IT-Planungsrat

Deutsche Verwaltungscloud-Strategie
Proof-of-Concept

Ergebnisbericht

– Version 20. April 2022 –

Impressum

Herausgeber

FITKO (Föderale IT-Kooperation)

Zum Gottschalkhof 3

60594 Frankfurt am Main

E-Mail: poststelle@fitko.de

Anstalt des öffentlichen Rechts | Präsidentin: Dr. Annette Schmidt

Ansprechpartner

Referat DG II 2 „Digitale Souveränität für die IT der öffentlichen Verwaltung“

Bundesministerium des Innern und für Heimat

Postanschrift: Alt-Moabit 140, 10557 Berlin

Hausanschrift: Salzufer 1 (Zugang Englische Straße), 10587 Berlin

E-Mail: DGII2@bmi.bund.de

www.cio.bund.de

Stand

April 2022

Nachdruck, auch auszugsweise, ist genehmigungspflichtig.

Inhaltsübersicht

1. Einleitung

1.1 Ausgangslage

In der 33. Sitzung des IT-Planungsrates (IT-PLR) wurde das Konzeptpapier zur Deutschen Verwaltungscloud-Strategie – Föderaler Ansatz beschlossen[1]. Die Maßnahme ist Teil der beschlossenen Strategie zur Stärkung der Digitalen Souveränität der IT der Öffentlichen Verwaltung (ÖV)[2] und ist dem definierten Lösungsansatz „Herstellerunabhängige Modularität, (offene) Standards und Schnittstellen in der IT“ der Strategie zugeordnet.

Die im Oktober 2020 durch den IT-PLR beschlossene Deutsche Verwaltungscloud-Strategie (DVS, DVS steht hier auch für „Deutsche Verwaltungscloud“) soll gemeinsame Standards und offene Schnittstellen für Cloud-Lösungen der ÖV schaffen, um übergreifend eine interoperable sowie modulare föderale Cloud-Infrastruktur zu etablieren.

Mit dem Beschluss Nr. 2020/54 des IT-PLR wurde die Arbeitsgruppe Cloud-Computing und Digitale Souveränität (AG Cloud) beauftragt, die Zielarchitektur der DVS zu erarbeiten. Die AG Cloud hat auf Grundlage der Entscheidung des IT-PLR die technische Konzeption und Operationalisierung an die Unterarbeitsgruppe Technik & Betrieb (UAG Technik) übergeben. Parallel zur Erarbeitung des Rahmenwerks der Zielarchitektur wurde ein erstes Pilotierungsprojekt gestartet. Dieses Pilotierungsprojekt im Rahmen eines Proof-of-Concepts (PoCs) sollte ausgewählte User Stories (siehe Kapitel 1.3) erproben.

Das Rahmenwerk der Zielarchitektur wurde in der 36. Sitzung des IT-PLR beschlossen[3]. Mit dem Abschluss des PoCs im Januar 2022 werden die Arbeitsergebnisse an die UAG Technik und an die AG Cloud zurückgespielt und in geeigneten Gruppen, z.B. in der „Interessengruppe Betrieb von Containern“ (IG BvC) weiterentwickelt und bei der kontinuierlichen Detaillierung der Standards für die DVS berücksichtigt.

Ein übergreifendes technisches Glossar für die DVS befindet sich in Erstellung.

[1] IT-PLR, Beschluss 2020/54 vom 28.10.2020.

[2] IT-PLR, Beschluss 2021/09 vom 17.03.2021.

[3] IT-PLR, Beschluss 2021/46 vom 29.10.2021.

1.2 Grundzüge der Deutschen Verwaltungscloud-Strategie

Das Rahmenwerk der Zielarchitektur der DVS setzt den Rahmen für die Umsetzung der DVS und damit auch für den PoC. An dieser Stelle sollen die Grundzüge der Zielarchitektur, in dessen Kontext der erste PoC durchgeführt wurde, kurz skizziert werden.

Die DVS beschreibt die übergreifende Etablierung und Nutzung von Standards für bestehende föderale Cloud-Lösungen der ÖV. Die Standardisierung soll die wesentlichen Bereiche der Cloud-Architekturschichten von Entwicklung über Inbetriebnahme bis zum Betrieb von Anwendungen beinhalten. So werden ein verteilter Betrieb und eine standardisierte Bereitstellung in Rechenzentren von Bund, Ländern und Kommunen ermöglicht. Dabei wird Open Source Software (OSS) als geeignetes Mittel für den Aufbau einer vernetzten Cloud-Infrastruktur angesehen und dabei bei der Lösungserstellung priorisiert. Die erweiterte Zusammenarbeit zwischen Betreibern von Cloud-Standorten schafft zudem Synergieeffekte über den gesamten Software-Lebenszyklus hinweg.

Ein besonderer Fokus liegt auf der Gestaltung und Ausrichtung der DVS nach dem Prinzip „privacy by design / security by design“, die die Sicherheitsanforderungen über alle föderalen Ebenen hinweg berücksichtigen. Die strenge Ausrichtung der definierten Standards an bestehenden Richtlinien/Vorgaben für IT-Sicherheit unterstützt dabei, die Informationssicherheit der Infrastruktur weiter zu stärken.

Im Rahmenwerk der Zielarchitektur wurden bereits wesentliche Standards für Cloud-Standorte definiert, die im PoC ihre Anwendung fanden.

1.3 Zielstellung des Proof-of-Concept

Der erste DVS PoC fokussierte sich auf die Interaktion zwischen Softwarebetreiber/-lieferant und Plattformbetreiber[4] (siehe Abbildung 1). Ziel war es, gleichartige Cloud-Services für Softwarebetreiber an unterschiedlichen Cloud-Standorten bereitzustellen. Die Lauffähigkeit der Cloud-Services sollte dabei anhand von anschaulichen Anwendungen für den Endnutzer nachvollziehbar sein. Gemäß den Anforderungen der DVS sollte für alle Bereiche OSS priorisiert werden.

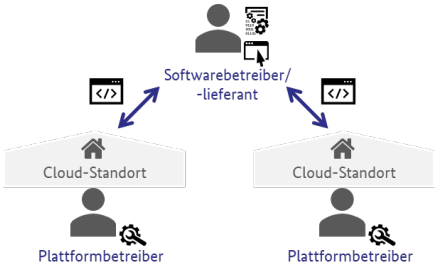

Abbildung 1: Schematischer Fokus des PoCs

[4] Festlegung der Rollen gemäß „Deutsche Verwaltungscloud-Strategie: Rahmenwerk der Zielarchitektur“.

Die Spezifizierung des Umfangs des PoCs wurde durch prioritäre und nachgeordnete *User Stories* vorgenommen. User Stories beschreiben Anforderungen einzelner Rollen (hier z.B. vom Softwarebetreiber) an andere und deren Interaktionen und geben damit einen Einblick, was innerhalb der DVS perspektivisch ermöglicht werden soll. Tabelle 1 gibt einen Überblick über die durch den PoC abzudeckenden User Stories.

Beschreibung der priorisierten User Stories
Als **Softwarelieferant** möchte ich definierte Richtlinien für Software erhalten, um weitestgehend plattformunabhängig entwickeln und in verschiedenen Cloud-Standorten testen zu können.
Als **Softwarelieferant** möchte ich in Rechenzentren der ÖV einfach und ohne großen Anpassungsaufwand Softwarelösungen zum Testen ausrollen können.
Als **Softwarebetreiber** möchte ich containerisierte Anwendungen in unterschiedlichen Cloud-Standorten betreiben, um Wiederverwendbarkeit zu erzielen.
Als **Softwarebetreiber** möchte ich einheitliche / gleichartige Container-Cluster bereitgestellt bekommen, um das Deployment von Anwendungen zu erleichtern.
Als **Softwarebetreiber** möchte ich die Kompatibilität des Cloud-Standortes für meine benötigten Ressourcen überprüfen, um den einwandfreien Betrieb zu gewährleisten.
Als **Plattformbetreiber** möchte ich ein einheitliches und gemeinsames Regelwerk für die Konfiguration von Container-Clustern auf Basis von OSS benutzen, um standardisierte Cloud-Services bereitstellen zu können.
Beschreibung der User Stories nachgeordneter Priorität
Als **Softwarebetreiber** möchte ich Standard-Images aus einer (zentralen) Container-Registry beziehen, um Anwendungen zu nutzen.
Als **Softwarebetreiber** möchte ich Zugriff auf die Container-Registry des Plattformbetreibers, um meine Anwendungen zu deployen.
Als **Softwarebetreiber** möchte ich einen Zugang seitens des Plattformbetreibers zur Verfügung gestellt bekommen, um Cloud-Services am Cloud-Standort nutzen zu können.
Als **Softwarebetreiber** möchte ich Anforderungen (insb. notwendige Infrastruktur, Netztopologien) an den Cloud-Service des Plattformbetreibers definieren können, um den benötigten Zielzustand zu erhalten.

Als **Plattformbetreiber** möchte ich innerhalb der DVS einen Open Source-Software-Stack (z. B. Sovereign Cloud Stack, SCS) nutzen können, um Abhängigkeiten zu anderen Anbietern zu verringern.

