IG Betrieb von Containern

Inhaltsverzeichnis

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.

[KK] Alt: Archiv SYS.1.6.A2 Anmerkung: Siehe ggf. auch BSI SYS.1.6.A5 Separierung der Container (B) , erweiterte Anforderungen ggü. der alten Maßnahme

Vor der Inbetriebnahme MUSS geplant werden, wie die in Containern betriebenen Anwendungen und deren unterschiedlichen Test- und Produktions-Betriebsumgebungen separiert werden. Auf Basis des Schutzbedarfs der Anwendungen, des Netzzonenkonzepts und einer Risikobetrachtung MUSS die Planung festlegen, welche Architektur angemessen auf die Risiken eingeht.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarebetreiber	muss	die in den Pods betriebenen Anwendungen auf Basis des Schutzbedarfs planen. Die Planung geht angemessen auf die Risiken ein und umfasst mindestens: • Separierung von Nichtproduktions- und Produktionsumgebungen in verschiedenen Clustern • Architektur der Namespaces, Meta-Tags, Cluster und Netze • Regelungen zur Separierung von Netz-, CPU- und Festspeicher-Ressourcen • Einsatz virtualisierter Server und Netze	Meta-Tags == Labels und Annotations?! Bezug zu Cluster und Netze?	X		P
Softwarebetreiber	muss	bei der Planung berücksichtigen, dass: • Anwendungen in jeweils eigenen Kubernetes-Namespaces betrieben werden • sich nur Anwendungen mit gleichem Schutzbedarf und ähnlichen möglichen Angriffsvektoren einen Kubernetes-Cluster teilen				
Softwarebetreiber	muss	Fachverfahren mit verschiedenen Schutzbedarfen in getrennten Clustern betreiben, wenn das Fachverfahren das für den Kubernetes-Cluster definierte Schutzniveau nicht dauerhaft garantieren kann.	Zielstellung: Umsetzung der verschiedenen Anforderungen aus den Schutzbedarfen. Werden verschiedene Anwendungen/Fachverfahren mit unterschiedlichen Schutzbarfen in einem gemeinsamen Kubernetes-Cluster betrieben, müssen diese Anwendungen/Fachverfahren das gleiche Schutzniveau erreichen. Zu berücksichtigen ist, dass sich das Schutznivau von Anwendungen/Fachverfahren im Vergleich zu anderen im zeitlichen Verlauf verändern kann (z.B. fehlende Fortentwicklung bzw. Sicherheitspatche, nicht mehr dem Stand der Technik entsprechende Sicherheitsmechanimen). (siehe SYS.1.6.A2 bei Plattformbetreiber)			

Softwarebetreiber muss	Fachverfahren mit verschiedenen Schutzniveaus in getrennten Clustern betreiben, wenn das Fachverfahren das für den Kubernetes-Cluster definierte Schutzniveau nicht erreichen kann. ist diese Differenzierung zum Punkt davor notwendig?	Zielstellung: Umsetzung der verschiedenen Anforderungen aus den Schutzbedarfen. Werden verschiedene Anwendungen/Fachverfahren mit unterschiedlichen Schutzbarfen in einem gemeinsamen Kubernetes-Cluster betrieben, müssen diese Anwendungen/Fachverfahren das gleiche Schutzniveau erreichen. Zu berücksichtigen ist, dass sich das Schutznivau von Anwendungen/Fachverfahren im Vergleich zu anderen im zeitlichen Verlauf verändern kann (z.B. fehlende Fortentwicklung bzw. Sicherheitspatche, nicht mehr dem Stand der Technik entsprechende Sicherheitsmechanimen). (siehe SYS.1.6.A2 bei Plattformbetreiber)		
Softwarebetreiber sollte	Fachverfahren mit sehr verschiedenen Schutzbedarfen oder Schutzniveaus ~~auf getrennten Workern und wenn möglich~~ auf getrennter Hardware bzw. getrennten Virtualisierungshosts betreiben.	Zielstellung: Umsetzung der verschiedenen Anforderungen aus den Schutzbedarfen und die Sicherheit der Plattform Beim Betrieb verschiedener Workloads muss geprüft werden, in wie weit sich diese Virtualisierungshosts teilen dürfen. Die Anforderungen aus den Bausteinen SYS.1.5 und NET.1.1 müssen berücksichtigt werden. (siehe SYS.1.6.A2 bei Plattformbetreiber)		
Softwarebetreiber muss	sicherstellen, dass • Workloads keine (System-)Namespaces mit dem Host teilen • Master-Nodes keine Nutzer-Workloads ausführen dürfen • eine hinreichende Isolation der Container durch geeignete Berechtigungen auf Ressourcen und Kernel-Funktionen gegeben ist	(siehe SYS.1.6.A5 bei Plattformbetreiber)		o
Softwarebetreiber sollte	in der Planungsphase folgende Aspekte mindestens berücksichtigen: • Einsatz einer Prozessüberwachung, die ungewollte Ausbrüche aus den Namespaces überwacht und im besten Fall verhindert • Namespaces für seine Anwendung auf die notwendigen Ressourcen (z.B. CPU, RAM, Netzwerkverbindungen, Storage usw.) beschränken • die Verteilung der Workloads auf die Cluster, Workernodes und Namespaces entsprechend Mandantenzuordnung und Schutzbedarf • Betrieb von Workloads unter clustereinzigartigen UIDs	(siehe SYS.1.6.A5 bei Plattformbetreiber)	X	
Softwarebetreiber muss	die Planungen dem Softwarelieferanten und dem Plattformbetreiber in geeigneter Dokumentation zur Verfügung stellen und deren Ergebnisse auf Erfüllung prüfen.	Hier müssen ggf. auch Lösungen gefunden werden, die über eine reine Rückmeldung hinaus gehen.		
Softwarelieferant muss	die vom Softwarebetreiber erstellten Planungen auf Erfüllung der geforderten Aspekte prüfen und das Ergebnis dem Softwarebetreiber in einer angemessenen Dokumentation zur Verfügung stellen.	Hier müssen ggf. auch Lösungen gefunden werden, die über eine reine Rückmeldung hinaus gehen.		

