Landesamt für Digitalisierung,
Breitband und Vermessung

Container-Technologie im E-Government

Zwischen Automatisierung und IT-Sicherheit

06.10.2021

Agenda

- Vorstellung
  - Matthias Kanzog
  - IT-Dienstleistungszentrum des Freistaats Bayern
- Containerbetrieb im IT-DLZ
- Aufbau einer Kubernetes Infrastruktur
- Entwicklung von Container Anwendungen
- Deployment
- IT-Sicherheit
- Lessons Learned

Vorstellung

Matthias Kanzog

- 33 Jahre aus Rosenheim
- Ausbildung zum Bankkaufmann
- Studium Wirtschaftsingenieurwesen
- Masterabschluss und Weiterführung des Projektes
- Seit 2 Jahren im IT-DLZ
  - Aufbau eines Container-Services
  - DevOps orientierte Arbeitsweise

Vorstellung IT-Dienstleistungszentrum des Freistaats Bayern

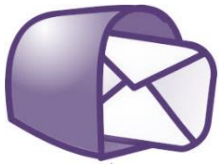
- **zentraler IT-Dienstleister staatlicher Stellen in Bayern**
- ca. 500 Mitarbeiterinnen und Mitarbeiter an 7 Standorten
- Technischer Betrieb geschäftsbereichsübergreifender Infrastruktur für das Bayerische Behördennetz
- Technischer Betrieb von Servern und Datenbanken
- Betrieb hochverfügbarer zentraler Plattformen
- Entwicklung und Anwendungsbetrieb
- Teile des IT-DLZ sind seit 2016 nach ISO 27001 auf Basis von IT-Grundschutz zertifiziert