Tabelle 1: Beschreibung der priorisierten und nachgeordneten User Stories für den DVS PoC

1.4 Projektorganisation

Die Umsetzung des PoCs erfolgte vom 1. Juli 2021 bis zum 14. Januar 2022 durch die UAG Technik mit organisatorischer Unterstützung durch das Bundesministerium des Innern und für Heimat (BMI). Aufbauend auf den User Stories wurde eine auf Arbeitspaketen (AP) aufbauende Aufgabenstruktur definiert. Anhand eines agilen Vorgehens in monatlichen Entwicklungssprints wurden die Arbeitsergebnisse der AP technisch erprobt und iterativ weiterentwickelt. Abbildung 2 skizziert das Vorgehen.

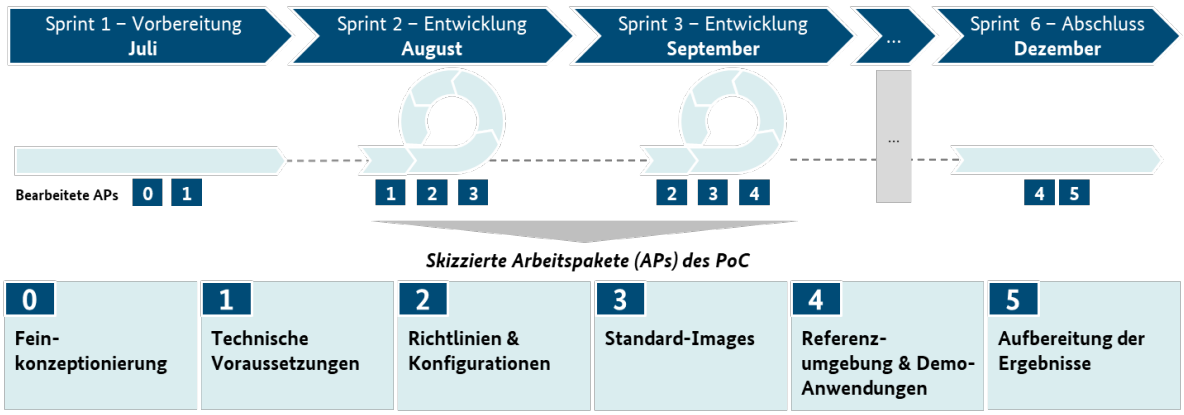

Abbildung 2: Arbeitsstruktur des PoCs

Die einzelnen Arbeitspakete und ihre festgelegten Ziele werden im Folgenden zusammengefasst.

Arbeitspaket 0: Feinkonzeptionierung

In AP 0 erfolgte die Planung der konkreten Ausgestaltung des PoCs. Die AP und die zu erreichenden Ziele wurden definiert, und die für die einzelnen AP verantwortlichen Kernteams festgelegt. Es wurde sich auf eine wöchentliche Abstimmungsrunde für alle Teilnehmenden am PoC verständigt.

Arbeitspaket 1: Bereitstellung der technischen Voraussetzungen

AP 1 teilte sich in zwei Unterarbeitspakete auf:

- AP 1.1: Bereitstellung von Testclustern durch beteiligte Datenzentralen,
- AP 1.2: Installation und Bereitstellung eines zentralen Repositorys zur Verwaltung von Standard-Images, Demoanwendungen und Richtlinien.

Übergeordnetes Ziel war es, die Container-Cluster bereitzustellen und deren Zugangsmöglichkeiten zu dokumentieren. Eine detaillierte Beschreibung der Ergebnisse erfolgt in Kapitel 2. Darüber hinaus sollte eine Entwicklungsumgebung und eine zentrale Registry für den PoC eingerichtet werden. Letzteres wurde mittels der OS-Plattform der ÖV Open CoDE[5] realisiert.

Arbeitspaket 2: Spezifikation und Umsetzung von Richtlinien und Konfigurationen

AP 2 teilte sich in fünf Unterarbeitspakete auf:

- AP 2.1: Spezifikation einheitlicher Richtlinien und Konfigurationen,
- AP 2.2: Bereitstellung der Richtlinien und Konfigurationen über das Repository,
- AP 2.3: Inbetriebnahme und Test der Anwendbarkeit der Richtlinien und Konfigurationen,
- AP 2.4: Entwicklung einer Testanwendung zur Prüfung der Kompatibilität der Container-Cluster hinsichtlich der Spezifikationen[6],
- AP 2.5: Durchführung von Konformitätstests mit dem Referenzdeployment in den Testclustern.

Im AP 2 sollten die einzusetzenden Werkzeuge zur Durchsetzung von Richtlinien (Policy Engines) sowie zur Prüfung von Richtlinienkonformität (Konformitätstests) definiert werden und eine Sammlung anwendbarer Richtlinien zusammengestellt werden. Es sollten Vorgaben für die Richtlinien als Code spezifiziert werden, auf deren Basis Konfigurationsdateien erstellt und dokumentiert werden sollten. Die Richtlinien und Konfigurationen sollten unter Anwendung der einzusetzenden Werkzeuge in den verschiedenen Datenzentralen ausgerollt und getestet werden; etwaige auftretende Probleme sollten dokumentiert werden. Für die Abfrage der Umsetzung der Richtlinien (Konformitätstest) sollte Software entwickelt oder genutzt werden, die ihr Ergebnis in einem übersichtlichen Berichtsformat ausgibt. Diese Erweiterung sollte in den Clustern getestet werden, sodass aus jedem Cluster ein Bericht der Kompatibilitätsprüfung vorliegt. Die Ergebnisse des AP 2 sind in Kapitel 3 ausführlich zusammengefasst.

Arbeitspaket 3: Standardisierung der Standard-Images

AP 3 teilte sich in drei Unterarbeitspakete auf:

- AP 3.1: Erarbeitung einer Systematik für Standard-Images,
- AP 3.2: Entwicklung ausgewählter Standard-Images zur Nutzung in Demo-Anwendungen,

[5] Open CoDE ist ein Projekt der ÖV, initiiert durch das BMI sowie die Länder Baden-Württemberg und Nordrhein-Westfalen und Teil der DVS. Derzeit bietet es ein zentrales und durchsuchbares Verzeichnis der verfügbaren Projekte und Lösungen, eine GitLab-Plattform zur Ablage von offenen Quellcodes und zur Beteiligung an Projekten sowie eine Diskussionsplattform. https://gitlab.o4oe.de/

[6] Der ursprüngliche Name des AP lautete „Entwicklung eines Referenzdeployments zur Prüfung der Kompatibilität der Container-Cluster hinsichtlich der Spezifikationen“, wurde aber im Laufe des PoCs präzisiert.

- AP 3.3: Aufbau einer Community zur kontinuierlichen Weiterentwicklung der Standard-Images.

Das Ziel des AP 3 war es, eine Begriffsdefinition für ein „Standard-Image“ für die DVS vorzunehmen und eine Systematik für sowie Anforderungen an solche Standard-Images im Rahmen der DVS zu erarbeiten. Auf Basis dessen sollten für die DVS umzusetzende Standard-Images festgelegt, implementiert und getestet werden. Langfristig sollte eine Community aufgebaut werden, die die Standard-Images kontinuierlich weiterentwickelt. Die Ergebnisse des AP 3 werden in Kapitel 4 erörtert.

Arbeitspaket 4: Test der Referenzumgebung mit Demoanwendungen

AP 4 teilte sich in zwei Unterarbeitspakete auf:

- AP 4.1: Bereitstellung der Demoanwendungen für das Deployment durch die Softwarelieferanten,
- AP 4.2: Herstellung der technischen Voraussetzungen und Deployment der Demoanwendungen in den bereitgestellten Testclustern.

AP 4 sollte sich mit der Auswahl und der Dokumentation der Anforderungen möglicher Demoanwendungen für den PoC befassen. Anschließend sollten die Testanwendungen in den Clustern ausgerollt werden und potenzielle Herausforderungen bei der Bereitstellung dokumentiert werden. Die Kandidaten für die Testanwendungen und die weiteren Ergebnisse des AP 4 werden in Kapitel 5 detailliert.

Arbeitspaket 5: Aufbereitung der Ergebnisse

AP 5 teilte sich in drei Unterarbeitspakete auf:

- AP 5.1: Dokumentation der Ergebnisse,
- AP 5.2: Initiierung einer Community für Weiterentwicklung und Pflege der Lösungen,
- AP 5.3: Identifikation notwendiger Weiterentwicklungen für Kompatibilität zum SCS.

Ziel des AP 5 war es, eine ausführliche Dokumentation der Ergebnisse des PoCs anzufertigen, um die Nachnutzung sicherzustellen. Darüber hinaus sollte eine Community entwickelt werden, die die Ergebnisse des PoCs weiterträgt und weiterentwickelt. Unter anderem sollten auf Open Source (OS) basierende Projekte für Richtlinien und Basisimages ins Leben gerufen werden. Zudem sollte notwendiger Entwicklungsbedarf für die Kompatibilität zum SCS identifiziert werden. Die Dokumentation der Ergebnisse erfolgt mit diesem Dokument, auf weitere Ergebnisdokumentation wird verwiesen. Im Rahmen der Nutzung eines zentralen Repositorys zur Verwaltung von

Standard-Images, Demoanwendungen und Richtlinien (AP 1.2) wurde ein Grundstein für die Weiterentwicklung und Pflege entstandener Lösungen gelegt. Die Kompatibilität zum SCS wurde aus Zeit- und Kapazitätsgründen zum Ende des PoCs nicht geprüft.