Plattformbetreiber muss	die vom Softwarebetreiber erstellten Planungen auf Erfüllung der geforderten Aspekte prüfen und das Ergebnis dem Softwarebetreiber in einer angemessenen Dokumentation zur Verfügung stellen.	Hier müssen ggf. auch Lösungen gefunden werden, die über eine reine Rückmeldung hinaus gehen.

APP.4.4.A2 Planung der Automatisierung mit CI-CD (B)

Wenn eine Automatisierung des Betriebs von Anwendungen in Kubernetes mithilfe von CI/CD stattfindet, DARF diese NUR nach einer geeigneten Planung erfolgen. Die Planung MUSS den gesamten Lebenszyklus von Inbetrieb- bis Außerbetriebnahme inklusive Entwicklung, Tests, Betrieb, Überwachung und Updates umfassen. Das Rollen- und Rechtekonzept sowie die Absicherung von Kubernetes Secrets MÜSSEN Teil der Planung sein.

[KK] Anmerkung: Spezifische Maßnahme zur Planung vom Einsatz von CI/CD. Ansatzweise sind Teile der Anforderungen in SYS.1.6.A3 (Alt) , SYS.1.6.A20 (Alt) und SYS.1.6.A22 (Alt)

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	das Rechte- und Rollenkonzept für die Verwaltung des Sourcecodes umsetzen.	siehe auch CON.8		x	---
Softwarelieferant	muss	Softwarelösungen mit unterschiedlichen Schutzbedarfen strikt trennen.	Hinweis: Gerade beim Test muss die strikte Trennung (Schutzbedarf und Mandant) umgesetzt werden, da hier Code ausgeführt wird. Governikus: Auf welcher Ebene muss die Trennung erfolgen? Unterschiedliche SourceCode-Repos, ...?		x	
Softwarelieferant	muss	Geheimnisse und Schlüssel verschlüsselt ablegen.		x	x	
Softwarebetreiber	sollte	ein Rechte- und Rollenkonzept für die Verwaltung des Sourcecodes vorgeben.		x		
Softwarebetreiber	muss	sicherstellen, dass die CI-Pipeline mit einem spezifischen technischen Benutzer mit minimalen Rechte auf die Sourcecode-Verwaltung zugreift.		x		
Softwarebetreiber	muss	das Rechte- und Rollenkonzept für die Automatisierungswerkzeuge CI-Pipeline definieren und umsetzen.	Hinweis: inkl. der Deployment-Registries	x		
Softwarebetreiber	muss	das Rechte- und Rollenkonzept für den Entwicklungscluster definieren und umsetzen.	Hinweis: Secrets dürfen nur für die nutzenden Namespaces zugreifbar sein.	x		
Softwarebetreiber	muss	die strikte Mandantentrennung für die Automatisierungswerkzeuge der CI-Pipeline sicherstellen.	Hinweis: inkl. der Deployment-Registries	x		
Softwarebetreiber	muss	Pipelines für Softwarelösungen mit unterschiedlichen Schutzbedarfen strikt trennen.	Hinweis: Gerade beim Test muss die strikte Trennung (Schutzbedarf und Mandant) umgesetzt werden, da hier Code ausgeführt wird.	x		
Softwarebetreiber	muss	für das Deployment in den Entwicklungscluster die Deployment-Registry des Plattformbetreibers nutzen.		x		
Softwarebetreiber	muss	die Deployment-Registries des Plattformbetreibers nutzen.		x		
Softwarebetreiber	muss	das Management für die Schwachstellen sicherstellen.	siehe SYS.1.6.A6	x		
Softwarebetreiber	muss	die Vorgaben zur Nutzung des Entwicklungsclusters vom Plattformbetreiber umsetzen.		x		
Softwarebetreiber	sollte	die Einstellungen der produktiven Cluster auf Entwicklungscluster anwenden.		x		
Softwarebetreiber	muss	Geheimnisse und Schlüssel verschlüsselt ablegen.		x		
Softwarebetreiber	muss	alle betriebenen Versionen in der Registry für die Produktion vorhalten.	Statische Deploymentbestandteile dürfen nur aus der Deploy- und Testregistry der Entwicklungsumgebung in die Produktionsregistry übernommen werden.			
Softwarebetreiber	muss	die Deploymentregistry der Produktion für das Deployment in den produktiven Cluster benutzen.		x		
Softwarebetreiber	muss	sicherstellen, dass die Deploymentbestandteile für die Produktion unveränderlich und versioniert abgelegt werden.		x		
Softwarebetreiber	muss	sicherstellen, dass alle Veränderungen am produktiven Deployment nachvollziehbar sind.		x		
Plattformbetreiber	sollte	ein Source-Code-Repository mit einer umfangreichen Rechteverwaltung bereitstellen. Zugriffsrechte auf die abgelegten Daten muss der Softwarebetreiber nutzen können.		x		
Plattformbetreiber	sollte	Automatisierungswerkzeuge für die CI-Pipeline mit einer umfangreichen Rechteverwaltung bereitstellen. Zugriffsrechte auf die Pipeline-Konfiguration muss der Softwarebetreiber nutzen können.		x		