IT-DLZ in Zahlen

Server

ca. 8.100

E-Mail-Postfächer



ca. 105.000

Datenbanken

ca. 3.500

Storage

ca. 15 Petabyte

genutzter Speicher

Virtualisierungsquote

73%

Tickets am Service-Desk

24.000

pro Jahr

„Gretchenfrage“

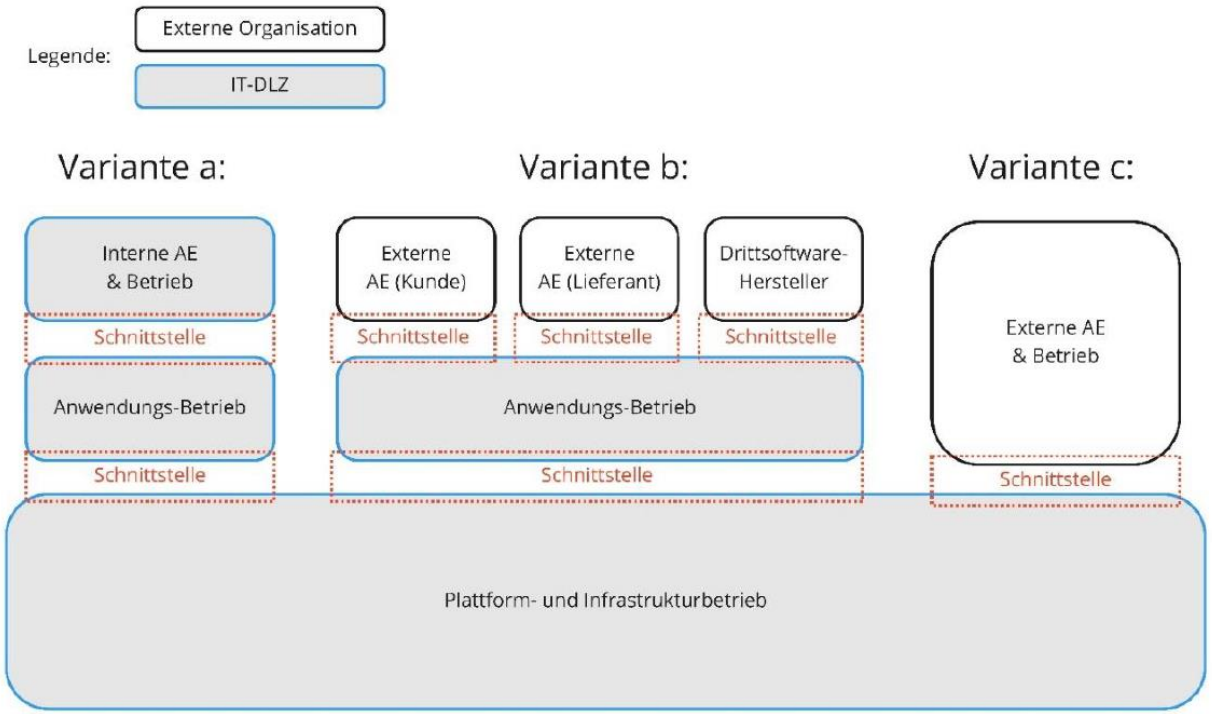
Staatliche Behörde

Flexibilität, Agilität

Containerbetrieb im IT-DLZ

Authentication & Authorization	Microsoft Active Directory		
Repository	Bitbucket	bitnami	Container Secrets
CI/CD - Pipeline	HELM, TEKTON, argo, Bamboo	FLEET, ANSIBLE	Automation
Logging	ELK Stack, splunk>, graylog	Grafana, Prometheus	Monitoring
Registry	HARBOR	clair, trivy	Image Scanning
Container Engine	containerd		
Networking & Service Mesh	canal	NFS Network File System	Persistent Storage
Platform & Orchestration	RANCHER, kubernetes	RKE 2	
Operating Systems	SUSE		
Infrastructure Drivers	vmware vSphere		

Containerbetrieb im IT-DLZ



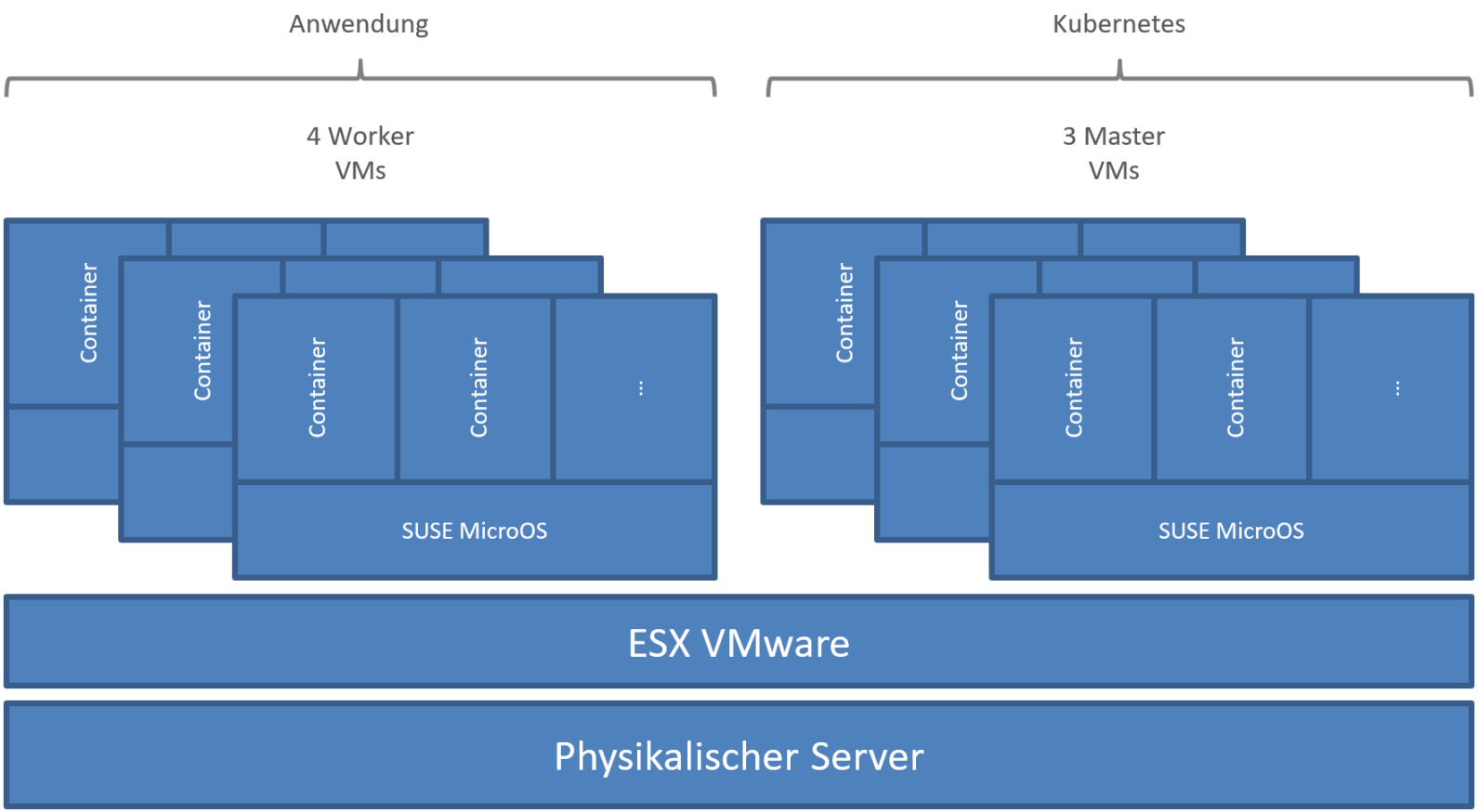
Aufbau K8s Infrastruktur

- Aufbau einer Container-Plattform inkl. Image-Registry
  - Bereitstellung mehrerer, zum Teil bereits produktiv genutzter, Kubernetes-Cluster auf Basis von Rancher (RKE2)
  - Verwaltung über Rancher Management Server