1.5 Projektteilnehmende

Insgesamt beteiligten sich 13 Datenzentralen aktiv an der Umsetzung, sechs weitere unterstützten in einer beratenden Rolle. Von den aktiv beteiligten Datenzentralen stellten acht jeweils einen Container-Cluster bereit. Neben IT-Dienstleistern der Länder und Kommunen nahm mit dem Bundesrechenzentrum (BRZ) auch ein europäischer Partner aus Österreich teil. Abbildung 3 listet die einzelnen Partner und ihre Rolle im PoC auf.

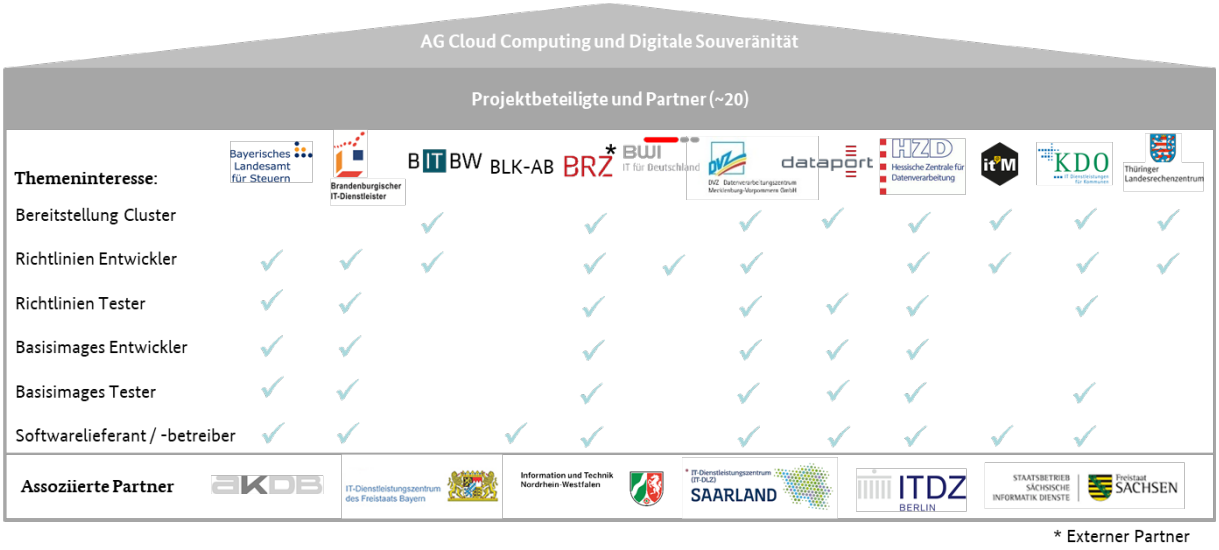

Abbildung 3: Projektbeteiligte und Partner des PoCs

Zwischen den Datenzentralen bestanden zu Beginn des PoCs teils sehr verschiedene Voraussetzungen – sowohl hinsichtlich der Wissensstände als auch in Bezug auf die technischen Ausprägungen. So waren beispielsweise Datenzentralen vertreten, die bereits Erfahrungen im produktiven Betrieb von Container-Clustern vorweisen konnten (sechs der acht Datenzentralen), aber auch solche, bei denen im Rahmen des PoCs Containerlösungen erstmals betrieben worden sind. Wie in der Planung angestrebt waren bei den Datenzentralen verschiedene Software-Grundlagen in Hinblick auf Virtualisierung und Containerisierung vorhanden. Unter diesen fanden sich OpenShift, VMware und native Open Source-Lösungen, teilweise auf Basis von Rancher (s. a. Tabelle 2).

2. Clusterbereitstellung

Die Bereitstellung der Cluster erfolgte in Eigenverantwortung der teilnehmenden Datenzentralen planmäßig und termingerecht zum 01.10.2021. Für jede Datenzentrale mit bereitgestelltem Cluster war ein fester Ansprechpartner vertreten. Vorab wurde zunächst die Dimensionierung und Ausstattung der einzelnen Cluster festgelegt; diese hat sich im Laufe des PoCs als adäquat erwiesen. Die Container-Orchestrierung erfolgte übergreifend in Kubernetes. Tabelle 2 gibt einen Überblick über die Datenzentralen und die unterschiedlichen Plattformen.

Datenzentrale	OS	Administration
Dataport AöR	Ubuntu	Rancher
DVZ-MV	Ubuntu	TKGI
BitBW	Fedora CoreOS (Red Hat CoreOS)	OKD
KDO	CentOS 7	Kubectl
BRZ	RedHat CoreOS	OpenShift
it@M	RedHat CoreOS	OpenShift
TLRZ	Oracle Enterprise Linux	Rancher
Komm.ONE	Ubuntu	Rancher

Tabelle 2: Übersicht über die Datenzentralen mit bereitgestelltem Cluster und deren Plattformen

Der Erfüllungsgrad der Bereitstellung der Cluster lag bei 100%: Alle Cluster konnten wie geplant bereitgestellt werden und waren auch von extern erreichbar. Die vorab festgelegten Mindestanforderungen an Master und Worker (Anzahl, Funktion, Dimensionierung) wurden erfüllt, insbesondere stand in allen Clustern auch persistenter Speicher zur Verfügung. Wie auch in den Plattformen wurden von den verschiedenen Datenzentralen auch zur Bereitstellung des persistenten Speichers unterschiedliche Technologien verwendet, darunter Rancher Longhorn, VMware vSAN (CNS) und NetApp Trident.

Das Ziel, ein möglichst diverses Umfeld an genutzten Technologien zu bespielen, um das Konzept DVS an einer möglichst großen Bandbreite an Produkten und Herstellern zu testen, wurde somit erreicht.

2.1 Parameter für die Bereitstellung von Container-Clustern und Namespaces

Im Rahmen des PoCs wurde ein Katalog an Parametern zusammengestellt, die für die Bereitstellung eines Container-Clusters bzw. zur Bereitstellung einzelner Namespaces benötigt werden.

Diese werden an die UAG Technik und ihre Handlungsfelder zurückgespiegelt, um sie bei der weiteren Konzeption der DVS sowie insbesondere des Cloud-Service-Portals und des Servicekatalogs zu berücksichtigen. Die Parameter sind im Folgenden aufgeführt.

Für die Clusterbereitstellung notwendige Konfigurationsdaten:

- Anzahl Master,
- Anzahl Worker,
- Technische Ausstattung der Worker (CPU, RAM, Storage),
- Art und Größe des persistenten Speichers,
- Notwendige Netzanbindungen außerhalb des Clusters (bspw. für einen Datenbankserver außerhalb des Clusters, Verzeichnisdienste oder andere Cloudlösungen),
- Strukturierung des Clusters mit Namespaces, ggf. inkl. Angabe von Ressourcen (s.u.).

Für die Bereitstellung einzelner Namespaces notwendige Konfigurationsdaten:

- Technische Ausstattung (CPU, RAM, Storage, Anzahl der Worker),
- Notwendige Netzanbindungen außerhalb des Namespaces,
- Gewünschtes homogenes Schutzniveau der Anwendungen im Cluster (z.B. Erfüllung Schutzbedarf „hoch“ für Vertraulichkeit).

Darüber hinaus wird für jede Containerumgebung eine eigene produktive Container-Registry benötigt.

3. Richtlinien für Cluster

Im Rahmen des PoCs, aber auch allgemein, basieren die Kubernetes-Cluster der verschiedenen Cloud-Standorte auf unterschiedlichen technischen Voraussetzungen (Kapitel 2). Im Zielbild der DVS sollen diese Cluster aus Sicht der Fachanwendung jedoch gleichartig sein, damit diese einheitlich betrieben werden können. Insbesondere sollen keine (bzw. nur über die Konfiguration) Anpassungen an Fachanwendungen notwendig sein, um diese datenzentralenübergreifend zu betreiben. Daher kommen in der DVS gemeinsame Standards zum Einsatz. Wenn diese maschinenlesbar gestaltet werden können, können diese automatisiert in den Datenzentralen überprüft werden, sodass eine Einhaltung der Standards garantiert werden kann. Im Rahmen des PoCs wurden solche Standards als Richtlinien erstellt und unter Anwendung von Policy Engines umgesetzt. Mittels Konformitätstest wurde die Einhaltung der Richtlinien geprüft.

3.1 Richtlinienerstellung

Im Folgenden wird die im PoC gewählte Vorgehensweise zur Erstellung und Ausgestaltung von Policies näher erläutert. Eine Policy beschreibt eine Richtlinie für den Betrieb von Container-Clustern und darin betriebener Softwarelösungen in natürlicher Sprache, die als Code umgesetzt wurde und automatisiert in Kubernetes-Clustern durchgesetzt werden kann. Weitere Ausführungen zu diesem Thema finden sich in dem während des PoCs durch das AP 2.1 erstellte Dokument „Vorgaben für die Erstellung von IG BvC-Richtlinien als Code“ sowie „Festlegung einzusetzender Werkzeuge“[7].