Plattformbetreiber	muss	eine strikte Mandantentrennung für die Automatisierungswerkzeuge der CI-Pipeline anbieten.	Hinweis: inkl. der Deployment-Registries	x
Plattformbetreiber	muss	eine Registry für das Deployment in den Entwicklungscluster bereitstellen.	Hinweis: siehe Deployment-Registry, Die Deployment-Registry kann auch Test-Informationen für das automatisierte Testen beinhalten.	
Plattformbetreiber	muss	eine Registry für das Deployment in den Produktivcluster bereitstellen.		
Plattformbetreiber	muss	einen umfangreichen Schwachstellenscan auf den bereitgestellten Registries aktivieren und die Information der Softwarebetreiber sicherstellen.	Hinweis: der Scan muss beim Upload und kontinuierlich erfolgen siehe SYS.1.6.A6	
Plattformbetreiber	muss	einen Entwicklungscluster mit spezifschen Berechtigungen für den Softwarebetreiber bereitstellen.	Der Softwarebetreiber benötigt im Entwicklungs-Cluster erhöhte Zugriffsrechte, damit dieser seine Software testen und Analysewerkzeuge einsetzen kann. Der Plattformbetreiber muss die Zugriffsrechte so gestalten, dass die Plattform nicht in Gefahr gebracht wird.	
Plattformbetreiber	muss	die Mindestanforderungen für den Betrieb der Entwicklungscluster für den Softwarebetreiber vorgeben.		
Plattformbetreiber	muss	die etcd verschlüsseln.		
Plattformbetreiber	muss	sicherstellen, dass die Deploymentbestandteile für die Produktion unveränderlich und versioniert abgelegt werden können.		x
Plattformbetreiber	sollte	sicherstellen, dass Veränderungen in der produktiven Registry zentral protokolliert werden können.		

APP.4.4.A3 Identitäts- und Berechtigungsmanagement bei Kubernetes (B)

Kubernetes und alle anderen Anwendungen der Control Plane MÜSSEN jede Aktion eines Benutzers oder, im automatisierten Betrieb, einer entsprechenden Software authentifizieren und autorisieren, unabhängig davon, ob die Aktionen über einen Client, eine Weboberfläche oder über eine entsprechende Schnittstelle (API) erfolgt. Administrative Handlungen DÜRFEN NICHT anonym erfolgen.

Jeder Benutzer DARF NUR die unbedingt notwendigen Rechte erhalten. Berechtigungen ohne Einschränkungen MÜSSEN sehr restriktiv vergeben werden.

Nur ein kleiner Kreis von Personen SOLLTE berechtigt sein, Prozesse der Automatisierung zu definieren. Nur ausgewählte Administratoren SOLLTEN in Kubernetes das Recht erhalten, Freigaben für Festspeicher (Persistent Volumes) anzulegen oder zu ändern.

[KK]Alt: Archiv SYS.1.6.A24 Anmerkung: Erweiterte Anforderungen
Die Verwaltungssoftware SOLLTE jede Aktion eines Benutzers oder im automatisierten Betrieb einer entsprechenden Software authentifizieren und autorisieren, unabhängig davon, ob die Aktionen über eine Weboberfläche oder über eine API erfolgt. Aktionen SOLLTEN NICHT anonym erfolgen

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	darlegen, für welche Zwecke technische User (z.B.: Kubernetes ServiceAccounts) Berechtigungen erfordern.			x	P
Softwarebetreiber	muss	darlegen, für welche Zwecke technische User (z.B.: Kubernetes ServiceAccounts) Berechtigungen erfordern.		x		
Plattformbetreiber	muss	dokumentieren, für welche Zwecke technische User (z.B.: Kubernetes ServiceAccounts) Berechtigungen erfordern.				
Softwarebetreiber	muss	beim Plattformbetreiber benötigte Accounts und Berechtigungen anfordern und den Priviliegienrichtlinien entsprechen		x	x	
Softwarebetreiber	muss	gewährleisten, dass die eingeräumten Ausnahmen für (technische) User überprüft und dokumentiert sind und muss diese dem Plattformbetreiber zur Verfügung stellen.	Eine Umsetzung und Genehmigung obliegt aber weiterhin dem Plattformbetreiber.			
Softwarebetreiber	soll	soweit wie möglich und technisch realisierbar ist, die Zugriffe auf Zugangsdaten überwachen				
Softwarebetreiber	muss	anonyme Zugriffe für administrative Handlungen verhindern.		x	x	
Softwarebetreiber	soll	*sicherstellen, dass nur ein kleiner Kreis von Personen berechtigt ist, Prozesse der Automatisierung zu authorisieren.*				
Plattformbetreiber	muss	notwendige Accounts mit entsprechenden Berechtigungen für die Softwarebetreiber bereitstellen. (Least Privilige)				
Softwarebetreiber	muss	sicherstellen, dass jede Aktion eines Benutzers oder, im automatisierten Betrieb, einer entsprechenden Software authentifizieren und autorisieren, unabhängig davon, ob die Aktionen über einen Client, eine Weboberfläche oder über eine entsprechende Schnittstelle (API) erfolgt.				
Plattformbetreiber	muss	ein rollenbasiertes Identitäts- und Berechtigungsmanagement für die Plattform bereitstellen und dem Softwarebetreiber dessen Nutzung ermöglichen.	Das Rollen- und Berechtigungssystem sollte klar strukturiert und möglichst einfach gestaltet sein.	x		
Plattformbetreiber	muss	*sicherstellen, dass nur ein kleiner Kreis ausgewählter Administratoren in Kubernetes das Recht erhalten, Freigaben für Festspeicher (Persistent Volumes) anzulegen oder zu ändern*	Nur Cluster Administratoren besitzen dieses Recht.			

