienstkomposition

- Bedarf wegen Fortentwicklung zu *Microservice*-Architekturen
- Skalierbarkeit der heterogenen Bestandteile
- Schaffung einer konsistenten Bereitstellungsplattform
- Übergang zu *Continuous Delivery* (CD) und zur *DevOps*-Arbeitsweise

Abbildung 4: Container-Bereitstellung durch Orchestrierung als Dienstkomposition [Luk18, vgl. S. 20]

Dienstkomposition

- Bedarf wegen Fortentwicklung zu *Microservice*-Architekturen
- Skalierbarkeit der heterogenen Bestandteile
- Schaffung einer konsistenten Bereitstellungsplattform
- Übergang zu *Continuous Delivery* (CD) und zur *DevOps*-Arbeitsweise

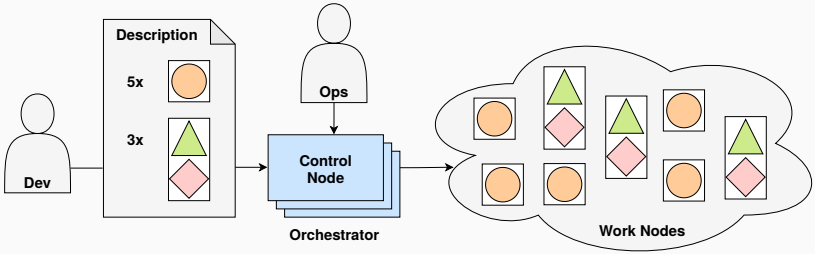
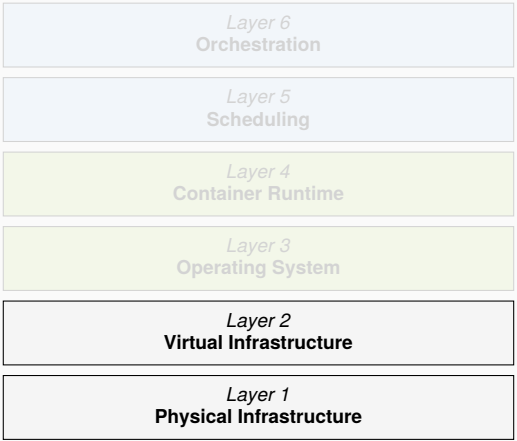


Abbildung 4: Container-Bereitstellung durch Orchestrierung als Dienstkomposition [Luk18, vgl. S. 20]

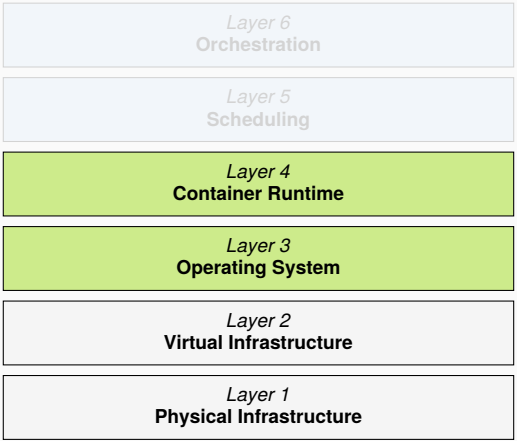
Skalierbare Container-Cluster



- *On-Premise* vs. *Cloud Computing*
- *CaaS* als Service-Modell
- *DevOps*
- *Cloud Native*
- Maximale Automatisierung durch CI / CD

Abbildung 5: (Vereinfachte) Schichten der Container-Welt [Lie19, vgl. S. 79, 507]

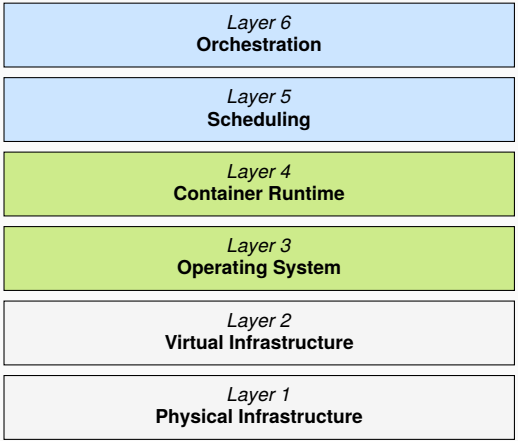
Skalierbare Container-Cluster



- *On-Premise* vs. *Cloud Computing*
- *CaaS* als Service-Modell
- *DevOps*
- *Cloud Native*
- Maximale Automatisierung durch CI / CD

Abbildung 5: (Vereinfachte) Schichten der Container-Welt [Lie19, vgl. S. 79, 507]

Skalierbare Container-Cluster



- *On-Premise* vs. *Cloud Computing*
- *CaaS* als Service-Modell
- *DevOps*
- *Cloud Native*
- Maximale Automatisierung durch CI / CD

Abbildung 5: (Vereinfachte) Schichten der Container-Welt [Lie19, vgl. S. 79, 507]

Skalierbare Container-Cluster

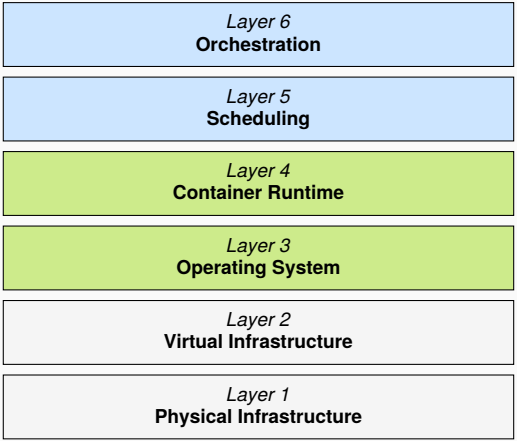


Abbildung 5: (Vereinfachte) Schichten der Container-Welt [Lie19, vgl. S. 79, 507]

- *On-Premise* vs. *Cloud Computing*
- *CaaS* als Service-Modell
- *DevOps*
- *Cloud Native*
- Maximale Automatisierung durch CI / CD

Anforderungen

Funktionale Anforderungen

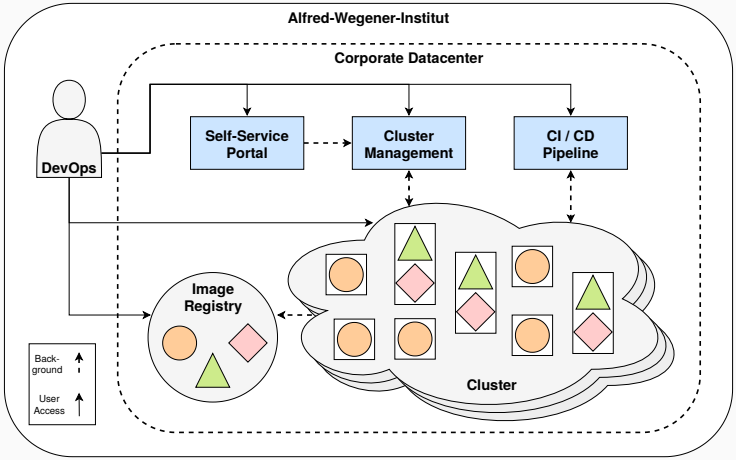


Abbildung 6: Kontextdiagramm des Szenarios

- *Image Registry*
- *Cluster Management*
- *Self-Service Portal*
- *CI / CD Pipeline*

Nicht-funktionale Anforderungen

- **Sicherheit** – Stand der Technik, *Best Practices*
- **Verwendung vorhandener Systeme** – *VMware vSphere*, *vRealize Suite*, *GitLab*
- **Plattformunabhängigkeit** – (*Ubuntu*) *Linux*, *Apple Mac OS X*, *Microsoft Windows*
- **Wartbarkeit** – Aktualisierung, Konfiguration, zentrales Monitoring, Wartung im laufenden Betrieb
- **Anwendungsbereitstellung** – Intuitive Benutzeroberfläche, *CI / CD Pipeline*
- **Verfügbarkeit** – Hochverfügbare und ausfallsichere Architektur, *Load Balancer*
- **Lizenzkosten, *Open Source*** – Vermeidung von Kosten
- **Knoten-Betriebssystem** – Angepasste Betriebssysteme, *Cloud Native*
- ***Container Runtime*** – Vorwiegend *Docker*
- **Föderierte Cluster** – Zusammenarbeit fördern (HIFIS)
- **Einbindung von *Cloud*-Anbietern** – AWS, Azure, GCP