Im Rahmen des PoCs wurden von der IG BvC auf Basis der IT-Grundschutz-Bausteine des Bundesamts für Sicherheit in der Informationstechnik (BSI)[8] definierte Richtlinien zum sicheren Betrieb von Containern als Quellcode umgesetzt und entsprechend in Policies überführt. Die Policies sind dabei entweder validierend und verifizierend, mutierend oder generierend[9]. Policies sollen keinen Konformitätstest ersetzen, da die Herstellung der Richtlinienkonformität dem Betreiber überlassen werden soll. Vorschläge für weitere Policies können durch angemeldete Nutzer im Richtlinien-Repository in der OS-Plattform der ÖV Open CoDE als Issues eingereicht werden. Gleiches gilt, falls sich Policies als praktisch untauglich erweisen. Grundsätzlich soll die Koordinierung der Erstellung zukünftig über die IG BvC erfolgen.

---

[7] Die Dokumente liegen derzeit in Code-Repositorys auf der OS-Plattform der ÖV Open CoDE ab. Sobald Open CoDE für die Öffentlichkeit verfügbar ist, können die Dokumente dort abgerufen werden.

[8] https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/IT-Grundschutz-Bausteine/Bausteine_Download_Edition_node.html

[9] Validierende und verifizierende Policies prüfen die Einhaltung der Richtlinien bzw. das Vorhandensein bestimmter digitaler Signaturen. Mutierende und generierende Policies verändern bestehende bzw. erstellen neue Ressourcen, um Konformität herzustellen.

Zur Durchsetzung der Richtlinien wurden verschiedene Lösungsansätze und Werkzeuge untersucht (sog. Policy Engines, darunter u.a. Open Policy Agent[10] und kyverno[11]) und kyverno in der API-Version kyverno.io/v1 als einzusetzendes Werkzeug festgelegt. Hierbei ist zu beachten, dass für teilnehmende Datenzentralen und die Container-Cluster keine Verpflichtung zur Nutzung von kyverno besteht, sondern die Möglichkeit offengehalten wird, dass Policies für alternative Policy Engines durch die Community bereitgestellt werden oder bereits in den Kubernetes-Distributionen vorhandene Eigenschaften genutzt werden. Maßgeblich ist die Einhaltung der Richtlinien, die durch das Bestehen der Konformitätstest nachgewiesen wird, nicht aber, welche Tools für die Durchsetzung der Richtlinien verwendet werden.

Für die Erstellung von Policies sind folgende Vorgaben als Arbeitsergebnis des PoCs entstanden:

- Jede Policy wird mittels Versionsverwaltung im Richtlinien-Repository auf der OS-Plattform der ÖV Open CoDE festgehalten.
- Alle Policies müssen getestet werden. Pro Policy müssen zwei Testfälle erstellt werden: Ein Positivfall (Test wird bestanden, die zu überprüfende Einstellung ist richtlinienkonform konfiguriert) und ein Negativfall (Test wird nicht bestanden).
- Alle Policies müssen in die CI-Pipeline des Richtlinien-Repository eingebunden und dort automatisierten Tests unterzogen werden.

Tabelle 3 gibt eine Übersicht über die im PoC umgesetzten Policies, deren Inhalt und die IT-Grundschutz-Anforderungen, die mit den Policies adressiert werden. Die Kurzbeschreibung der Policies ist dabei jeweils in englischer Sprache gehalten, was der gängigen Praxis in der Industrie entspricht. Die Grundlage für die genannten Policies ist der IT-Grundschutz-Baustein SYS.1.6 in der Version „Community Draft 2“ (Stand 24.03.2020).

Name der Policy	Kurzbeschreibung	IT-Grundschutz-Anforderung
disallow-add-capabilities	Capabilities permit privileged actions without giving full root access. Adding capabilities beyond the default set must not be allowed. This policy ensures users cannot add any additional capabilities to a pod.	SYS.1.6.A31
disallow-host-namespaces.yaml	Host namespaces (process ID namespace, inter-process communication namespace, and net-	SYS.1.6.A5

[10] https://www.openpolicyagent.org/

[11] https://kyverno.io/

	work namespace) allow access to shared information and can be used to elevate privileges. Pods should not be allowed access to host namespaces.	
disallow-privileged-containers.yaml	Privileged mode disables most security mechanisms and must not be allowed.	SYS.1.6.A21
disallow-selinux-options.yaml	SELinux options can be used to escalate privileges and should not be allowed. This policy ensures that the `seLinuxOptions` field is undefined.	SYS.1.6.A26
imagepullpolicy-always.yaml	Sample policy that sets imagePullPolicy to "always" when the "latest" tag is used.	SYS.1.6.A14
require-default-procmount.yaml	The default /proc masks are set up to reduce attack surface and should be required. This policy ensures nothing but the default procMount can be specified.	SYS.1.6.A5
require-health-and-liveness-check.yaml	Liveness and readiness probes need to be configured to correctly manage a pod's lifecycle during deployments, restarts, and upgrades. For each pod, a periodic `livenessProbe` is performed by the kubelet to determine if the pod's containers are running or need to be restarted. A `readinessProbe` is used by Services and Deployments to determine if the pod is ready to receive network traffic. This policy validates that all containers have liveness and readiness probes by ensuring the `periodSeconds` field is greater than zero.	SYS.1.6.A27
require-limits-and-requests.yaml	As application workloads share cluster resources, it is important to limit resources requested and consumed by each pod. It is recommended to require resource requests and limits per pod, especially for memory and CPU. If a namespace level request or limit is specified, defaults will automatically be applied to each pod based on the LimitRange configuration. This policy validates that all containers have something specified for memory and CPU requests and memory limits.	SYS.1.6.A16
require-non-root-groups.yaml	Containers should be forbidden from running with a primary or supplementary GID less than 1000. This policy ensures the `runAsGroup`, `supplementalGroups`, and `fsGroup` fields are set to a number less than 1000.	SYS.1.6.A26
require-run-as-non-root.yaml	Containers must be required to run as non-root users. This policy ensures `runAsNonRoot` is set to `true`.	SYS.1.6.A26

require_ro_rootfs.yaml	A read-only root file system helps to enforce an immutable infrastructure strategy; the container only needs to write on the mounted volume that persists the state. An immutable root filesystem can also prevent malicious binaries from writing to the host system.	SYS.1.6.A21
restrict-apparmor.yaml	On supported hosts, the 'runtime/default' AppArmor profile is applied by default. The default policy should prevent overriding or disabling the policy, or restrict overrides to an allowed set of profiles. This policy ensures pods do not specify any other AppArmor profiles than `runtime/default`	SYS.1.6.A26
restrict-image-registries.yaml	Images from unknown registries may not be scanned and secured. Requiring use of known registries helps reduce threat exposure.	SYS.1.6.A6
restrict-sysctls.yaml	Sysctls can disable security mechanisms or affect all containers on a host and should be disallowed except for an allowed "safe" subset. A sysctl is considered safe if it is namespaced in the container or the pod, and it is isolated from other pods or processes on the same node. This policy ensures that only those "safe" subsets can be specified in a pod.	SYS.1.6.A31

Tabelle 3: Umgesetzte Richtlinien

3.2 Erfahrungen mit den Policies

Ein wichtiges Ziel für das AP 2 war es, auftretende Hindernisse und Probleme bei der Umsetzung der für die DVS relevanten Richtlinien in den einzelnen Datenzentralen zu dokumentieren und ggf. entsprechende Lösungen zu entwerfen. Bei der Umsetzung der Policies hat sich gezeigt, dass eine Vorgabe von Richtlinien als Code im Kyverno Policy-Format[12] für viele Datenzentralen sinnvoll sein kann, aber auch die Umsetzung mit verschiedenen Werkzeugen ermöglicht werden sollte, um auf die Bedingungen in den einzelnen Datenzentralen Rücksicht zu nehmen.

Bei der Anwendung und Umsetzung der Richtlinien hat sich auch gezeigt, dass die für den PoC verwendeten Demo-Anwendungen (s. Kapitel 5) teilweise angepasst werden müssen, um diesen gerecht zu werden. Dies ist ein Hinweis darauf, dass möglicherweise viele Produkte den Anforderungen des IT-Grundschutzes des BSI bei Auslieferung noch nicht entsprechen.

[12] https://htmlpreview.github.io/?https://github.com/kyverno/kyverno/blob/main/docs/crd/v1/index.html#kyverno.io/v1.ClusterPolicy

Die in Kapitel 3.1 gelisteten Richtlinien stellen lediglich die Auswahl dar, die für diesen PoC verwendet worden sind. In weiteren PoCs und schließlich im Wirkbetrieb der DVS können problemlos zusätzliche Richtlinien ein- und umgesetzt werden. Allerdings ist dabei die kontinuierliche Weiterentwicklung der Richtlinien, beispielsweise aufgrund technologischen Fortschritts und neuer Sicherheitsvorgaben, notwendig.