APP.4.4.A4 Separierung von Pods (B)

Der Betriebssystem-Kernel der Nodes MUSS über Isolationsmechanismen zur Beschränkung von Sichtbarkeit und Ressourcennutzung der Pods untereinander verfügen (vgl. Linux Namespaces und cgroups). Die Trennung MUSS dabei mindestens Prozess-IDs, Inter-Prozess-Kommunikation, Benutzer-IDs, Dateisystem und Netz inklusive Hostname umfassen.

[KK]Alt: Archiv SYS.1.6.A5 Anmerkung: Maßnahmen nahe zu identisch

Der Betriebssystem-Kernel MUSS über Namespaces (wie Linux cgroups) oder andere geeignete Mechanismen die Container voneinander und von anderen Prozessen auf dem Container-Host trennen. Die Trennung MUSS dabei mindestens Prozess-IDs, Inter-Process-Kommunikation, Benutzer-IDs, Dateisystem und Netz inklusive Hostname umfassen.

Wenn der Container-Dienst oder die Cluster-Betriebssoftware mehrere Container miteinander in einer Einheit betreibt, dürfen diese Einheiten sich einen Namespace teilen.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	bei der Entwicklung seiner Software-Produkte, die durch die Plattform vorgegebenen Isolationsmechanismen und die plattformspezifischen Isolationsmechanismen berücksichtigen.	- durch Plattform vorgegebene Isolationsmechanismen wie z.B. SE Linux und CGroups - plattformspezifische Isolationsmechanismen wie z.B. Namespaces, Pod Security Admission und Network Policies		x	P
Softwarelieferant	muss	sicherstellen, dass die vom Plattformbetreiber definierte Standard-Pod-Security und die Netzwerk-Security vom Softwarebetreiber eingehalten werden kann.	- Standard-Pod-Security (Pod Security Admission) - Netzwerk-Security (Network Policy) Governikus: Wünschenswert sind standardisierte Vorgaben. Es ist sonst zu befürchten, dass jeder Plattformbetreiber eigene Security-Anforderungen aufstellt und daher Plattformbetreiber-spezifischen Auslieferungen bereitgestellt werden müssen.		x	
Softwarebetreiber	muss	die vom Plattformbetreiber definierte Standard-Pod-Security und die Netzwerk-Security verwenden und diese gegebenenfalls durch eigene Standard-Pod-Security und Netzwerk-Security nach dem Least-Privilege-Prinzip erweitern.	- Standard-Pod-Security (Pod Security Admission) - Netzwerk-Security (Network Policy)	x		
Plattformbetreiber	muss	sicherstellen, dass der Betriebssystemkernel der Nodes Isolationsmechanismen unterstützt. Es muss eine Trennung auf Prozess-ID, Inter-Prozesskommunikation, Benutzer-ID, Dateisystem und Netzwerk möglich sein. Die genutzten Isolationsmechanismen müssen dokumentiert sein und dem Softwarelieferanten und dem Softwarebetreiber bei Bedarf zur Verfügung gestellt werden.	- Isolationsmechanismen wie z.B. System Namespaces, SE Linux und CGroups		x	
Plattformbetreiber	muss	Cluster bereitstellen, die die Verwendung der Isolationsmechanismen auf Basis von Role Based Access Control ermöglichen.	- wie z.B. Namespaces, Pod Security Admission und Network Policies			
Plattformbetreiber	sollte	die Verwendung von Isolationsmechanismen angepasst an den Schutzbedarf der jeweiligen Anwendung durchsetzen.				
Plattformbetreiber	muss	sicherstellen, dass Master-Nodes keine Nutzer-Workloads ausführen dürfen.	vgl. APP.4.4.A1			
Plattformbetreiber	sollte	eine Prozessüberwachung einsetzen, die Ausbrüche aus den Linux-Namespaces überwacht und im besten Fall verhindert.				

APP.4.4.A5 Datensicherung im Cluster (B)

Es MUSS eine Datensicherung des Clusters erfolgen. Die Datensicherung MUSS umfassen:

· Festspeicher (Persistent Volumes),

· Konfigurationsdateien von Kubernetes und den weiteren Programmen der Control Plane,

· den aktuellen Zustand des Kubernetes-Clusters inklusive der Erweiterungen,

· Datenbanken der Konfiguration, namentlich hier etcd,

· alle Infrastrukturanwendungen, die zum Betrieb des Clusters und der darin befindlichen Dienste notwendig sind und

· die Datenhaltung der Code und Image Registries.