- Aufbau einer Automatisierung zur schnelleren Bereitstellung der Rancher-Cluster mittels „Ansible“

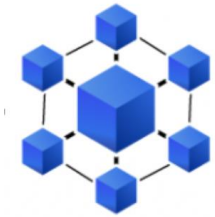
- Image-Registry Harbor
  (mit integriertem CVE-Schwachstellenscanner)

Aufbau K8s Infrastruktur



Entwicklung von Container Anwendungen

- Entwicklung der Anwendungen
  - Eigenentwicklung (intern)
  - Externe Entwicklung durch
    - Kunden
    - Dienstleister
    - Drittsoftwarehersteller



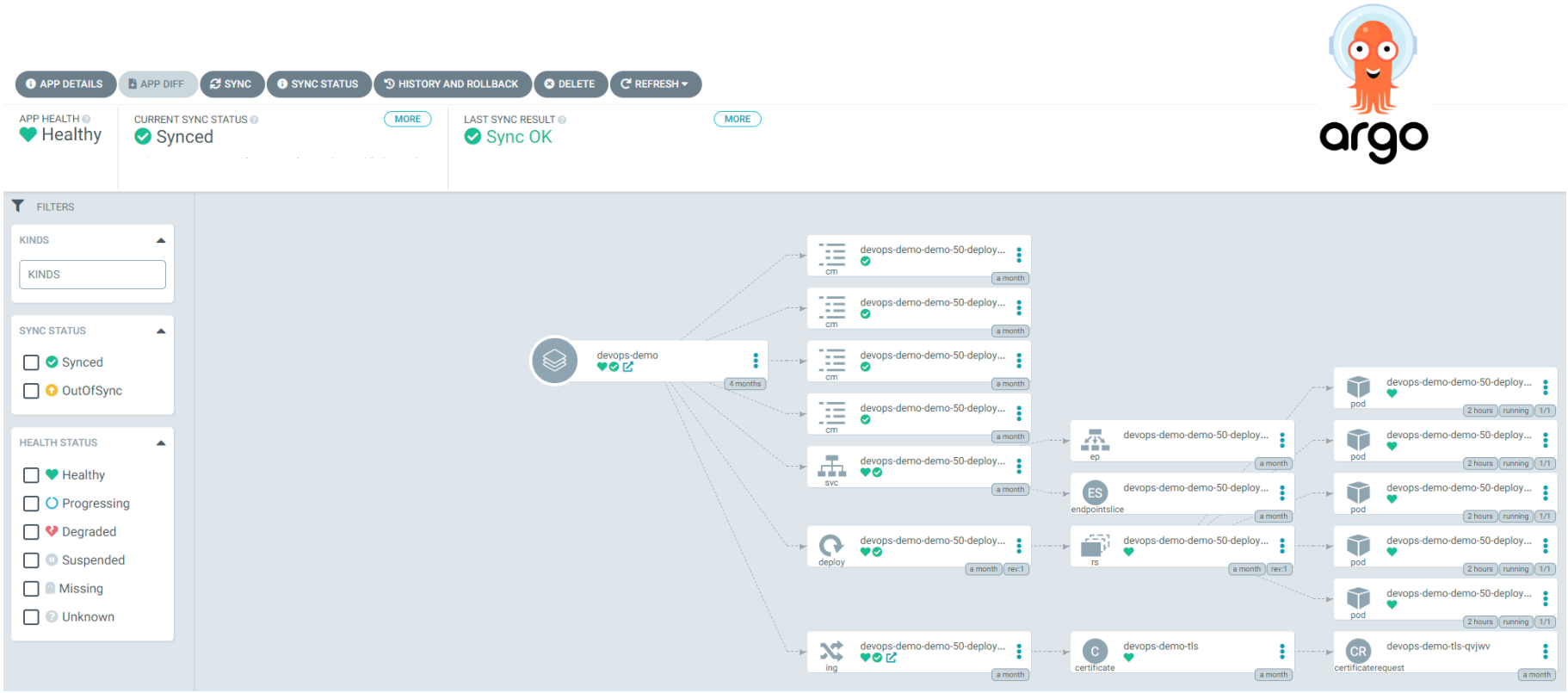
- Vorgabe von technischen und sicherheitsrelevanten Mindeststandards
  - Best Practices für die Anwendungsentwicklung
    - Container Images sind mit minimalen Privilegien und auf keinen Fall mit root-Privilegien gebaut.
    - Container Images sind mit minimalem Inhalt gebaut.
    - Container Images sind mit aktuellen Paketen gebaut und werden regelmäßig aktualisiert.
    - Ressourcenanforderungen sind konfigurierbar und vernünftige Werte voreingestellt.
    - Die Anwendung und ihre Konfigurationsmöglichkeiten sind vollständig dokumentiert.
    - Es gibt einfach zugängliche Schnittstellen für die Konfiguration.
    - Secrets in Helm Values lassen sich vermeiden.