Nicht-funktionale Anforderungen

- **Sicherheit** – Stand der Technik, *Best Practices*
- **Verwendung vorhandener Systeme** – *VMware vSphere*, *vRealize Suite*, *GitLab*
- **Plattformunabhängigkeit** – (*Ubuntu*) *Linux*, *Apple Mac OS X*, *Microsoft Windows*
- **Wartbarkeit** – Aktualisierung, Konfiguration, zentrales Monitoring, Wartung im laufenden Betrieb
- **Anwendungsbereitstellung** – Intuitive Benutzeroberfläche, *CI / CD Pipeline*
- **Verfügbarkeit** – Hochverfügbare und ausfallsichere Architektur, *Load Balancer*
- **Lizenzkosten, *Open Source*** – Vermeidung von Kosten
- **Knoten-Betriebssystem** – Angepasste Betriebssysteme, *Cloud Native*
- ***Container Runtime*** – Vorwiegend *Docker*
- **Föderierte Cluster** – Zusammenarbeit fördern (HIFIS)
- **Einbindung von *Cloud*-Anbietern** – AWS, Azure, GCP

Analyse

Sicherheit

- „Informationssicherheit hat den Schutz von Informationen als Ziel. [...] Die Schutzziele oder auch Grundwerte der Informationssicherheit sind Vertraulichkeit, Integrität und Verfügbarkeit“ [BSI20a, vgl. S. 37]

- IT-Grundschutz-Kompendium [BSI20a] vom Bundesamt für Sicherheit in der Informationstechnik (BSI)
- Baustein *SYS.1.6: Container* (*Community Draft*, Stand: 19.03.2020) [BSI20b]

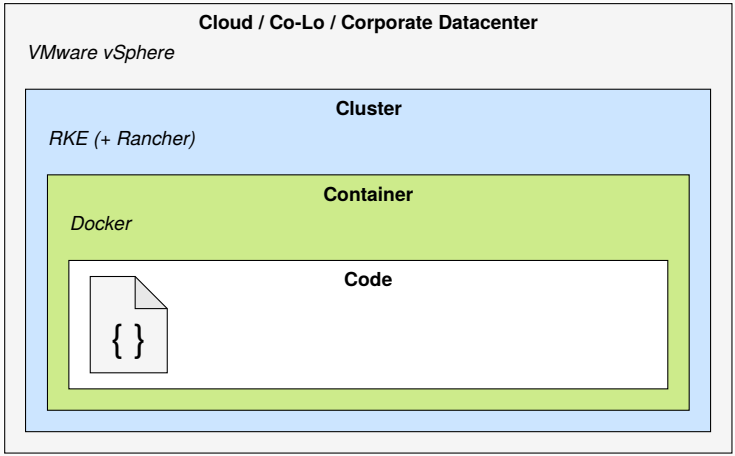


Abbildung 7: *The 4C's of Cloud Native Security* [Kub20e]

- *„Defense in Depth"*-Konzept
- *„Vanilla Kubernetes"*
- *Kubernetes Security Whitepaper* [ECTP19]

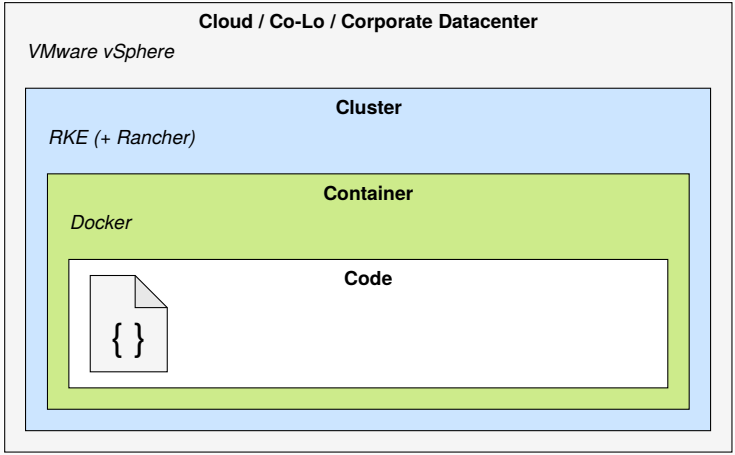


Abbildung 7: *The 4C's of Cloud Native Security* [Kub20e]

- „*Defense in Depth*"-Konzept
- „*Vanilla Kubernetes*"
- *Kubernetes Security Whitepaper* [ECTP19]

Cloud / Co-Lo / Corporate Datacenter — Vertrauenswürdige Basis eines *Kubernetes*-Clusters. Sind die hier platzierten Komponenten anfällig oder falsch konfiguriert, dann kann die Sicherheit der darauf aufbauenden Komponenten nicht gewährleistet werden.

Virtualisierungsumgebung: *VMware vSphere*

Problembereiche der *Kubernetes*-Infrastruktur:

- Netzwerkzugriff auf den API-Server (*Control Plane*)
- Netzwerkzugriff auf die Knoten (*Nodes*)
- Zugang zum Schlüsselwertspeicher (*etcd*)
- Verschlüsselung des Schlüsselwertspeichers (*etcd*)

Cloud / Co-Lo / Corporate Datacenter — Vertrauenswürdige Basis eines *Kubernetes*-Clusters. Sind die hier platzierten Komponenten anfällig oder falsch konfiguriert, dann kann die Sicherheit der darauf aufbauenden Komponenten nicht gewährleistet werden.

Virtualisierungsumgebung: *VMware vSphere*

Problembereiche der *Kubernetes*-Infrastruktur:

- Netzwerkzugriff auf den API-Server (*Control Plane*)
- Netzwerkzugriff auf die Knoten (*Nodes*)
- Zugang zum Schlüsselwertspeicher (*etcd*)
- Verschlüsselung des Schlüsselwertspeichers (*etcd*)

Cluster — Zwei Bereiche sind in dieser Schicht für die Sicherheit von *Kubernetes* von entscheidender Bedeutung.

K8s-Distribution und -Verwaltungsplattform: RKE & *Rancher*

Sicherung der Komponenten, aus denen der Cluster besteht:

- Steuerung des Zugriffs auf die *Kubernetes*-API
- Kontrolle des *kubelet*-Zugangs
- Kontrolle der Berechtigungen eines *Workloads* oder Benutzers zur Laufzeit
- Schutz der Cluster-Komponenten vor Kompromittierung

Cluster — Zwei Bereiche sind in dieser Schicht für die Sicherheit von *Kubernetes* von entscheidender Bedeutung.

K8s-Distribution und -Verwaltungsplattform: RKE & *Rancher*

Sicherung der Komponenten, aus denen der Cluster besteht:

- Steuerung des Zugriffs auf die *Kubernetes*-API
- Kontrolle des *kubelet*-Zugangs
- Kontrolle der Berechtigungen eines *Workloads* oder Benutzers zur Laufzeit
- Schutz der Cluster-Komponenten vor Kompromittierung

Cluster —

Sichern der Container-Anwendungen (*Workloads*), die letztendlich im Cluster ausgeführt werden:

- RBAC-Autorisierung (Zugriff auf die *K8s*-API)
- Authentifizierung
- *Secrets*-Verwaltung von Anwendungen (*etcd*-Verschlüsselung im Ruhezustand)
- *Pod*-Sicherheitsrichtlinien (PSPs)
- *Quality of Service* (QoS) und Cluster-Ressourcenmanagement
- Netzwerkrichtlinien
- TLS für *Kubernetes-Ingress*

Container — In einem *Kubernetes*-Cluster wird die bereitzustellende Software in einem Container, genauer gesagt innerhalb eines *Pods*, ausgeführt.