Es SOLLTEN auch Snapshots für den Betrieb der Anwendungen betrachtet werden. Snapshots DÜRFEN die Datensicherung NICHT ersetzen.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	soll	Software-Produkte so entwickeln, dass zu jedem beliebigen Zeitpunkt eine konsistente Datensicherung seiner Anwendung durch den Software- oder Plattformbetreiber durchgeführt werden kann und eine Point in Time Wiederherstellung möglich ist.	*Bei einem Recovery ist der exakt gleiche Zustand des Clusters (gleiche Anzahl an Pods) nicht erreichbar.* Folgende Mechanismen sollten beachtet werden: • *Festspeicher (Persistent Volumes)* • Konfigurationsdateien von Kubernetes und den weiteren Programmen der Control Plane, • plattformspezifische Erweiterungen wie z.B. Monitoring, Logging, Ingress etc. • Datenbanken der Konfiguration, namentlich hier etcd, • alle Infrastrukturanwendungen die zum Betrieb des Clusters und der darin befindlichen Dienste notwendig sind • die Datenhaltung der Code und Image Registries		x	---
Softwarelieferant	muss	Software-Produkte so entwickeln, dass ein Re-Deployment auf einem "Restore-Cluster" jederzeit möglich ist	*siehe PoC: Standardisierung des Deployments und Sicherstellung der Deployment-Kompatibilität*		x	
Softwarebetreiber	muss	die Datensicherung so implementieren, dass die Anwendung über die CI/CD-Manifeste einschließlich Konfigurationsdateien, physischen Volumes und Datenbanken wiederhergestellt werden kann.				

Plattformbetreiber muss	Datensicherungsmechanismen einsetzen, die mit einem oder mehreren Kubernetes-Clustern interagieren und eine konsistente Datensicherung über: • Festspeicher (Persistent Volumes), • Konfigurationsdateien von Kubernetes und den weiteren Programmen der Control Plane, • plattformspezifische Erweiterungen wie z.B. Monitoring, Logging, Ingress etc. • Datenbanken der Konfiguration, namentlich hier etcd, • alle Infrastrukturanwendungen inkl. z.B. Jenkins, die zum Betrieb des Clusters und der darin befindlichen Dienste notwendig sind und • die Datenhaltung der Code und Image Registries ermöglichen und eine Point in Time Wiederherstellung anbieten.	
Plattformbetreiber kann	*Snapshots für die Sicherung der Anwendungen einsetzen.*	
Plattformbetreiber muss	sicherstellen, das die Datensicherung nicht über Snapshots realisiert wird.	

APP.4.4.A6 Initialisierung von Pods (S)

Sofern im Pod zum Start eine Initialisierung z.B. einer Anwendung erfolgt, SOLLTE diese in einem eigenen Init-Container stattfinden. Es SOLLTE sichergestellt sein, dass die Initialisierung alle bereits laufenden Prozesse beendet. Kubernetes SOLLTE NUR bei erfolgreicher Initialisierung die weiteren Container starten.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	den Abschluss eines Init-Container sicherstellen.			x	---
Softwarelieferant	muss	den Init-Container eindeutig von der Applikation logisch trennen.			x	
Softwarelieferant	muss	den Einsatz eines Init-Containers dokumentieren.	Es müssen die Berechtigungen dargestellt werden. Er muss die notwendigen Ressourcen dokumentieren und eine Empfehlung bereitstellen.		x	
Softwarelieferant	muss	die Ressourcenanforderungen an die Init-Container beschreiben und dem Softwarebetreiber mitteilen.			x	
Softwarebetreiber	muss	den Lebenzyklus des Init-Containers überwachen.				
Softwarebetreiber	muss	sicherstellen, dass ein Init-Container in seinen Ressourcen limitiert ist.	RAM, CPU siehe auch SYS.1.6.A15	x		
Plattformbetreiber	sollte	Werkzeuge zum Monitoren der Pods bereitstellen				

APP.4.4.A7 Separierung der Netze bei Kubernetes (S)

Die Netze für die Administration der Nodes, der Control Plane sowie die einzelnen Netze der Anwendungsdienste SOLLTEN separiert werden.

Es SOLLTEN NUR die für den Betrieb notwendigen Netzports der Pods in die dafür vorgesehenen Netze freigegeben werden. Bei mehreren Anwendungen auf einem Kubernetes-Cluster SOLLTEN zunächst alle Netzverbindungen zwischen den Kubernetes-Namespaces untersagt und nur benötigte Netzverbindungen gestattet sein (Whitelisting). Die zur Administration der Nodes, der Runtime und von Kubernetes inklusive seiner Erweiterungen notwendigen Netzports SOLLTEN NUR aus dem Administrationsnetz und von Pods, die diese benötigen, erreichbar sein.

Nur ausgewählte Administratoren SOLLTEN in Kubernetes berechtigt sein, das CNI zu verwalten und Regeln für das Netz anzulegen oder zu ändern.