Deployment

- Administration von Anwendungen mittels Git-Repository
  - kein Kuberntes-Expertenwissen mehr nötig
- CD-Pipeline zum automatischen Deployment
  - keinerlei direkte Interaktion mit dem Cluster
  - kein Umgang mit K8s-CLI
  - Know-How zum Bau von Pipelines
  - Permanenter Abgleich zwischen K8s-Cluster und Git-Repository
- Definierte Schnittstelle zur Übergabe
  - Pull-Prinzip
  - Auslieferungen über Git-Repositories und Container-Registries
  - Konfigurationsänderungen bei Versionsupdates sind dokumentiert
  - Die Anwendung besteht aus einer minimalen Anzahl an einzelnen Deployments

argo

Deployment


APP DETAILS
APP DIFF
SYNC
SYNC STATUS
HISTORY AND ROLLBACK
DELETE
REFRESH
APP HEALTH
Healthy
CURRENT SYNC STATUS
Synced
MORE
LAST SYNC RESULT
Sync OK
MORE
argo
FILTERS
KINDS
KINDS
SYNC STATUS
Synced
OutOfSync
HEALTH STATUS
Healthy
Progressing
Degraded
Suspended
Missing
Unknown
devops-demo
4 months
cm
svc
deploy
ing
ep
endpointslice
rs
pod
certificate
certificaterequest
devops-demo-tls
devops-demo-tls-qvjwv
a month
2 hours
running
1/1
rev:1

Deployment

Basisimages

- Automatisierter Bau von Basisimages mittels Pipeline
- Aktuell Prototyp mit Ubuntu/Java/Tomcat
- Bereitstellung über gesonderten Basis-Image-Harbor

Entwicklung

- IT-DLZ intern oder Extern durch Kunden oder Dienstleister
- Bereitstellung der entwickelten Anwendung (Helm-Charts und Images)
  Alternative: eigenständiger Anwendungsbetrieb durch Kunden

Deployment

- Download der Anwendung (Images und Helm-Charts) -> PULL-Verfahren
- Ggf. Anpassung der Konfiguration (Values) an das IT-DLZ
- Deployment der Anwendung (zukünftig mittels CI/CD-Pipeline)

argo

Plattform und Infrastruktur-betrieb

- Bereitstellung der K8s-/Rancher-Cluster und der Basis-Infrastruktur

IT-Sicherheit

- Verarbeitung sensibler Daten in E-Government-Verfahren durch staatliche Stellen
- Schutz personenbezogener Daten der Bürgerinnen und Bürger hat oberste Priorität!
- Gewährleistung maximaler Sicherheit

- Sicherheitsrichtlinien für den Containerbetrieb
  - PodSecurity und Network-Policies
  - technische und sicherheitsrelevante Mindeststandards
- Berechtigungskonzepte für das Kubernetes-Ökosystem
- Sicherheitskonzepte

- Umsetzung erfolgt nach den Standards des BSI

IT-Sicherheit

BSI - Bundesamt für Sicherheit in der Informationstechnik

- IT-Grundschutz-Kompendium
- Bausteine
  - ISMS: Sicherheitsmanagement
  - ORP: Organisation und Personal
  - CON: Konzepte und Vorgehensweisen
  - OPS: Betrieb
  - DER: Detektion und Reaktion

  - APP: Anwendungen
  - SYS: IT-Systeme
  - IND: Industrielle IT
  - NET: Netze und Kommunikation
  - INF: Infrastruktur

IT-Sicherheit

BSI - Bundesamt für Sicherheit in der Informationstechnik

- Schutzbedarfsfeststellung
  - Vertraulichkeit
  - Integrität
  - Verfügbarkeit
- Schutzbedarfskategorien
  - „normal“
  - „hoch“
  - „sehr hoch“
- Sicherheitskonzept
  - Informationsverbund
  - Modellierung
    - Anforderungen und Maßnahmen
    - MUSS oder SOLL
  - Risikoanalyse
- Zertifikat