Container Runtime und *Image Registry*: *Docker* & *Harbor*

Allgemeine Empfehlungen für die Gewährleistung der Sicherheit:

- Sicherheit des zugrunde liegenden Betriebssystems
- *Scannen* von Container-Schwachstellen
- *Image*-Signierung
- Privilegierte Benutzer sperren

Container — In einem *Kubernetes*-Cluster wird die bereitzustellende Software in einem Container, genauer gesagt innerhalb eines *Pods*, ausgeführt.

Container Runtime und *Image Registry*: *Docker* & *Harbor*

Allgemeine Empfehlungen für die Gewährleistung der Sicherheit:

- Sicherheit des zugrunde liegenden Betriebssystems
- *Scannen* von Container-Schwachstellen
- *Image*-Signierung
- Privilegierte Benutzer sperren

Code — Die *Code*-Ebene der Anwendungen stellt die niedrigste Schicht im Modell dar und kann als primäre Angriffsfläche durch die darüberliegenden Abhängigkeiten gesehen werden.

Empfehlungen, die vorwiegend im Entwicklungsprozess einer Software herangezogen werden:

- Zugang nur über TLS
- Begrenzung der Port-Bereiche für die Kommunikation
- Sicherheit der „*3rd Party*"-Abhängigkeiten
- Statische Code-Analyse
- *Dynamic Probing Attacks*

Code — Die *Code*-Ebene der Anwendungen stellt die niedrigste Schicht im Modell dar und kann als primäre Angriffsfläche durch die darüberliegenden Abhängigkeiten gesehen werden.

Empfehlungen, die vorwiegend im Entwicklungsprozess einer Software herangezogen werden:

- Zugang nur über TLS
- Begrenzung der Port-Bereiche für die Kommunikation
- Sicherheit der „*3rd Party*"-Abhängigkeiten
- Statische Code-Analyse
- *Dynamic Probing Attacks*

Automatisierung I

- „IT-Automatisierung […] ist die Verwendung von Software zur Erstellung wiederholbarer Anweisungen und Prozesse, die eine menschliche Interaktion mit IT-Systemen ersetzen oder reduzieren“ [Red20a]

- IT-Optimierungen und einhergehendes ökonomisches Handeln
- Digitale Transformation in Richtung des *Software-Defined Datacenters* (SDDC)
- Operative Erleichterungen wie Automatisierung und Orchestrierung

Mögliche Anwendungsfälle …

- Bereitstellung von Ressourcen (Basis-Infrastruktur)
- Konfigurationsmanagement
- Automatisiertes Erzeugen von Anwendungsumgebungen (VMs, Container)
- Bereitstellung von Anwendungen (CI / CD)
- *Security* und *Compliance*
- *Governance*
- *Self-Service*-Portale

Mögliche Anwendungsfälle …

- Bereitstellung von Ressourcen (Basis-Infrastruktur)
- Konfigurationsmanagement
- Automatisiertes Erzeugen von Anwendungsumgebungen (VMs, Container)
- Bereitstellung von Anwendungen (CI / CD)
- *Security* und *Compliance*
- *Governance*
- *Self-Service*-Portale

Realisierung der Infrastruktur

Überblick des Szenarios

- ***Image Registry*** – *Harbor*
- ***Cluster Management*** – *Rancher*, RKE
- ***Self-Service Portal*** – *VMware vRealize Suite* (*vRA*, *vRO*)
- ***CI / CD Pipeline*** – *GitLab*, *Runner*

Abbildung 8: Virtualisierungsumgebung des Szenarios

Überblick des Szenarios

- ***Image Registry*** – *Harbor*
- ***Cluster Management*** – *Rancher*, RKE
- ***Self-Service Portal*** – *VMware vRealize Suite* (*vRA*, *vRO*)
- ***CI / CD Pipeline*** – *GitLab*, *Runner*

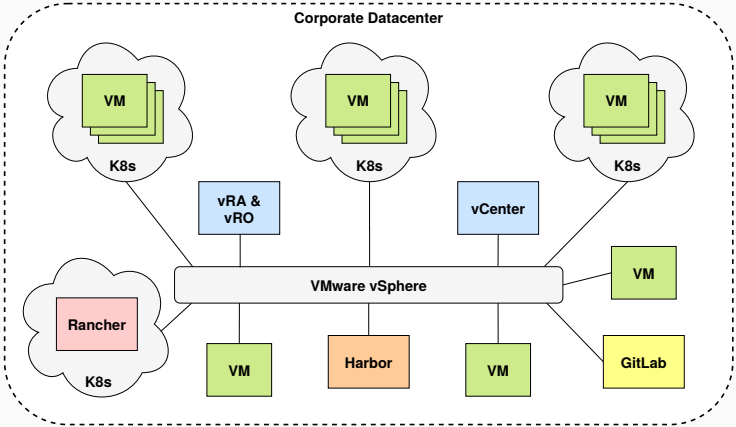


Abbildung 8: Virtualisierungsumgebung des Szenarios

Abhängigkeiten innerhalb des Szenarios

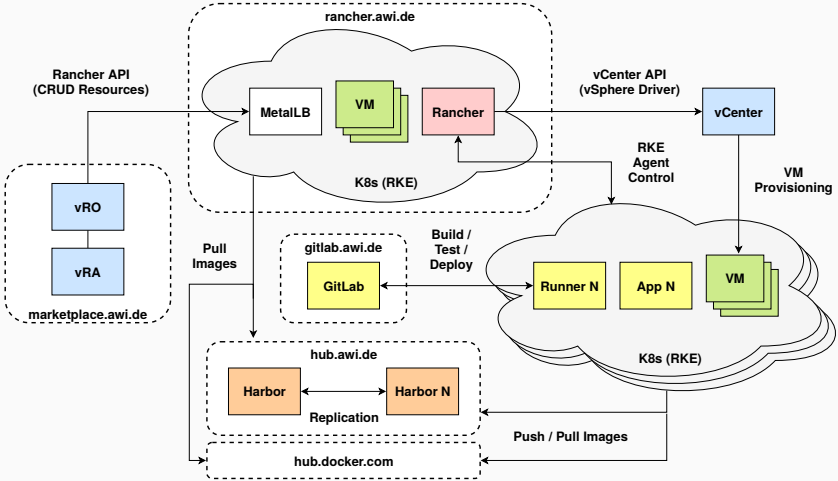


Abbildung 9: Abhängigkeiten der Interaktionsplattfromen innerhalb des Szenarios

Evaluation

Sicherheit

- Generische Überprüfung der Sicherheitsanforderungen eines *Kubernetes*-Clusters durch die Anwendung *kube-bench* [Aqu20]
- Prüfung basiert auf der *Kubernetes Benchmark* des CIS (*Center for Internet Security*) [CIS19]

- Ergebnis eines Tool-Durchlaufs ist nahezu vollständig positiv (*PASS*)
- Einsatz gehärteter RKE-*Templates* mit RBAC-Richtlinien, PSPs und *Network Policies*
- Absicherung der verteilten Knoten mit installiertem Betriebssystem und der *Container Runtime* (Host-Sicherheit)

Sicherheit

- Generische Überprüfung der Sicherheitsanforderungen eines *Kubernetes*-Clusters durch die Anwendung *kube-bench* [Aqu20]
- Prüfung basiert auf der *Kubernetes Benchmark* des CIS (*Center for Internet Security*) [CIS19]

- Ergebnis eines Tool-Durchlaufs ist nahezu vollständig positiv (*PASS*)
- Einsatz gehärteter RKE-*Templates* mit RBAC-Richtlinien, PSPs und *Network Policies*
- Absicherung der verteilten Knoten mit installiertem Betriebssystem und der *Container Runtime* (Host-Sicherheit)

Tabelle 1: Kontrollübersicht der CIS *Kubernetes Benchmark* (*v1.5.0*) bezogen auf einen *Rancher*-verwalteten RKE-Cluster [CIS19, Ran20a]

ID	*Controls* ($\sum$ 122)	*Scored* ($\sum$ 92)	*Not Scored* ($\sum$ 30)	*Not Applicable* ($\sum$ 23)
1	*Control Plane Components*	58	7	14
2	*etcd*	6	1	0
3	*Control Plane Configuration*	1	2	0
4	*Worker Nodes*	20	3	9
5	*Policies*	7	17	0