Archiv: BSI SYS.1.6.A18 Absicherung der Wirk- und Administrations-Netze (S) Anmerkung: Die folgende Anforderung aus dem SYS.1.6 (neu) ist zum Teil identisch:

SYS.1.6.A5 Separierung der Administrations- und Zugangsnetze bei Containern (B)

Die Netze für die Administration der Hosts, die Administration des Container-Dienstes und die einzelnen Netze der Anwendungsdienste SOLLTEN separiert werden.

Es SOLLTEN NUR die für den Betrieb notwendigen Netzports der Container in die dafür vorgesehenen Produktivnetze freigegeben werden.

Die zur Administration der Container-Hosts, der Container-Dienste und der Cluster-Betriebssoftware notwendigen Netzports SOLLTEN NUR aus dem Administrationsnetz erreichbar sein, Ausnahme sind hier die Container der Cluster-Betriebssoftware inklusive der Container des Netz-Managements („CNI"), die per SSH, mit lokalen Agenten der Cluster-Betriebssoftware und der Datenhaltung der Cluster-Betriebssoftware oder dem Container-Host kommunizieren.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	MUSS	alle erforderlichen Kommunikationsverbindungen inklusive Ziele, Kommunikationsprotokolle und Ports beschreiben, die für Inbetriebnahme und Betrieb erforderlich sind.	evtl. Aufnahme einer Vorlage für die Kommunikationsbeziehungen. Bei klar definierten Zielen sollten FQDN oder IP-Adressen zur Beschreibung genutzt werden.		X	P
Softwarelieferant	KANN	in Abstimmung mit dem Softwarebetreiber die Kommunikationsbeziehungen in einer technisch formalen Form darstellen.	siehe https://kubernetes.io/docs/concepts/services-networking/network-policies		X	
Softwarebetreiber	MUSS	alle erforderlichen Kommunikationsverbindungen planen, dokumentieren und mit allen Beteiligten abstimmen.				
Softwarebetreiber	MUSS	die Kommunikation auf die erforderlichen Kommunikationsverbindungen inklusive Zielhosts, Kommunikationsprotokolle und Ports explizit freigeben (Whitelisting), die für Inbetriebnahme und Betrieb erforderlich sind. Nicht freigegebene Kommunikationsverbindungen MÜSSEN unterbunden bleiben.	z.B. Nutzung von Network-Namespaces, Netzwerk-Funktion mit hinreichender Isolations- und Filterwirkung bis mindestens OSI Layer 4 (z.B. Calico, openvSwitch, ACI, NSX-T). Alles was nicht ausdrücklich erlaubt ist, ist verboten.			
Softwarebetreiber	KANN	zur Beschreibung der Einschränkungen der Kommunikationsverbindungen Labels nutzen.				
Softwarebetreiber	SOLLTE	für jeden nach außen exponierten Service eine eigene IP-Adresse nutzen.	Verschiedene, nach außen exponierte Services sollten sich über die IP-Adressierung unterscheiden lassen, um Schutzmaßnahmen außerhalb der Kubernetes-Umgebung zu vereinfachen.			
Plattformbetreiber	MUSS	Netze für die Administration der Hosts, des Container-Dienstes und den Anwendungsnetzen logisch trennen, z. B: auf Basis VLANs.	Anforderung ist Bestandteil "Planung der Plattform". Diese Anforderung hat einen Bezug zu BSI IT-GS NET.1.1 Netzarchitektur und Design.			
Plattformbetreiber	SOLLTE	Netze für die Administration der Hosts, des Container-Dienstes und des Anwendungsnetzes trennen.	Diese Anforderung hat einen Bezug zu BSI IT-GS NET.1.1 Netzarchitektur und Design. Anmerkung: Siehe SYS.1.6.A5			
Plattformbetreiber	MUSS	die Kommunikation auf die erforderlichen Kommunikationsverbindungen inklusive Nodes, Kommunikationsprotokolle und Ports explizit freigeben (Whitelisting), die für Inbetriebnahme und Betrieb des Clusters und seiner Nodes erforderlich sind. Nicht freigegebene Kommunikationsverbindungen MÜSSEN unterbunden bleiben.				
Plattformbetreiber	SOLLTE	für jeden nach außen exponierten Service eine eigene IP-Adresse bereitstellen.	Verschiedene, nach außen exponierte Services sollten sich über die IP-Adressierung unterscheiden lassen, um Schutzmaßnahmen außerhalb der Kubernetes-Umgebung zu vereinfachen.			
Plattformbetreiber	MUSS	für die Kommunikation zwischen den Nodes des Kubernetes-Clusters sichere Tunnelprotokolle nutzen.	Der zwischen den Cluster-Nodes bestehende Netzwerkverkehr muss geeignet voneinander isoliert und ggf. auch verschlüsselt werden (z.B. VXLAN, NVGRE).			
Plattformbetreiber	MUSS	sicherstellen, dass CNI-Zugriffe und Konfigurationen nur von ausgewählten Administratoren verwaltet und/oder verändert werden können.				

APP.4.4.A8 Absicherung von Konfigurationsdateien bei Kubernetes (S)

Die Konfigurationsdateien des Kubernetes-Cluster, inklusive aller Erweiterungen und Anwendungen SOLLTEN versioniert und annotiert werden.

Zugangsrechte auf die Verwaltungssoftware der Konfigurationsdateien SOLLTEN minimal vergeben werden. Zugriffsrechte für lesenden und schreibenden Zugriff auf die Konfigurationsdateien der Control Plane SOLLTEN besonders sorgfältig vergeben und eingeschränkt sein.