3.3 Konformitätstests

Ursprünglich wurden die sich aus den Richtlinien der IG BvC ergebenden Policies zur Herstellung und Prüfung der Konformität der Container-Cluster gedacht. Ein positiver Seiteneffekt der Erstellung und Festlegung der Richtlinien als Code in Form von Policies ist, dass diese auch an anderen Stellen im Softwarelebenszyklus genutzt werden können, um bereits frühzeitig Fehlentwicklungen (in Form von Richtlinienverstößen) entgegenzuwirken. Grundsätzlich können Richtlinien also an allen Punkten des Anwendungslebenszyklus automatisiert mit geprüft werden (s. Abbildung 4):

1. Entwicklung (z.B. in CI-Pipelines): Durch eine statische Analyse der Kubernetes-Manifeste mit Command-Line-Tools oder in Test-Clustern beim Entwickler,
2. Deployment: durch die Prüfung via Admission-Controller im Zielrechenzentrum,
3. Betrieb: Durch kontinuierliche Prüfung, bspw. auf manuelle Veränderungen.

Bei Regelverstößen im ersten Fall erhält der Softwarelieferant eine Rückmeldung, dass seine Anwendung nicht richtlinienkonform ist und Anpassungen erforderlich sind, um die Kompatibilität herzustellen. Im zweiten Fall wird der Softwarebetreiber benachrichtigt, falls hier ein Regelverstoß festgestellt werden sollte. Dieser koordiniert dann die Ursachenbehebung. Im dritten Fall geht die Benachrichtigung an die Plattformbetreiber. In diesem Fall wurden die Kubernetes-Objekte (Manifeste) der Anwendung in einen nicht-regelkonformen Zustand gebracht. Hier gilt es zu prüfen, ob ein Angriff auf die Infrastruktur oder ein Administrationsfehler vorliegt.

Die Richtlinienkonformität der Container-Cluster lässt sich mittels eines Konformitätstests ermitteln. Im PoC wurde Sonobuoy[13] als zu verwendendes Diagnosewerkzeug festgelegt. Sonobuoy ermöglicht es, die Einhaltung von Richtlinien automatisiert zu testen und Abweichungen von den definierten Policies in einem entsprechenden Berichtsformat darzulegen.

[13] https://sonobuoy.io/

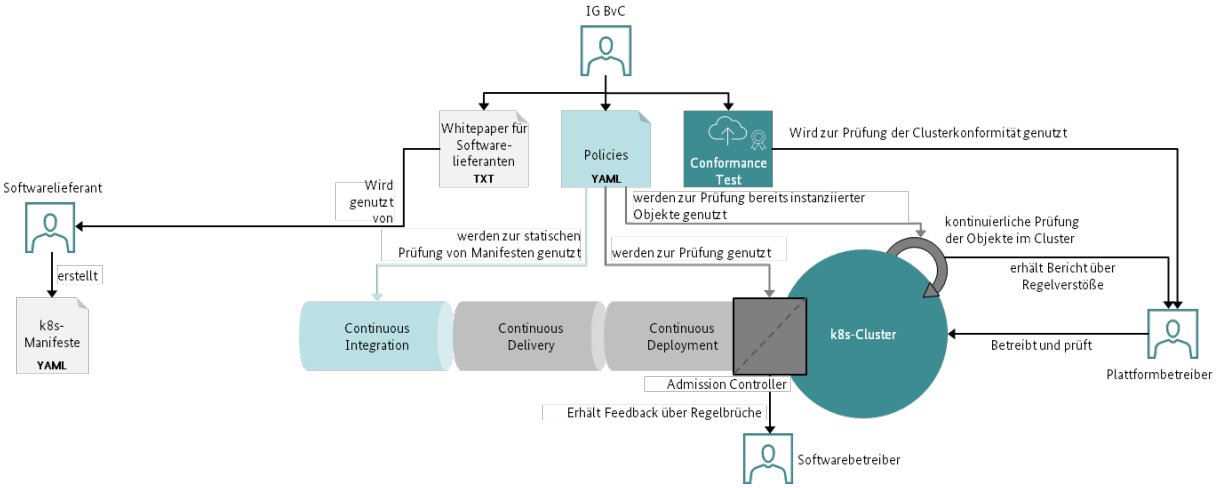

Abbildung 4: Einsatzgebiete von Policies (Richtlinien als Code) und Testanwendung (Konformitätstest) im Software-Lebenszyklus und deren Beziehungen zu den beteiligten Parteien.

3.4 Erfahrungen mit den Konformitätstests

Die automatisierte Einhaltung der Richtlinien und Durchführung der Konformitätstest durch die Cloud-Standorte hat sich als gut machbar erwiesen. Insbesondere stellen die Richtlinien eine adäquate Grundlage zur Durchsetzung der Standards der DVS dar. Die eingesetzten Technologien zum Policy Management waren weitgehend homogen. Die meisten Standorte setzten Kyverno[14] ein, das Bundesrechenzentrum aus Österreich nutzte den Open Policy Agent. Die Diversität in den Tools war jedoch einfach realisierbar, da in den Policy-Katalogen der beiden Werkzeuge[15] schon viele der benötigten Richtlinien vorhanden waren. Diese bildeten ein breites Spektrum der Anforderungen der einschlägigen Bausteine des IT-Grundschutzes des BSI bereits ab, sodass kaum Policy-Eigenentwicklungen notwendig waren. Es zeigte sich außerdem, dass die eingesetzten Containerplattformen einen Teil der Richtlinien bereits standardmäßig umsetzen.

Für die Bereitstellung und Entwicklung eines ausgereiften Produktes zur Prüfung der Kompatibilität, welches auch von Softwarelieferanten genutzt werden kann, bedarf es zusätzlicher Entwicklungskapazitäten, die im Rahmen des PoCs nicht gegeben waren.

[14] https://kyverno.io/

[15] Policy-Katalog Kyverno: https://github.com/kyverno/policies;
Policy-Katalog Open Policy Agent: https://github.com/open-policy-agent/gatekeeper-library

4. Standard-Images

Standard-Images bilden die Grundlage zur Bereitstellung von Fachanwendungen. Sie bestehen aus den wesentlichen Softwarekomponenten, die für die Bereitstellung einer Anwendung erforderlich sind. Im PoC wurde eine Systematik zur Definition von Standard-Images definiert, die folgende Begriffe umfasst:

- **Grund-Image:** Das Grund-Image bezeichnet ein Minimalimage, das nur die für eine Fachanwendung notwendigen Komponenten auf Betriebssystemebene enthält. Es handelt sich hierbei um eine Minimaldistribution eines Anbieters, etwa um eine bestimmte Linux-Distribution (z.B. Debian).
- **Runtime-Image:** Das Runtime-Image erweitert das Grund-Image um bestimmte Komponenten der Middleware und Frameworks, die für die Lauffähigkeit einer Anwendung notwendig sind.
- **Application-Image:** Das Application-Image erweitert ein Grund-Image oder Runtime-Image um anwendungsbezogene Komponenten. Mit dem Application-Image kann die Zielanwendung schließlich betrieben werden.

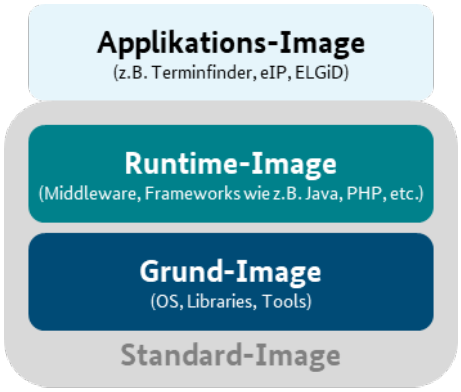

Abbildung 5: Aufbau der verschiedenen Image-Typen

Die o.g. verschiedenen Image-Typen bauen dabei aufeinander auf (s. Abbildung 5). Während Applikations-Images in vielen unterschiedlichen Konfigurationen vorliegen können, ist die Idee des Standard-Images, dass es die unveränderbare, standardmäßige Basis für eine oder mehrere Applikationen, unabhängig von deren Konfiguration, ist. Standard-Images können entweder nur aus dem Grund-Image oder aus dem Runtime-Image bestehen, das auf dem Grund-Image aufbaut. So wird eine Standardisierung der Grundlagen für verschiedene Applikationen erreicht. Zudem soll bei der Entwicklung der Images soll darauf geachtet werden, dass die Container die Standards der

Open Container Initiative[16] erfüllen, um einen möglichst hohen Standardisierungsgrad zu erreichen. Ein weiterer Vorteil (von Standard-Images) ergibt sich durch die überprüfbare Herkunft und durch die Versionierung der Images.

4.1 Organisation der Standard-Images

Um die unübersichtliche Ablage vieler verschiedener Versionsstände eines Images zu vermeiden und damit möglichen Sicherheitslücken vorzubeugen, soll die Image-Registry für Standard-Images (sowie für darauf basierende Images) einem regelmäßigen „Aufräum-Prozess“ („Clean-Up“) unterliegen. Dieser ist für die DVS noch zu definieren, genau wie die konkrete Ausgestaltung der Image-Registries. Im Rahmen des PoCs wurde für die Organisation der Ablage und Verwaltung der Standard-Images vom Code Repository der OS-Plattform der ÖV Open CoDE Gebrauch gemacht[17]. Zusätzlich wurde ein für den PoC eigens eingerichtetes Microsoft Sharepoint-Portal als Wissensspeicher verwendet. Das Repository ist aufgeteilt in Verzeichnisse für Grund-, Runtime- und Application-Images, in denen sich pro Image jeweils ein Projekt findet. Im PoC wurden bereits unterstützte Betriebssystem-Images als Basis für darauf aufbauende Runtime- und Applikationsimages identifiziert: CentOS, Debian und Alpine. Die dazugehörigen Projekte sind im Verzeichnis „Grund-Images“[18] zu finden. Die Organisation der (Weiter-) Entwicklung der Standard-Images im Rahmen von OS-Projekten, beispielsweise unter Nutzung des Code Repository der OS-Plattform der ÖV Open CoDE, wie es im Rahmen des PoC angestoßen worden ist, ist Teil der weiteren Umsetzung der DVS.