Tabelle 1: Kontrollübersicht der CIS *Kubernetes Benchmark* (*v1.5.0*) bezogen auf einen *Rancher*-verwalteten RKE-Cluster [CIS19, Ran20a]

ID	*Controls* ($\sum$ 122)	*Scored* ($\sum$ 92)	*Not Scored* ($\sum$ 30)	*Not Applicable* ($\sum$ 23)
1	*Control Plane Components*	58	7	14
2	*etcd*	6	1	0
3	*Control Plane Configuration*	1	2	0
4	*Worker Nodes*	20	3	9
5	*Policies*	7	17	0

Verwendbarkeit

- Bewertung der umgesetzten Lösung bezogen auf den praktischen Einsatz
- Durchführung der zugrunde liegenden Anwendungsfälle (Anforderungen s. Folie 8f.)
- Systematische Bewertung der Infrastruktur durch eine interaktive *Google Forms*-Umfrage [Goo20a]

- Umfrage besteht aus kleinen Arbeitsaufträgen und anschließenden Fragestellungen bezogen auf die vier Interaktionsplattformen
- Aufgrund der Stichprobengröße (Quantität) von sechs Teilnehmern nicht repräsentativ, dennoch lassen sich qualitative Aussagen schlussfolgern …

Verwendbarkeit

- Bewertung der umgesetzten Lösung bezogen auf den praktischen Einsatz
- Durchführung der zugrunde liegenden Anwendungsfälle (Anforderungen s. Folie 8f.)
- Systematische Bewertung der Infrastruktur durch eine interaktive *Google Forms*-Umfrage [Goo20a]

- Umfrage besteht aus kleinen Arbeitsaufträgen und anschließenden Fragestellungen bezogen auf die vier Interaktionsplattformen
- Aufgrund der Stichprobengröße (Quantität) von sechs Teilnehmern nicht repräsentativ, dennoch lassen sich qualitative Aussagen schlussfolgern …

Verwendbarkeit

- Höherer Zeitbedarf für die Auseinandersetzung mit allen Interaktionsplattformen
- Bedarf weiterer *Workshops* bzgl. der *K8s*-Domäne
- Wiederholung der Umfrage bei einem fortgeschritteneren Kenntnisstand

- Genereller Zuspruch für getätigte Sicherheitskonfigurationen trotz vorherrschender Einschränkungen
- Ausschließliche Verwendung einer *Private Registry* wäre akzeptabel
- CVE-*Scanner* und *Docker Content Trust* wurden als unterstützend und wichtig aufgenommen
- Einsatz von *Shared Runners* im Kontext der *CI / CD Pipelines* sind anzustreben
- Redundante Handlungsmöglichkeiten im *Self-Service Portal* und im *Cluster Management* vereinigen

Verwendbarkeit

- Höherer Zeitbedarf für die Auseinandersetzung mit allen Interaktionsplattformen
- Bedarf weiterer *Workshops* bzgl. der *K8s*-Domäne
- Wiederholung der Umfrage bei einem fortgeschritteneren Kenntnisstand

- Genereller Zuspruch für getätigte Sicherheitskonfigurationen trotz vorherrschender Einschränkungen
- Ausschließliche Verwendung einer *Private Registry* wäre akzeptabel
- CVE-*Scanner* und *Docker Content Trust* wurden als unterstützend und wichtig aufgenommen
- Einsatz von *Shared Runners* im Kontext der *CI / CD Pipelines* sind anzustreben
- Redundante Handlungsmöglichkeiten im *Self-Service Portal* und im *Cluster Management* vereinigen

Fazit und Ausblick

Fazit

- Erfolgreicher Aufbau von *On-Premises*-Cluster auf der Grundlage der Container-basierten Virtualisierung (*Kubernetes*, *Docker*)
- Fokus auf die zugrunde liegende Sicherheit und die automatisierte Bereitstellung über mehrere involvierte Interaktionsplattformen
- Realisierung der Infrastruktur (*Harbor*, *Rancher* mit RKE, *VMware vRealize Suite* (*vRA*, *vRO*), *GitLab*) im Rechenzentrum des AWIs
- *Kubernetes* als relativ stabile Container-Plattform im dynamischen und komplexen Umfeld der Container-Technologien
- Sicherheitsbetrachtung durch die anerkannte CIS *Kubernetes Benchmark* und Bewertung der Infrastruktur durch eine interaktive Umfrage in der Nutzerschaft

Fazit

- Erfolgreicher Aufbau von *On-Premises*-Cluster auf der Grundlage der Container-basierten Virtualisierung (*Kubernetes*, *Docker*)
- Fokus auf die zugrunde liegende Sicherheit und die automatisierte Bereitstellung über mehrere involvierte Interaktionsplattformen
- Realisierung der Infrastruktur (*Harbor*, *Rancher* mit RKE, *VMware vRealize Suite* (*vRA*, *vRO*), *GitLab*) im Rechenzentrum des AWIs
- *Kubernetes* als relativ stabile Container-Plattform im dynamischen und komplexen Umfeld der Container-Technologien
- Sicherheitsbetrachtung durch die anerkannte CIS *Kubernetes Benchmark* und Bewertung der Infrastruktur durch eine interaktive Umfrage in der Nutzerschaft

Ausblick I

Weitere Maßnahmen für eine produktive Nutzung der *K8s*-Infrastruktur:

- Fortlaufende Anpassungen im Bereich der Sicherheitsmaßnahmen nach dem Stand der Technik
- Standardmäßige Ressourcenbegrenzungen (*Namespaces*, *Pods*)
- *Chaos Engineering* bzw. *Testing* heranziehen (*chaoskube*, *kube-monkey*) [AD19, vgl. S. 118ff]
- Erprobung der *K8s*-Funktion „*Rolling Update*“ oder die Verwendung von *Git*-SHA-*Tags* in den *CI / CD Pipelines* [AD19, vgl. S. 249]
- Speicheranbindung (*Storage*) für Container-Anwendungen in den Clustern
- *Shared Runners* in der *GitLab*-Plattform
- *Load Balancer*-Problematik in den RKE-*Downstream*-Clustern [Kub20c]
- Erneute Durchführung der interaktiven *Google Forms*-Umfrage

Ausblick I

Weitere Maßnahmen für eine produktive Nutzung der *K8s*-Infrastruktur:

- Fortlaufende Anpassungen im Bereich der Sicherheitsmaßnahmen nach dem Stand der Technik
- Standardmäßige Ressourcenbegrenzungen (*Namespaces*, *Pods*)
- *Chaos Engineering* bzw. *Testing* heranziehen (*chaoskube*, *kube-monkey*) [AD19, vgl. S. 118ff]
- Erprobung der *K8s*-Funktion „*Rolling Update*“ oder die Verwendung von *Git*-SHA-*Tags* in den *CI / CD Pipelines* [AD19, vgl. S. 249]
- Speicheranbindung (*Storage*) für Container-Anwendungen in den Clustern
- *Shared Runners* in der *GitLab*-Plattform
- *Load Balancer*-Problematik in den RKE-*Downstream*-Clustern [Kub20c]
- Erneute Durchführung der interaktiven *Google Forms*-Umfrage

Ausblick II

Weiterführende interessante Themen in diesem Umfeld:

- Aufbau föderierter oder geografisch verteilter *K8s*-Cluster im HIFIS-Kontext (*KubeFed* [Kub20a])
- Mikrosegmentierung mit Hilfe von NSX [WAG+18, vgl. S. 477ff] in einem von *VMware* geprägten SDDC bzw. einer *Hybrid Cloud*
- *Frakti* [Luk18, vgl. S. 602] als *Container Runtime*, um Container-*Images* direkt über einen *Hypervisor* ausführen zu lassen
- Unprivilegierte Verwendung des *Docker Daemons* durch *kaniko* [Goo20b] für *Docker Build*-Vorgänge
- *Singularity* [God19] als *Container Runtime* im HPC-Wissenschaftsbereich
- Übertragbarkeit der Umgebung durch vollständige *Open Source*-Lösungen (KVM [Red20b] als Virtualisierungstechnologie)
- Fortlaufende *Kubernetes*-Integration innerhalb von *VMware vSphere* ab der Version *7.0* [VMw20]