Anmerkung: Teilweise Anforderungen aus BSI SYS.1.6.A4 Härtung des Host-Systems (B)

Anmerkung: Unter "*Verwaltungssoftware der Konfigurationsdateien*" vertsehen wir ein Versionierungssystem, zum Beispiel Software-Repository (Git).

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	SWL Whitepaper	per Policy prüfbar
Softwarelieferant	muss	die Konfigurationsdateien in einer Verwaltungssoftware mit einer definierten Verfügbarkeit und einem definierten Vertraulichkeitsniveau managen.	Siehe auch CON.8.A10 "Versionsverwaltung des Quellcodes". Die Zugangsverwaltungssoftware muss in Anlehnung an CON.8.A1 "Definition von Rollen und Zuständigkeiten" konfiguriert sein.		x	---
Softwarelieferant	muss	die regelmäßige Sicherung und die Wiederherstellung der Verwaltungssoftware und deren Inhalte sicherstellen.		x	x	
Softwarelieferant	muss	die Berechtigungen auf die Konfigurationsdateien restriktiv vergeben und bei der Festlegung der erforderlichen Accounts und deren Berechtigungen ein Rollen-basiertes Rechte-System (RBAC) nutzen. Hierbei ist mindestens zwischen Pull- und Push-Berechtigungen zu unterscheiden.			x	
Softwarelieferant	muss	die Nachvollziehbarkeit von Veränderungen sicherstellen.	insbesondere für den Softwarebetreiber		x	
Softwarelieferant	muss	die Konfigurationsdateien manipulationssicher für den Softwarebetreiber bereitstellen.	Diese kann z. B. durch Zugriff auf ein Git-Repository erfolgen.		x	
Softwarebetreiber	muss	die Konfigurationsdateien in einer Verwaltungssoftware mit einer definierten Verfügbarkeit und einem definierten Vertraulichkeitsniveau managen.	Siehe auch CON.8.A10 "Versionsverwaltung des Quellcodes". Die Zugangsverwaltungssoftware muss in Anlehnung an CON.8.A1 "Definition von Rollen und Zuständigkeiten" konfiguriert sein.			
Softwarebetreiber	muss	die regelmäßige Sicherung und die Möglichkeit der Wiederherstellung der Verwaltungssoftware und deren Inhalte sicherstellen.		x		
Softwarebetreiber	muss	die Berechtigungen auf die Konfigurationsdateien restriktiv vergeben und bei der Festlegung der erforderlichen Accounts und deren Berechtigungen ein Rollen-basiertes Rechte-System (RBAC) nutzen. Hierbei ist mindestens zwischen Pull- und Push-Berechtigungen zu unterscheiden.				
Softwarebetreiber	muss	die Nachvollziehbarkeit von Veränderungen sicherstellen.	insbesondere für den Plattformbetreiber			
Softwarebetreiber	muss	die Konfigurationsdateien manipulationssicher für den Plattformbetreiber bereitstellen.	Diese kann z. B. durch Zugriff auf ein Git-Repository erfolgen.			
Softwarebetreiber	sollte	entsprechende Vorgaben zur Verwaltung der Konfigurationsdateien für den Softwarelieferanten definieren.			x	
Softwarebetreiber	muss	die Verwaltung der Konfigurationsdateien strikt zwischen den Stages insbesondere Produktion und Entwicklung trennen.		x		
Plattformbetreiber	muss	sicherstellen, dass die Konfigurationsdateien der Plattform in einer Verwaltungssoftware mit einer definierten Verfügbarkeit und einem definierten Vertraulichkeitsniveau verfügbar sind.	Siehe auch CON.8.A10 "Versionsverwaltung des Quellcodes". Die Zugangsverwaltungssoftware muss in Anlehnung an CON.8.A1 "Definition von Rollen und Zuständigkeiten" konfiguriert sein.			
Plattformbetreiber	muss	die regelmäßige Sicherung und die Möglichkeit der Wiederherstellung der Verwaltungssoftware und deren Inhalte sicherstellen.		x		

Plattformbetreiber	muss	die Berechtigungen auf die Konfigurationsdateien restriktiv vergeben und bei der Festlegung der erforderlichen Accounts und deren Berechtigungen ein Rollen-basiertes Rechte-System (RBAC) nutzen.	
Plattformbetreiber	muss	die Nachvollziehbarkeit von Veränderungen sicherstellen.	
Plattformbetreiber	muss	die Verwaltung der Konfigurationsdateien strikt zwischen den Stages insbesondere Produktion und Entwicklung trennen.	x
Plattformbetreiber	muss	Konfigurationsdateien für die Containerplattform strikt von Konfigurationsdateien des Softwarebetreibers trennen.	
Plattformbetreiber	sollte	entsprechende Vorgaben zur Verwaltung der Konfigurationsdateien für den Softwarebetreiber definieren.	
Plattformbetreiber	sollte	eine Verwaltungssoftware zum Managen der Konfigurationsdateien für die verschiedenen Stages für die Softwarebetreiber anbieten.	