Für den Übergang vom Entwicklungsprozess bzw. der Entwicklungsumgebung in eine Test-, Referenz-, oder Produktivumgebung wurde festgehalten, dass das Image einen Scan durchlaufen muss, um die Erreichung einer ersten Qualitätsstufe sicherzustellen. Dies gilt sowohl für interne als auch für externe Entwicklungsprozesse. Nach dem Scan soll das Image durch die Stelle, die den Scan durchführt, signiert werden[19]. In der Test-, Referenz- und Produktivumgebung dürfen dann entsprechend nur signierte Images deployed werden.

4.2 Erfahrungen mit Image-Scannern

Zuverlässige, sichere Images sind die Voraussetzung zum zuverlässigen und sicheren Betrieb von Containern. Um diese zu gewährleisten, wurde im Rahmen des PoC zunächst ein Vorgehen zur

[16] https://opencontainers.org/

[17] Das Repository für die Standard-Images findet sich hier: https://gitlab.o4oe.de/ig-bvc/standard-images

[18] https://gitlab.o4oe.de/ig-bvc/standard-images/grund-images

[19] Im Rahmen des PoC waren das die teilnehmenden Datenzentralen. Dies kann sich in der weiteren Umsetzung der DVS noch ändern.

Bewertung und zur Bereitstellung von Standard-Images definiert (s. a. Kapitel 6). Dieses umfasst insbesondere einen „Image-Scan“, d.h. eine genaue Prüfung des zu betrachtenden Standard-Images. Im Rahmen des Image-Scans werden die Images auf CVEs (Common Vulnerabilities and Exposures), also bekannte Schwachstellen und Schadsoftware geprüft. Die Images werden weiterhin auf das Vorhandensein von Build Dependencies, also zusätzlichen Software-Bausteinen, die nur für den Build-Prozess vorhanden sein müssen, untersucht. Außerdem werden die Images dahingehend überprüft, ob jeweils nur die für den Betrieb der gewünschten Anwendung erforderlichen Software-Bausteine vorhanden sind. Für Produktiv-Umgebungen sind beispielsweise Werkzeuge wie Shell, Curl und Wget, sowie Paketmanager, Editoren und Compiler ausgeschlossen. Im Umfeld der DVS soll für die Images entsprechend auch die Zulässigkeit ihrer Konfigurationen sowie die Einhaltung definierter Standards (BSI IT-Grundschutz, Standards der Cloud Native Computing Foundation[20], Empfehlungen der IG-BvC, an denen sich auch die Richtlinien für Cluster orientieren, s. Kapitel 3), überprüft werden. Auch dies dient der Vermeidung von Sicherheitsrisiken. Es muss ebenso eine Auflistung aller erforderlichen Lizenzen für das Image erstellt und ggf. auf im Image selbst vorhandene Lizenzen überprüft werden. Beim Deployment der Images muss außerdem auf das Vorhandensein bestimmter Signaturen geprüft werden, damit die sichere Herkunft der Images gewährleistet ist. Um die fortlaufende Sicherheit der Images sicherzustellen, sollen diese außerdem kontinuierlich auf veraltete Komponenten bzw. Komponenten mit fehlendem oder ausgelaufenem Support („deprecated“) geprüft werden. Für Positivfälle im Image-Scan ist im Detailstandard zum Standard-Image-Bereitstellungs- und -Prüfprozess (s. Kapitel 6) ein Vorgehen definiert. Beispielsweise soll beim Vorhandensein von Build Dependencies geprüft werden, ob diese eigenhändig durch den Entwickler angepasst werden können, oder ob der Hersteller des Standard-Images das Image den erforderlichen Sicherheitsbedingungen anpasst. Der Image-Scan hat im Laufe des PoC bereits erste Images geprüft. Eine detaillierte Auswertungsmethodik muss jedoch noch erarbeitet werden, da der Image-Scan sehr sensibel ist, und nicht unbedingt für jede Meldung des Scanners ein Eingreifen erforderlich werden wird[21].

[20] https://www.cncf.io/

[21] Der Image-Scan wurde auf das Universal Base Image (https://www.redhat.com/en/blog/introducing-red-hat-universal-base-image) angewandt und hat über 100 Einträge ausgegeben.

5. Demo-Anwendungen

Ein Ziel des PoCs war es, die Möglichkeit des einheitlichen Betriebs von Anwendungen in unterschiedlichen Datenzentralen nachzuweisen. Zu diesem Zweck wurden einige Demo-Anwendungen ausgewählt, angepasst und schließlich bereitgestellt.

Zur Auswahl möglicher Demo-Anwendungen wurde unter den Teilnehmenden des PoCs zunächst eine Umfrage hinsichtlich bevorzugter Anwendungsfälle und Softwarelösungen durchgeführt. Daraufhin wurden verschiedene Kandidaten zusammengestellt und deren Nutzbarkeit im Rahmen des PoCs kurz bewertet. Einschlägige Bewertungskriterien waren hierbei die Verfügbarkeit der Anwendung (Bereitstellung über ein öffentliches Repository oder direkter Bezug aus einer teilnehmenden Datenzentrale), die Komplexität des Anwendungsfalls und der Mehrwert für den PoC. Außerdem wurde für jede Anwendung der Ressourcenbedarf ermittelt. Anschließend fand die Auswahl der Demoanwendungen durch die Mitglieder des AP 4 statt. Die folgenden Anwendungen wurden im PoC zum Ausrollen in den teilnehmenden Datenzentralen festgelegt:

- PostgreSQL[22] als Datenbank-Basis für verschiedene Anwendungen,
- Nextcloud[23] als Anwendung mit einem öffentlichen Image eines Herstellers und
- Terminfinder SH[24] als Anwendung mit einem öffentlichen Image einer teilnehmenden Datenzentrale.

Weitere Kandidaten, die im PoC nicht bereitgestellt worden sind, umfassen beispielsweise Grafana[25], Keycloak[26] und Jitsi[27].

5.1 Organisation der Demo-Anwendungen

Ähnlich der Ablage von Standard-Images (vgl. Kapitel 4.1) wurde auch für die Organisation der Images für die Demo-Anwendungen das Code Repository der OS-Plattform der ÖV Open CoDE genutzt und ein entsprechendes Repository angelegt[28]. Für jede Anwendung existiert ein Projekt. In diesem Projekt ist entweder das Image des Herstellers verlinkt (z.B. bei Nextcloud) oder direkt

[22] https://www.postgresql.org/

[23] https://nextcloud.com/

[24] Ein Beispiel-Deployment ist zu finden unter https://terminfinder.schleswig-holstein.de/#/home

[25] https://grafana.com/

[26] https://www.keycloak.org/

[27] https://jitsi.org/

[28] https://gitlab.o4oe.de/ig-bvc/demo-apps

abgelegt (z.B. beim Terminfinder SH – hier wurde das Image durch die entwickelnde Datenzentrale bereitgestellt). Für die Installation bzw. das Ausrollen der Images in den jeweiligen Container Clustern wurde eine entsprechende Kurzdokumentation erstellt.

Die OS-Plattform der ÖV Open CoDE bildete den Ausgangspunkt für das Deployment insofern, als dass es als zentrale Registry zur Verteilung von Images und Artefakten genutzt wurde. Die einzelnen Datenzentralen führten für die Applikations-Images individuell Schwachstellenscans aus. Die Ergebnisse dieser Scans wurden gemeinsam bspw. hinsichtlich der Kritikalität gefundener Schwachstellen diskutiert und bewertet, sowie im Code Repository der OS-Plattform der ÖV Open CoDE veröffentlicht[29].

5.2 Anpassung der Demo-Anwendungen

Da die Images der Demo-Anwendungen im Auslieferungszustand nicht ohne Weiteres unter den gesetzten Vorgaben in den Datenzentralen bereitgestellt werden konnten, mussten die Deploymentbeschreibungen auf die Datenzentralen angepasst werden. Im PoC wurden die Standards der IG BvC als Maßstab genutzt und die Anwendungen jeweils mit der entsprechenden Deploymentbeschreibung bereitgestellt.

Auch bei der Inbetriebnahme von Containern auf Basis der Images waren teilweise Nacharbeiten bzw. technische Anpassungen erforderlich. Dies hatte unterschiedliche Gründe: In einigen Fällen wurden gesetzte Richtlinien nicht eingehalten, in anderen Fällen wiesen die Images Schwachstellen auf, die ausgeglichen werden mussten. Die Anpassung der Demo-Anwendungen erfolgte im Rahmen des PoC über die bereitgestellten Projekt-Mechanismen (Gitlab) des Code Repositorys der OS-Plattform der ÖV Open CoDE.