Ausblick II

Weiterführende interessante Themen in diesem Umfeld:

- Aufbau föderierter oder geografisch verteilter *K8s*-Cluster im HIFIS-Kontext (*KubeFed* [Kub20a])
- Mikrosegmentierung mit Hilfe von NSX [WAG+18, vgl. S. 477ff] in einem von *VMware* geprägten SDDC bzw. einer *Hybrid Cloud*
- *Frakti* [Luk18, vgl. S. 602] als *Container Runtime*, um Container-*Images* direkt über einen *Hypervisor* ausführen zu lassen
- Unprivilegierte Verwendung des *Docker Daemons* durch *kaniko* [Goo20b] für *Docker Build*-Vorgänge
- *Singularity* [God19] als *Container Runtime* im HPC-Wissenschaftsbereich
- Übertragbarkeit der Umgebung durch vollständige *Open Source*-Lösungen (KVM [Red20b] als Virtualisierungstechnologie)
- Fortlaufende *Kubernetes*-Integration innerhalb von *VMware vSphere* ab der Version *7.0* [VMw20]

Vielen Dank für Ihre Aufmerksamkeit!

Analyse der Sicherheit und der automatisierten Bereitstellung eines *On-Premises*-Clusters auf der Grundlage der Container-basierten Virtualisierung: *Kubernetes* im Wissenschaftsbetrieb

Masterarbeit im Studiengang Informatik (KSS)

Fabian Mangels <fabian@mangels.it>

Bremen, 05.10.2020

Anhang

Alfred-Wegener-Institut (AWI)

- Deutsche Forschungseinrichtung in der Polar- und Meeresforschung
- Hauptstandort in Bremerhaven
- $>$ 1000 Mitarbeiter
- Außenstellen: Helgoland, Oldenburg, Potsdam und Sylt
- Fachbereiche: Geo-, Bio- und Klimawissenschaften
- Leistungsfähige Infrastruktur: Stationen in der Arktis und Antarktis, Schiffe und Flugzeuge
- Rechenzentrum

Abbildung 10: AWI-Hauptgebäude in Bremerhaven [AWI19]

Vorherrschende Themen:

- „eine nahtlose, Zentren-übergreifende IT-Infrastruktur mit integrierten ICT-Dienstleistungen auf der Basis schneller Netze und einheitlichem Nutzerzugang“ [Hel18, S. 9]
- „einen in die Zusammenarbeits- und Forschungsprozesse integrierten sicheren, effizienten und weltweit verfügbaren Daten- und Anwendungszugriff auf der Basis von Cloud Diensten“ [Hel18, S. 9]
- „Ausbildung und Unterstützung, um qualitativ hochwertige und nachhaltige Software zu entwickeln und zu veröffentlichen“ [Hel18, S. 9]

Tabelle 2: Übersicht der *XaaS*-Modelle des *Cloud Computings* [Lie19, vgl. S. 56]

	On Premise	***Infrastructure*** (*as a Service*)	***Containers*** (*as a Service*)	***Platform*** (*as a Service*)	***Software*** (*as a Service*)
Applications	✓	✓	✓	✓	✗
Data	✓	✓	✓	✓	✗
Runtime	✓	✓	✓	✗	✗
Middleware	✓	✓	✓	✗	✗
OS	✓	✓	✗	✗	✗
Virtualization	✓	✗	✗	✗	✗
Servers	✓	✗	✗	✗	✗
Storage	✓	✗	✗	✗	✗
Networking	✓	✗	✗	✗	✗
✓ = *Self-Managed*, ✗ = *Provider-Supplied*					

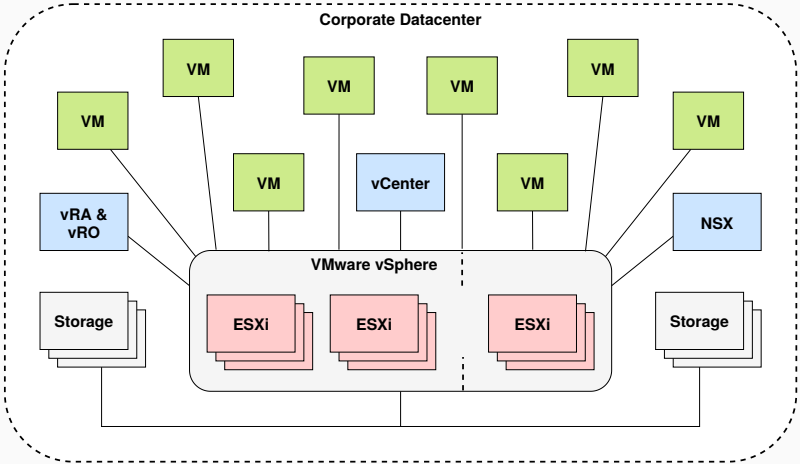


Abbildung 11: Aufbau einer *VMware vSphere*-Virtualisierungsumgebung [WAG+18, vgl. S. 53ff, 167ff]

Software-Defined Datacenter (SDDC)

Software-Defined Datacenter (SDDC) [WAG+18, vgl. S. 53ff, 1235]:

- ***Compute Virtualization*** –
  VMware vSphere, *ESXi-Hypervisor*
- ***Software-Defined Networking*** (SDN), *Network Virtualization* –
  VMware NSX (*Network and Security Virtualization*)
- ***Software-Defined Storage*** (SDS), *Storage Virtualization* –
  VMware vSAN, *NetApp*
- ***Management*** –
  vCenter Server, *Platform Services Controller* (PSC), *Update Manager*, *VMware vSphere Client*
- ***Automation*** –
  VMware vRealize Suite mit *vRealize Automation* (*vRA*) und *vRealize Orchestrator* (*vRO*)

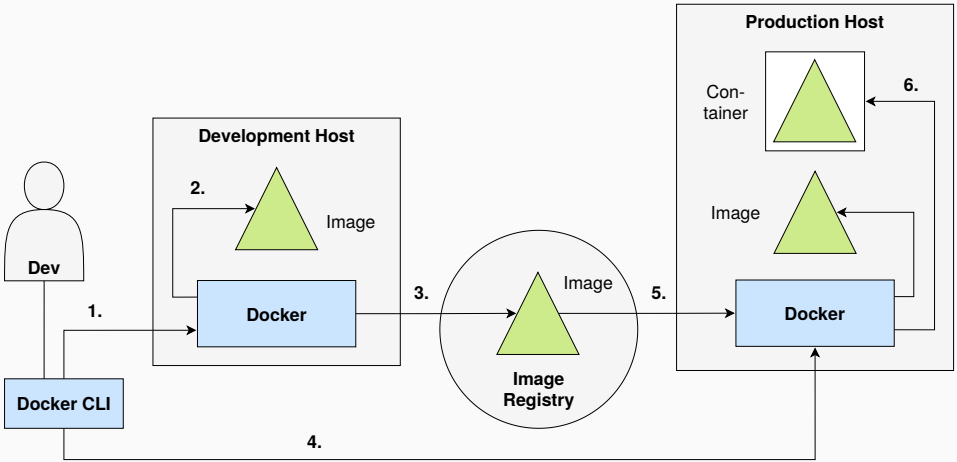


Abbildung 12: Bereitstellen von *Docker*-Containern [Luk18, vgl. S. 16], [Doc20a]

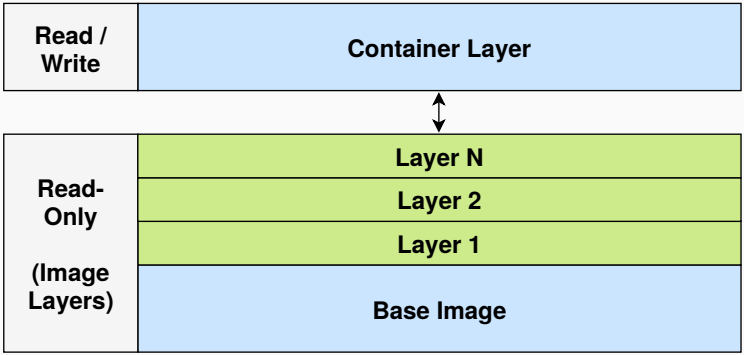


Abbildung 13: Schematischer Aufbau eines Container-*Images* [Lie19, vgl. S. 114]