IT-Sicherheit

SYS.1.6 Containerisierung

- A1 Planung des Container-Einsatzes (B)
- A2 Planung der Verwaltung von Containern (B)
- SYS.1.6.A3 Sicherer Einsatz containerisierter IT-Systeme (B)
- A4 Planung der Bereitstellung und Verteilung von Images (B)
- A5 Separierung der Administrations- und Zugangsnetze bei Containern (B)
- A6 Verwendung sicherer Images (B)
- A7 Persistenz von Protokollierungsdaten der Container (B)
- A8 Sichere Speicherung von Zugangsdaten bei Containern (B)
- A9 Eignung für Container-Betrieb (S)
- A10 Richtlinie für Images und Container-Betrieb (S)
- A11 Nur ein Dienst pro Container (S)
- A12 Verteilung sicherer Images (S)
- A13 Freigabe von Images (S)
- A14 Aktualisierung von Images (S)
- A15 Limitierung der Ressourcen pro Container (S)
- A16 Administrativer Fernzugriff auf Container (S)
- A17 Ausführung von Containern ohne Privilegien (S)
- A18 Accounts der Anwendungsdienste (S)
- A19 Einbinden von Datenspeichern in Container (S)
- A20 Absicherung von Konfigurationsdaten (S)

APP.4.4 Kubernetes

- A1 Planung der Separierung der Anwendungen (B)
- A2 Planung der Automatisierung mit CI/CD (B)
- A3 Identitäts- und Berechtigungsmanagement bei Kubernetes (B)
- A4 Separierung von Pods (B)
- A5 Datensicherung im Cluster (B)
- A6 Initialisierung von Pods (S)
- A7 Separierung der Netze bei Kubernetes (S)
- A8 Absicherung von Konfigurationsdateien bei Kubernetes (S)
- A9 Nutzung von Kubernetes Service-Accounts (S)
- A10 Absicherung von Prozessen der Automatisierung (S)
- A11 Überwachung der Container (S)
- A12 Absicherung der Infrastruktur-Anwendungen (S)

IT-Sicherheit

APP.4.4.A1 Planung der Separierung der Anwendungen (B)

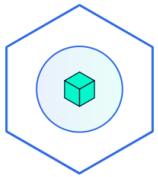
- Vor der Inbetriebnahme MUSS geplant werden, wie die in den Pods betriebenen Anwendungen und deren unterschiedlichen Test- und Produktions-Betriebsumgebungen separiert werden.
- Auf Basis des Schutzbedarfs der Anwendungen MUSS die Planung festlegen, welche Architektur der Namespaces, Meta-Tags, Cluster und Netze angemessen auf die Risiken eingeht und ob auch virtualisierte Server und Netze zum Einsatz kommen sollen.
- Die Planung MUSS Regelungen zu Netz-, CPU- und Festspeicherseparierung enthalten. Die Separierung SOLLTE auch das Netzzonenkonzept und den Schutzbedarf beachten und auf diese abgestimmt sein.
- Anwendungen SOLLTEN jeweils in einem eigenen Kubernetes-Namespace laufen, der alle Programme der Anwendung umfasst. Nur Anwendungen mit ähnlichem Schutzbedarf und ähnlichen möglichen Angriffsvektoren SOLLTEN einen Kubernetes-Cluster teilen.

Links:

- IT-Grundschutz-Kompendium
- SYS.1.6 Containerisierung
- APP.4.4 Kubernetes

Challenge accepted?

- Change und Releasemanagement von Images
- Freigabe von entwickelten Anwendungen/Images
- Maßnahmen im Rahmen von IT-Sicherheit
- Aufbau der CI/CD-Pipelines
- Logging und Monitoring
- Bereitstellung, Updates und Anpassungen/Upgrades an Kubernetes-Clustern
- Anwendungsentwicklung im Containerumfeld
- Betrieb von Containeranwendungen



Lessons Learned

klassische IT-Welt

Kubernetes-Ökosystem

Lessons Learned

Organisatorisch

- bestehende Prozesse und Schnittstellen
- definierte Verantwortlichkeiten
- starke Standardisierung in Services

Technisch

- Anbindung von Services das K8s-Ökosystem
- Stetige Evolution von Kubernetes und der gesamten Softwarelandschaft
  - Migration von Suse CaaSP – Rancher
  - Storage, Pipelining, Secrets
  - Signatur von Images

Vielen Dank für die Aufmerksamkeit!

Fragen?