Grundsätzlich hat sich gezeigt, dass für die Anwendungen weitere Maßnahmen zur Absicherung des Zugangs (insbesondere hinsichtlich eingehendem Netzverkehr) ergriffen werden müssen, da diese in der Regel nicht Bestandteil der Auslieferung sind. Dies war nicht Teil des ersten DVS PoC.

5.3 Erfahrungen aus der Installation der Demo-Anwendungen

Die Installation der Demo-Anwendungen in den Clustern erfolgte teilweise in gemeinsamen Sitzungen der Teilnehmenden. Dies hatte zum Ziel, einen möglichst hohen Wissenstransfer zu erreichen, und um die Teilnehmenden des PoCs in der selbstständigen Installation der Demo-Anwendungen zu befähigen.

[29] Die Ergebnisse der Schwachstellenscans für die Anwendung Nextcloud finden sich bspw. hier: https://gitlab.o4oe.de/ig-bvc/demo-apps/nextcloud/-/tree/main/scan-results/2020-10-19-trivy-scan-results

Übergreifend haben sich die bereitgestellten Cluster bei der Installation der Anwendungen als sehr gut geeignet erwiesen. Die CD-Pipeline, die in der UAG Technik & Betrieb entwickelt worden ist[30], eignete sich in den Anwendungsfällen sehr gut für die Bereitstellung der Anwendungen. Bei dem Deployment der Anwendungen hat sich gezeigt, dass die Voraussetzungen in den Datenzentralen zum Deployment von Lösungen noch weiter angeglichen und standardisiert werden müssen, um eine möglichst einfache Bereitstellung der Anwendungen zu gewährleisten. Perspektivisch sollten die Anwendungen bzw. das Deployment der Anwendungen noch weiter ausgebaut werden, um beispielsweise Szenarien zur Ausfallsicherheit abbilden zu können. Die Erfahrung im Rahmen des PoCs zeigte außerdem, dass ein Schwachstellenscan für die Images, beispielsweise wie in Kapitel 4.2 beschrieben, in der Verantwortung der Datenzentrale bzw. des Softwarebetreibers bleiben muss. Dies liegt zum einen daran, dass kaum Images ohne Schwachstellen verfügbar sind und neue Schwachstellen zu einem beliebigen Zeitpunkt auffällig werden können und zum anderen daran, dass jede Datenzentrale unterschiedliche Risikovoraussetzungen hat. Ein zentraler Schwachstellenscan wird unter diesen Voraussetzungen nicht als geeignet angesehen.

[30] Die Beschreibung der CD-Pipeline wird Teil eines Detailstandards der DVS sein und in dem Kontext veröffentlicht.

6. Ergebnisse

Übergreifend konnten im PoC die Grundzüge des einheitlichen Betriebs von Containerlösungen auf unterschiedlichen Plattformen nach einheitlichen Standards erarbeitet und deren praktische Umsetzbarkeit nachgewiesen werden. Damit wird ein Grundprinzip der DVS bestätigt; dies konstituiert einen wichtigen Meilenstein in der Implementierung der DVS in die Praxis.

Einige der bisherigen durch das Rahmenwerk der Zielarchitektur festgelegten Standards wurden im PoC auf ihre Umsetzbarkeit und Praktikabilität geprüft. Hierbei hat sich zum Beispiel die Festlegung auf Kubernetes als Basis der Vereinheitlichung der Systemumgebungen in den verschiedenen Datenzentralen als hilfreich erwiesen. Auch die Nutzung des Code-Repository der OS-Plattform der ÖV Open CoDE erwies sich als praktikabel. Weiterhin hat die Deployment-Pipeline nach den DVS-Standards funktioniert.

Im Rahmen des PoCs haben sich außerdem einige Mehrwerte der DVS konkret verstetigt. Der Verlauf des PoCs war geprägt durch die durchwegs sehr gute Zusammenarbeit der Datenzentralen, im Verlauf derer insbesondere ein neues Zusammenarbeitsmodell auf operativer Ebene entstanden ist. Auch der Wissenstransfer zwischen den IT-Dienstleistern wurde gut angenommen. Für das Wissensmanagement innerhalb der DVS wurde zudem ein einheitliches technisches Glossar und Wording entwickelt.

Die im Vorhinein definierten User Stories (s. Kapitel 1.3) wurden bis auf wenige Ausnahmen bei den nachgelagerten User Stories alle bearbeitet. Tabelle 4 gibt einen Überblick über die Ergebnisse zu den einzelnen User Stories und den weiteren Handlungsbedarf für die Zukunft.

User Story	Erreichtes Ergebnis	Zukünftiger Handlungsbedarf
Priorisierte User Stories		
Als **Softwarelieferant** möchte ich definierte Richtlinien für Software erhalten, um weitestgehend plattformunabhängig entwickeln und in verschiedenen Cloud-Standorten testen zu können.	Nachweis der Funktion	Weiterentwicklung bestehender Policies und Umsetzung weiterer Richtlinien als Code
Als **Softwarelieferant** möchte ich in Rechenzentren der ÖV einfach und ohne großen Anpassungsaufwand Softwarelösungen zum Testen ausrollen können.	Nachweis der Funktion	Weitere Standardisierung innerhalb der DVS vorantreiben

Als **Softwarebetreiber** möchte ich containerisierte Anwendungen in unterschiedlichen Cloud-Standorten betreiben, um Wiederverwendbarkeit zu erzielen.	Nachweis der Funktion	Ausbau und Betrachtung komplexerer Betriebsszenarien
Als **Softwarebetreiber** möchte ich einheitliche / gleichartige Container-Cluster bereitgestellt bekommen, um das Deployment von Anwendungen zu erleichtern.	Nachweis der Funktion	Kontinuierliche Weiterentwicklung, Erhöhung des Grads der Standardisierung
Als **Softwarebetreiber** möchte ich die Kompatibilität des Cloud-Standortes für meine benötigten Ressourcen überprüfen, um den einwandfreien Betrieb zu gewährleisten.	Nachweis der Funktion	Professionalisierung der aktuellen Lösung auf Basis von OS
Als **Plattformbetreiber** möchte ich ein einheitliches und gemeinsames Regelwerk für die Konfiguration von Container-Clustern auf Basis von OSS benutzen, um standardisierte Cloud-Services bereitstellen zu können.	Nachweis der Funktion	Kontinuierliche Weiterentwicklung, Erhöhung des Grads der Standardisierung
Beschreibung der User Stories nachgeordneter Priorität		
Als **Softwarebetreiber** möchte ich Standard-Images aus einer (zentralen) Container-Registry beziehen, um Anwendungen zu nutzen.	Grundlagen gelegt	Erstellung weiterer Standard-Images und weitere Projekte aus Standard-Images umsetzen
Als **Softwarebetreiber** möchte ich Zugriff auf die Container-Registry des Plattformbetreibers, um meine Anwendungen zu deployen.	Zugunsten der Nutzung der Registry des Code Repository der OS-Plattform der ÖV Open CoDE nicht erfolgt	Bereitstellung einer Container-Registry in den Cloudstandorten erarbeiten
Als **Softwarebetreiber** möchte ich einen Zugang seitens des Plattformbetreibers zur Verfügung gestellt bekommen, um Cloud-Services am Cloud-Standort nutzen zu können.	Nicht bearbeitet, da kein Fokus in den Tätigkeiten im Rahmen des PoCs	Umsetzung der Anforderungen der UAG Technik & Betrieb
Als **Softwarebetreiber** möchte ich Anforderungen (insb. notwendige Infrastruktur, Netztopologien) an den Cloud-Service des Plattformbetreibers definieren können, um den benötigten Zielzustand zu erhalten.	Prinzip funktioniert (Deployment von Demoanwendungen)	Formalisierung der Beschreibung des Zielzustands anhand eines DVS-Detailstandards

Als **Plattformbetreiber** möchte ich innerhalb der DVS einen Open Source-Software-Stack (z. B. Sovereign Cloud Stack, SCS) nutzen können, um Abhängigkeiten zu anderen Anbietern zu verringern.	Prinzip funktioniert, einige Teilnehmende nutzten vollständigen OS-Stack	Unterstützung bei Ausbau und Professionalisierung der OS-Lösungen, Betrachtung zur Nutzbarkeit des SCS

Tabelle 4: Erreichte Ergebnisse zu den User Stories

Aus den Ergebnissen des PoCs ergibt sich noch weiterer Handlungsbedarf für die weitere Umsetzung der DVS. Zum einen muss die gemeinsame Entwicklung von Lösungen auf Basis der DVS-Architektur ausgebaut werden. Hierfür sollten feste Ressourcenbereitstellungen durch die teilnehmenden Datenzentralen erfolgen, da mit einem kontinuierlichen Aufwand bei der Pilotierung der DVS zu rechnen ist. Weiterhin sollte die Verantwortung für DVS-Projekte auf möglichst viele Datenzentralen verteilt werden, um so einerseits einen breiten Wissenstransfer unter den (potenziellen) Teilnehmenden der DVS zu ermöglichen, und andererseits ein möglichst hohes Akzeptanzniveau für die eingesetzten Lösungen zu erreichen.