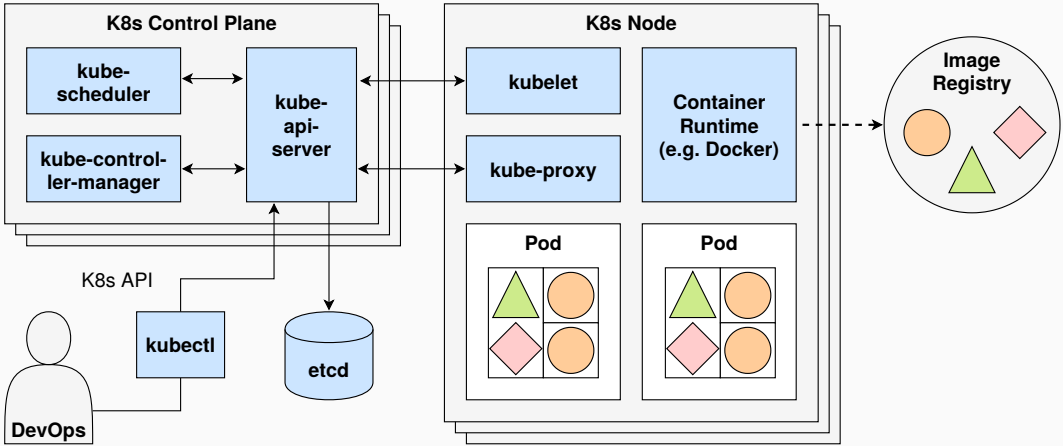


Abbildung 14: *Kubernetes*-Architektur mit *Control Plane* und *Nodes* [Kub20b, Kub20d]

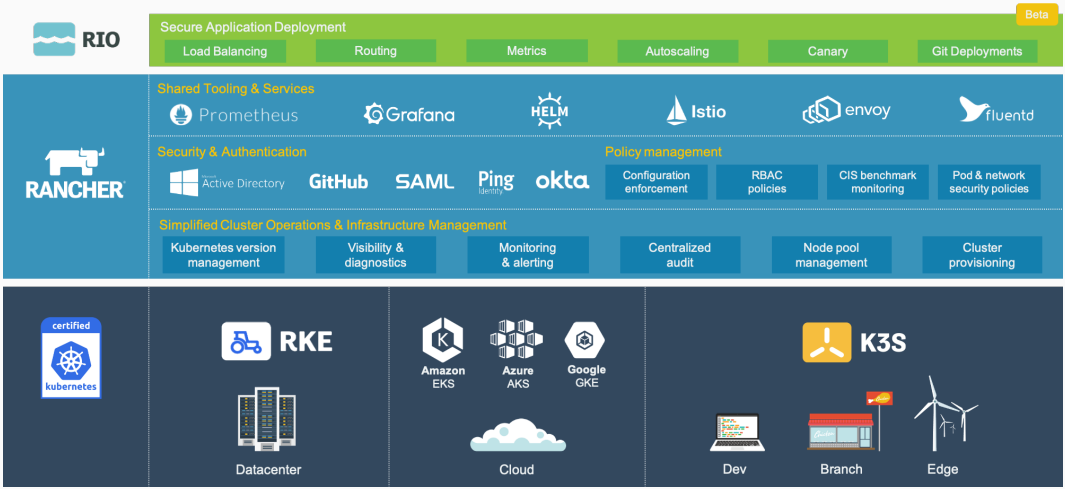


Abbildung 15: Bestandteile einer *Rancher*-Umgebung [Ran20c, S. 4]

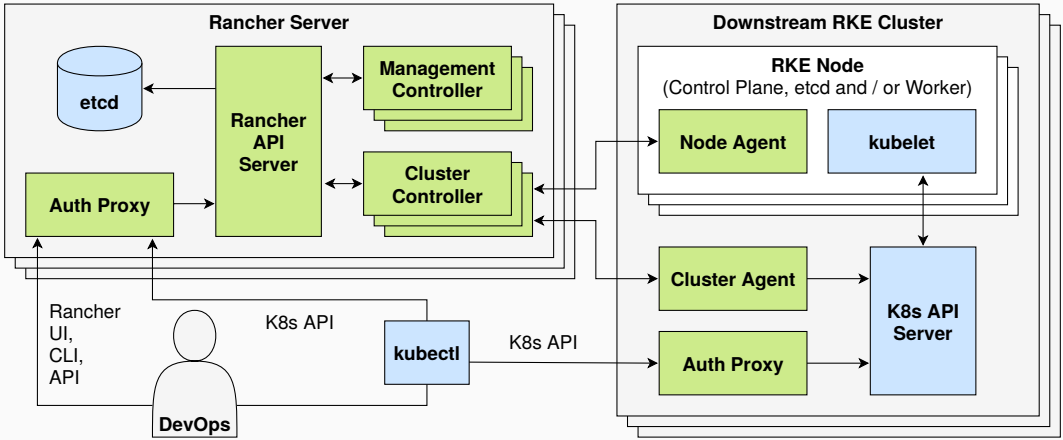


Abbildung 16: *Rancher*-Architektur mit *Downstream*-RKE-Cluster [Ran20c, vgl. S. 8], [Ran20b]

- Schutz von Informationen im gesamten Lebenszyklus eines Containers und darüber hinaus
- Anforderungen nach aufsteigendem Schutzbedarf: Basis-, Standard- und erhöhte Anforderungen
- Sicherheitsleitfaden für Container-Anwendungen [SMS17] des *National Institute of Standards and Technology* (NIST)

Besondere Gefährdungslage nach dem BSI bei …

- Schwachstellen in *Images*
- Administrative Zugänge ohne Absicherung
- Tool-basierte Orchestrierung ohne Absicherung
- Ausbruch aus dem Container
- Datenverluste durch fehlende Persistenz
- Vertraulichkeitsverlust von Zugangsdaten

CPU-Zugriff in VMs und Containern

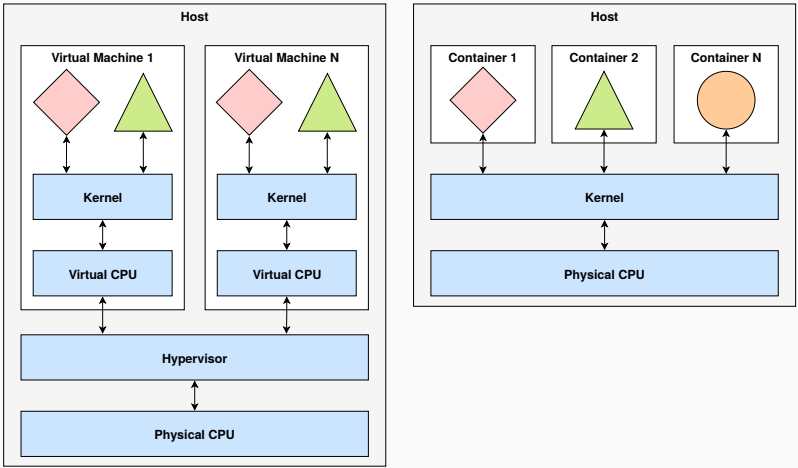


Abbildung 17: Unterschiedlicher Zugriff auf die CPU in VMs und Containern [Luk18, vgl. S. 12]

Container-Virtualisierung auf einem Host

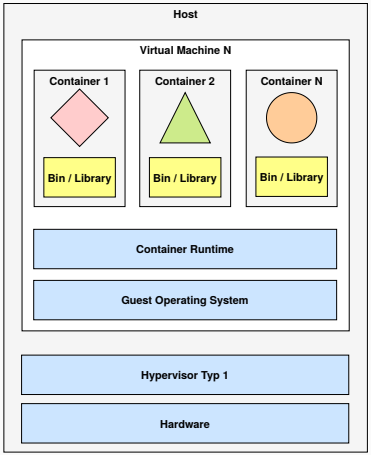


Abbildung 18: Container-Virtualisierung in virtuellen Maschinen [Lie19, vgl. S. 91, 509]

Bestandteile des *Linux*-Kernels:

- *Namespaces*, *Capabilities*, *cgroups*
- *Secure Computing Mode* (*seccomp*)
- *Linux Security Modules* (LSMs): *AppArmor*, *SELinux*

Evaluation – Verwendbarkeit I

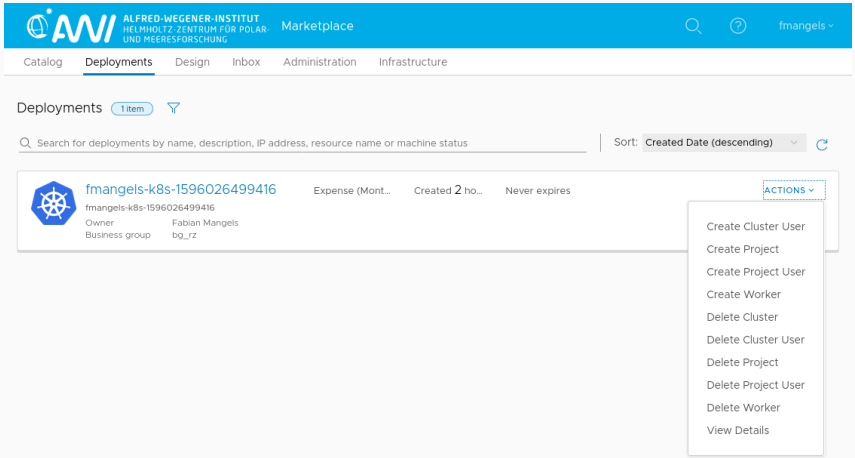


Abbildung 19: Anwenden von weiteren Operationen auf ein *K8s*-Cluster-*Deployment* im *Self-Service Portal*

Evaluation – Verwendbarkeit II

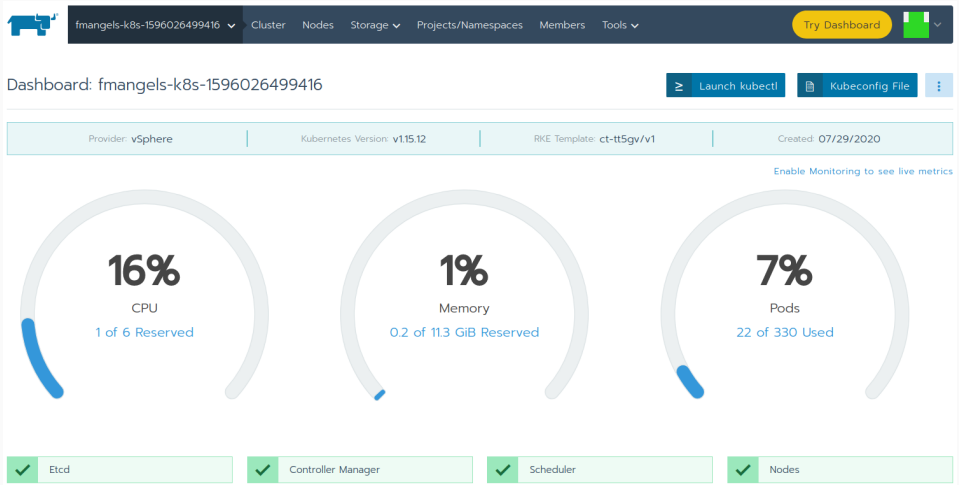


Abbildung 20: *Dashboard* eines im *Cluster Management Rancher* verwalteten *K8s*-Clusters

Evaluation – Verwendbarkeit III

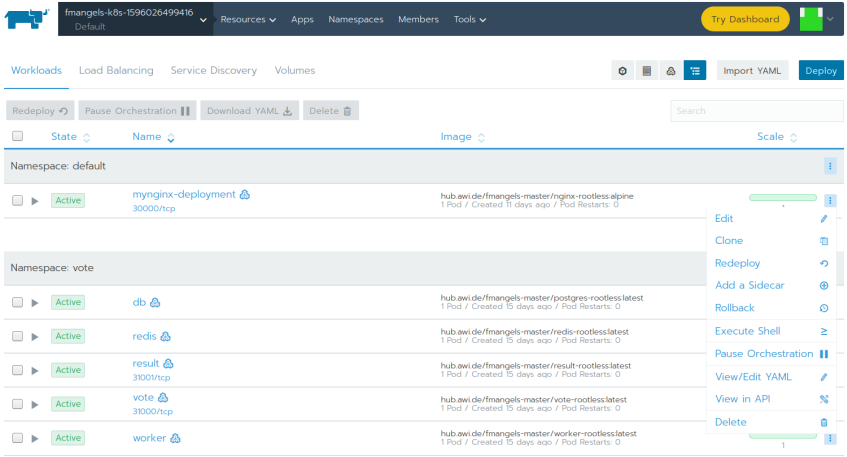


Abbildung 21: Übersicht bereitgestellter *Workloads* im *Cluster Management Rancher*

Evaluation – Verwendbarkeit IV

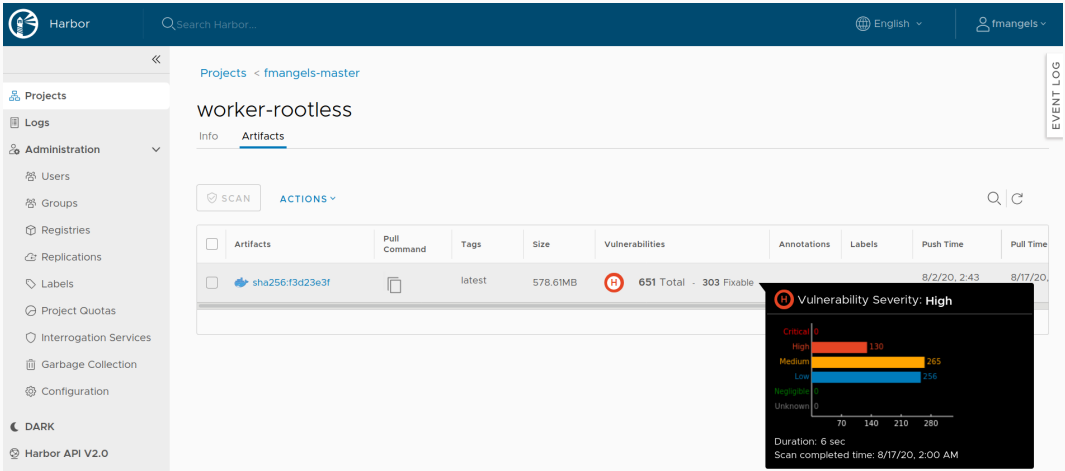


Abbildung 22: Schwachstellenanalyse eines *Docker Images* in der *Registry Harbor*

Komponenten des Minimalbeispiel „*Voting App*“

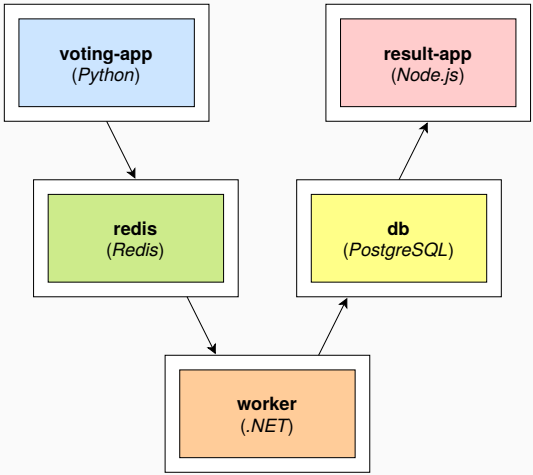


Abbildung 23: Komponenten des Minimalbeispiels „*Voting App*“ [Doc20b]