Weiterhin sind im Rahmen der Arbeit in den verschiedenen AP Prozesse erarbeitet und beschrieben worden, die in Zukunft in die Standards der DVS einfließen werden. Dazu gehören beispielsweise der Prozess zur Bereitstellung und Prüfung von Standard-Images mit den zugehörigen Unterprozessen. Dieser Prozess beschreibt, wie ein Standard-Image, dass durch den Image-Hersteller bereitgestellt wird, in der DVS geprüft und ggf. angepasst wird (s. Abbildung 6 für eine schematische Darstellung). Auch der Image-Scan-Prozess, der beschreibt, wie ein Image auf Sicherheit und Zuverlässigkeit geprüft wird (s. Kapitel 4.2) gehört zu den im PoC entwickelten Prozessen. Weitere Abläufe, die von der UAG Technik & Betrieb entwickelt worden sind, wie etwa hinsichtlich des Deployments bereitgestellter Anwendungen, wurden im Rahmen des PoC evaluiert und bei Bedarf weiterentwickelt.

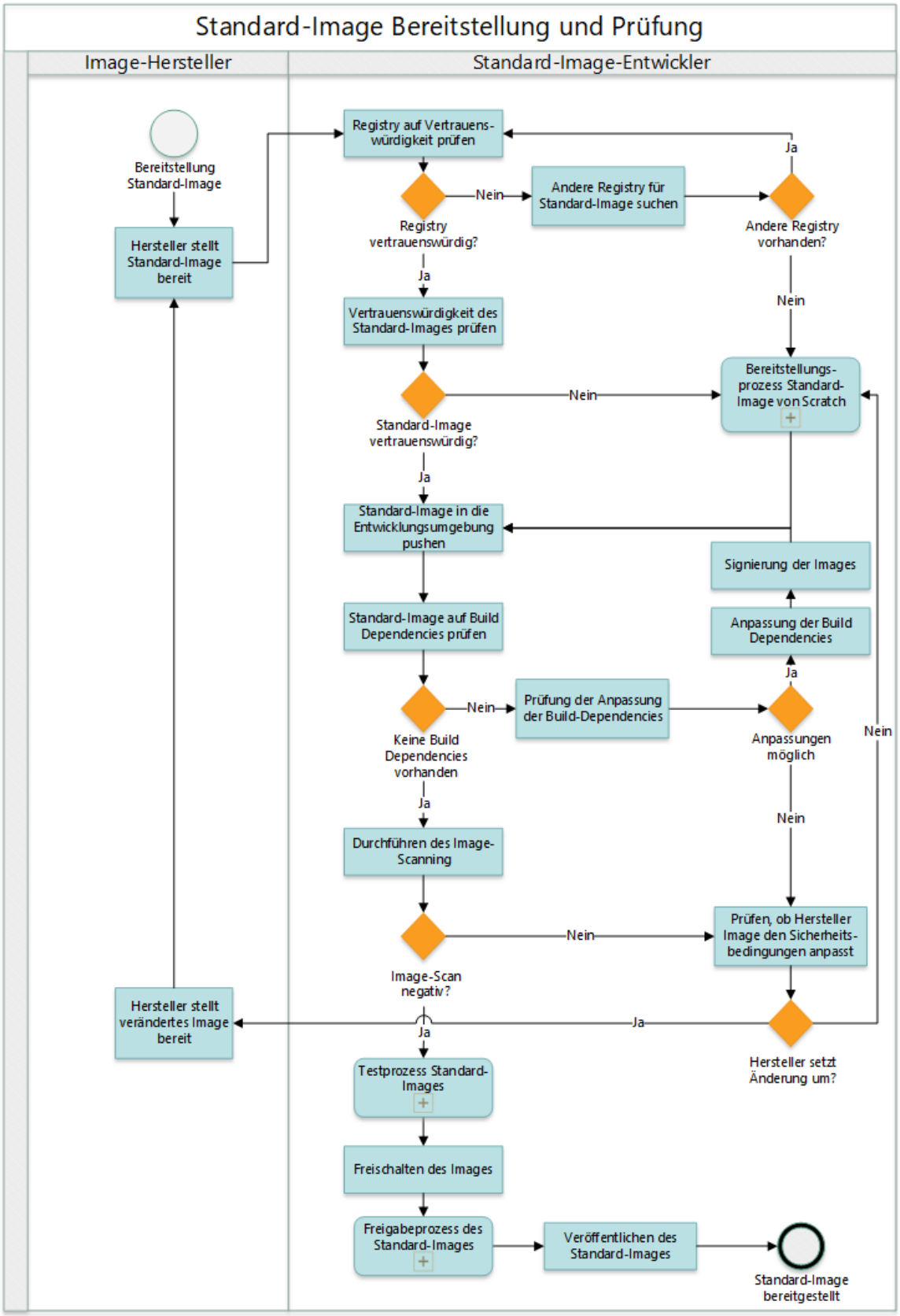

Abbildung 6: Prozess Bereitstellung und Prüfung von Standard-Images

7. Ausblick

Mit dem 1. DVS PoC wurde die erste Pilotierung der DVS erfolgreich abgeschlossen. Die hier beschriebenen Ergebnisse werden nun in Zusammenarbeit mit der IG BvC verstetigt und weiterentwickelt und fließen in die Konzeption der DVS mit ein. Um die kontinuierliche Weiterentwicklung der DVS voranzutreiben, sind für das Jahr 2022 bereits einige Ziele gesetzt, die im Rahmen der weiteren Pilotierung behandelt werden sollen.

Grundsätzlich sollen weitere potenzielle an der DVS teilnehmende IT-Dienstleister zur Teilnahme motiviert werden, sodass die Anzahl der DVS-Standorte mit kompatiblen Container-Clustern kontinuierlich ausgebaut wird. Das Deployment von Container-Clustern und Anwendungen nach einheitlichen Richtlinien, das bereits im PoC erzielt worden ist, ist auch in den perspektivisch neu teilnehmenden Datenzentralen vorzunehmen und zu testen.

Die durch den PoC aufgebaute, bestehende Entwicklungspartnerschaft zur Bereitstellung einheitlicher Container-Cluster soll im Jahr 2022 weiter ausgebaut und professionalisiert werden. Hierzu gehört insbesondere die Weiterführung des Richtlinienprojekts im Code Repository der OS-Plattform der ÖV Open CoDE (s. Kapitel 3) inkl. der Etablierung eines standardisierten Entwicklungsprozesses, die Initialisierung eines OS-Projekts zur Vereinheitlichung der Kompatibilitätstest und die Erstellung einer detaillierten Dokumentation hinsichtlich der Mitwirkung an der Richtlinienerstellung.

Außerdem sollen OS-Projekte zur Entwicklung und Verwaltung von Grund- und Runtime-Images ins Leben gerufen werden, die eine automatisierte Aktualisierung und Bereitstellung der Images umfassen. Hier soll insbesondere ein standardisierter Image-Lifecycle für die Images erarbeitet und ein Prozess für die Entwicklung der Images definiert werden. Hiermit sollen insbesondere die AP 3.3 und 5.2 aufgegriffen werden, die in diesem PoC nicht abschließend behandelt worden sind.

Neben den oben genannten Punkten besteht Handlungsbedarf zur weiteren Umsetzung der DVS etwa im Ausbau der Zusammenarbeit über die OS-Plattform der ÖV Open CoDE, bei den Standard-Images und bei der Umsetzung weiterer Entwicklungsprojekte. Im Bereich der Standard-Images steht beispielsweise die Weiterentwicklung und Härtung der Demo-Anwendungen aus dem PoC an (bspw. zur Absicherung des Zugangs zu den Anwendungen über einen einheitlichen Mechanismus), sowie deren Test bzw. Scan sowie die Umstellung der bisher genutzten Images auf Standard-Images. Zudem sollten weitere Standardlösungen, wie z.B. E-Government-Lösungen vom Hersteller Governikus, in das Anwendungsportfolio der DVS-Pilotierung aufgenommen werden. Die Umsetzung weiterer Entwicklungsprojekte umfasst z. B. die Entwicklung eines einheitliches

Loggings und Monitorings für Container-Cluster. Einen weiteren neuen Schritt in der Zusammenarbeit der Datenzentralen stellt die gemeinsame Entwicklung von Lösungen auf Basis von OS-Projekten dar.

8. Anhang

8.1 Abbildungsverzeichnis

8.2 Tabellenverzeichnis

8.3 Abkürzungsverzeichnis

Abkürzung	Bedeutung
AG Cloud	Arbeitsgruppe Cloud-Computing und Digitale Souveränität
AP	Arbeitspaket
BMI	Bundesministerium des Innern und für Heimat
BSI	Bundesamt für Sicherheit in der Informationstechnik
CD	Continuous Deployment; Continuous Delivery
CI	Continuous Integration
CPU	Central Processing Unit
CVE	Common Vulnerabilities and Exposures
DVS	Deutsche Verwaltungscloud; Deutsche Verwaltungscloud-Strategie
IaaS	Infrastructure-as-a-Service
IG BvC	Interessengruppe Betrieb von Containern
IT	Informationstechnologie
IT-PLR	IT-Planungsrat
OS	Open Source
OSS	Open Source-Software
ÖV	Öffentliche Verwaltung
PaaS	Platform-as-a-Service
PoC	Proof-of-Concept
RAM	Random-Access Memory
SaaS	Software-as-a-Service
UAG	Unterarbeitsgruppe
UAG Technik	Unterarbeitsgruppe Technik & Betrieb