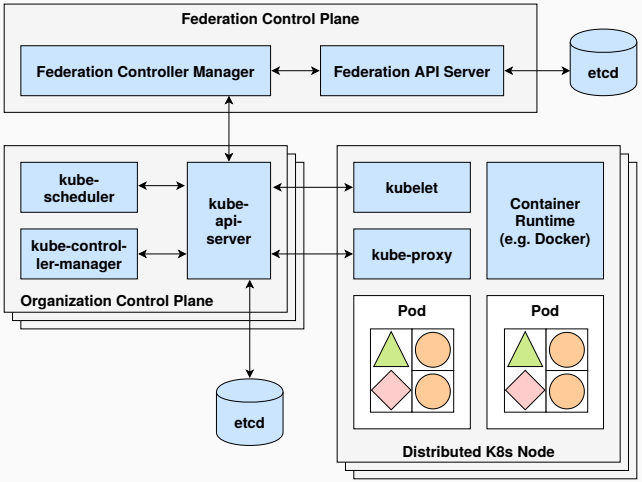


Abbildung 24: *K8s*-Cluster in einer Föderation [Luk18, vgl. S. 604], [Lie19, vgl. S. 1164]

Literaturverzeichnis

Literaturverzeichnis i

Arundel, John ; Domingus, Justin:
Cloud Native DevOps mit Kubernetes.
dpunkt.verlag, 2019 https://learning.oreilly.com/library/view/cloud-native-devops/9781098123178/. –

ISBN 9783960888284

Aqua Security:
kube-bench – Checks whether Kubernetes is deployed according to security best practices as defined in the CIS Kubernetes Benchmark.
https://github.com/aquasecurity/kube-bench.
Version: Juni 2020. –
abgerufen am 27.06.2020

Literaturverzeichnis ii

AWI:
AWI Bremerhaven.
https://www.awi.de/ueber-uns/standorte/bremerhaven.html.
Version: April 2019. –
abgerufen am 24.04.2020

BSI:
IT-Grundschutz-Kompendium.
Reguvis, 2020 https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/IT_Grundschutz_Kompendium_Edition2020.pdf?__blob=publicationFile&v=6. –
ISBN 9783846209066. –
abgerufen am 30.05.2020

BSI:
SYS.1.6: Container.
https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Drafts/Community_Draft/SYS_1_6_Container_CD.pdf?__blob=publicationFile&v=11.
Version: März 2020. –
abgerufen am 30.05.2020

CIS:
CIS Kubernetes Benchmark – v1.5.0 - 10-14-2019.
https://www.cisecurity.org/benchmark/kubernetes/.
Version: Oktober 2019. –
abgerufen am 08.06.2020


Docker:
Docker overview.
https://docs.docker.com/engine/docker-overview/.
Version: Mai 2020. –
abgerufen am 02.05.2020


Docker Samples:
Example Voting App.
https://github.com/dockersamples/example-voting-app.
Version: Februar 2020. –
abgerufen am 26.07.2020

Edwards, Stefan ; Czarnota, Dominik ; Tonic, Robert ; Perez, Ben:
Kubernetes : Security Whitepaper.
https://github.com/kubernetes/community/blob/master/wg-security-audit/findings/Kubernetes%20White%20Paper.pdf.
Version: Juni 2019. –
abgerufen am 22.03.2020

Godlove, David:
Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning): Singularity : Simple, secure containers for compute-driven workloads.
https://dl.acm.org/doi/10.1145/3332186.3332192.
Version: August 2019. –
abgerufen am 29.05.2020


Google:
Google Forms: Free Online Surveys for Personal Use.
https://www.google.com/forms/about/.
Version: August 2020. –
abgerufen am 07.08.2020


GoogleContainerTools:
kaniko – Build Container Images In Kubernetes.
https://github.com/GoogleContainerTools/kaniko.
Version: August 2020. –
abgerufen am 04.08.2020

Helmholtz:
Helmholtz Infrastructure for Federated ICT Services (HIFIS).
https://www.helmholtz.de/fileadmin/user_upload/01_forschung/Helmholtz_Inkubator_HIFIS.pdf.
Version: September 2018. –
abgerufen am 11.03.2020

Kubernetes:
KubeFed – Kubernetes Cluster Federation.
https://github.com/kubernetes-sigs/kubefed.
Version: Mai 2020. –
abgerufen am 13.06.2020

Kubernetes:
Kubernetes Components.
https://kubernetes.io/docs/concepts/overview/components/.
Version: März 2020. –
abgerufen am 13.05.2020

Kubernetes:
NGINX Ingress Controller – Bare-metal considerations.
https://kubernetes.github.io/ingress-nginx/deploy/baremetal/.
Version: Juli 2020. –
abgerufen am 12.07.2020


Kubernetes:
Options for Highly Available topology.
https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/ha-topology/.
Version: Mai 2020. –
abgerufen am 16.06.2020


Kubernetes:
Overview of Cloud Native Security.
https://kubernetes.io/docs/concepts/security/overview/.
Version: Juni 2020. –
abgerufen am 16.06.2020

Kubernetes:
What is Kubernetes?
https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/.
Version: März 2020. –
abgerufen am 28.04.2020

Liebel, Oliver:
Skalierbare Container-Infrastrukturen : das Handbuch für Administratoren.
2., aktualisierte und erweiterte Auflage.
Bonn : Rheinwerk Verlag, 2019 (Rheinwerk Computing). –
ISBN 3836263858 and 9783836263856 and 9783836263856


Lukša, Marko:
Kubernetes in Action : Anwendungen in Kubernetes-Clustern bereitstellen und verwalten.
München : Hanser, 2018
https://www.hanser-elibrary.com/doi/book/10.3139/9783446456020. – ISBN 9783446456020


Mangels, Fabian:
***Analyse der Sicherheit und der automatisierten Bereitstellung eines On-Premises-Clusters auf der Grundlage der Container-basierten Virtualisierung: Kubernetes im Wissenschaftsbetrieb*, Hochschule Bremen, Masterarbeit, September 2020.**
https://epic.awi.de/id/eprint/52946/


Rancher Labs:
CIS Benchmark Rancher Self-Assessment Guide - v2.4.
https://releases.rancher.com/documents/security/2.4/Rancher_Benchmark_Assessment.pdf.
Version: Juni 2020. –
abgerufen am 07.06.2020


Rancher Labs:
Rancher – Architecture.
https://rancher.com/docs/rancher/v2.x/en/overview/architecture/.
Version: Mai 2020. –
abgerufen am 17.05.2020

Rancher Labs:
Rancher 2.4: Technical Architecture.
https://cdn2.hubspot.net/hubfs/468859/eBooks,%20reports,%20and%20whitepapers/20200305_Rancher_2_4_Architecture_WP.pdf.
Version: Februar 2020. –
abgerufen am 25.03.2020

Red Hat:
Automatisierung – IT-Automatisierung erklärt.
https://www.redhat.com/de/topics/automation/whats-it-automation.
Version: Juni 2020. –
abgerufen am 24.06.2020


Red Hat:
Virtualisierung – KVM erklärt.
https://www.redhat.com/de/topics/virtualization/what-is-KVM.
Version: September 2020. –
abgerufen am 07.09.2020


Souppaya, Murugiah ; Morello, John ; Scarfone, Karen:
NIST : Application Container Security Guide.
http://dx.doi.org/https://doi.org/10.6028/NIST.SP.800-190.
Version: September 2017. –
abgerufen am 30.05.2020

VMware:
vSphere with Kubernetes Configuration and Management.
https://docs.vmware.com/en/VMware-vSphere/7.0/vsphere-esxi-vcenter-server-70-vsphere-with-kubernetes-guide.pdf.
Version: Juli 2020. –
abgerufen am 25.08.2020

Wöhrmann, Bertram ; Alder, Urs S. ; Grosse, Jan ; Baumgart, Günter ; Schönfeld, Thomas ; Zimmer, Dennis ; Wegner, Frank ; Söldner, Jens-Henrik:
VMware vSphere 6.7 : das umfassende Handbuch.
5., aktualisierte und erweiterte Auflage.
Bonn : Rheinwerk Verlag, 2018 (Rheinwerk Design). –
ISBN 383626336X and 9783836263368 and 9783836263368