APP.4.4.A9 Nutzung von Kubernetes Service-Accounts (S)

Pods SOLLTEN NICHT den "default"-Service-Account nutzen. Dem "default"-Service-Account SOLLTEN keine Rechte eingeräumt werden. Pods für unterschiedliche Anwendungen SOLLTEN jeweils unter eigenen Service-Accounts laufen. Berechtigungen für die Service-Accounts der Pods der Anwendungen SOLLTEN auf die unbedingt notwendigen Rechte beschränkt werden.

Pods, die keinen Service-Account benötigen, SOLLTEN diesen nicht einsehen können und keinen Zugriff auf entsprechende Token haben.

Nur Pods der Control Plane und Pods, die diese unbedingt benötigen, SOLLTEN privilegierte Service- Accounts nutzen.
Programme der Automatisierung SOLLTEN jeweils eigene Token erhalten, auch wenn sie aufgrund ähnlicher Aufgaben einen gemeinsamen Service-Account nutzen.

Beispielhafte Umsetzung: https://gitlab.o4oe.de/ig-bvc/app/app.4.4.a9-serviceaccount

[pw]Alt: Archiv SYS.1.6.A25 Anmerkungen: Anforderungen wurden eher umgedreht formuliert. Zugriffsrechte wurden ausführlicher formuliert.

Container SOLLTEN jeweils eigene Service-Accounts nutzen, um miteinander und mit den Diensten der Cluster-Betriebssoftware authentifiziert zu kommunizieren, Gruppen von Containern können, wenn sie gleiche Aufgaben haben, einen gemeinsamen Service-Account nutzen. Berechtigungen für die Service-Accounts SOLLTEN nur minimal vergeben sein. Jeder Dienst, der einen Service-Account nutzt, SOLLTE ein eigenes Token erhalten.

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	MUSS	ein Manifest oder eine Definitionsdatei für Service-Accounts seiner Applikation sowohl auf Cluster-Ebene als auch auf Namespace-Ebene bereitstellen.		x	X	P
Softwarelieferant	SOLLTE	die Software so gestalten, dass Pods nicht den default-Service-Account benötigen.	Bei OpenShift sind die Rechte des "default" Service Account bereits maximal eingeschränkt.	x	X	
Softwarelieferant	MUSS	die Software so gestalten, dass für Pods verschiedener Anwendungen jeweils eigene Service-Accounts genutzt werden.		x	X	
Softwarelieferant	SOLLTE	die Software so gestalten, dass Credentials des Service-Accounts (Token) nur initial zum Verbindungsaufbau verwendet werden und danach mit temporären Tokens gearbeitet wird. Sollte ein Service-Account von mehreren gleichartigen Pods genutzt werden, sollte pro Pod ein individuelles Token erstellt werden.	Achtung: Überprüfung der Machbarkeit der Maßnahme hinsichtlich individueller Token für einen Service-Account muss erfolgen.	x	X	
Softwarelieferant	SOLLTE	die Software so gestalten, dass sie mit einer möglichst geringen Anzahl von Service-Accounts betrieben werden kann.	https://docs.google.com/document/d/1fftLBt3XjDzyYQisEKH3TZXL1QnT_cHIbBnFtW98UOs/edit#bookmark=id.1gf8i83		X	

Softwarelieferant	MUSS	die Software so entwickeln, dass die Service-Accounts mit den geringst möglichen Berechtigungen (Least-Privilage) betrieben werden können.		X		
Softwarebetreiber	MUSS	die Service-Accounts innerhalb der eigenen Namespaces mit den minimal erforderlichen Berechtigungen einrichten und deren Notwendigkeit prüfen.		x		
Softwarebetreiber	SOLLTE	Anforderungen des Softwarelieferanten zur Nutzung des default-Service-Accounts ablehnen.		x		o
Softwarebetreiber	MUSS	die Service-Accounts außerhalb der eigenen Namespaces mit den minimal erforderlichen Berechtigungen beim Plattformbetreiber beantragen.	Anmerkung: Anwendungen, die über mehrere Namespaces (z.B. getrennte DB-Namespaces) betrieben werden, können Service-Accounts außerhalb des eigenen Namespaces benötigen.	x		
Softwarebetreiber	MUSS	sicher stellen, dass Pods für unterschiedliche Verfahren/Anwendungen unter jeweils eigenen Service-Accounts betrieben werden.	Beispiel: serviceAccountName: build-robot automountServiceAccountToken: false siehe auch: https://kubernetes.io/docs/reference/access-authn-authz/service-accounts-admin/ https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/			o
Softwarebetreiber	MUSS	Berechtigungen für Service-Accounts so vergeben, dass Pods welche keinen Service-Account benötigen, diese nicht einsehen können und keinen Zugriff auf entsprechende Token haben.	https://kubernetes.io/docs/reference/access-authn-authz/rbac/#service-account-permissions			o
Plattformbetreiber	MUSS	privilegierte Service-Accounts auf begründete Anforderung des Softwarebetreibers nur an Pods der Control Plane und Pods, die diese unbedingt benötigen, vergeben.	Frage: Macht das der Softwarebetreiber oder der Plattformbetreiber, oder beide? -> beide gehen abgestimmt vor, dre Plattformbetreiber muss es umsetzen			
Plattformbetreiber	MUSS	die notwendigen Service-Accounts mit den minimal erforderlichen Berechtigungen zur Verfügung stellen und deren Notwendigkeit prüfen.				
Plattformbetreiber	MUSS	Service-Accounts sowohl auf Cluster-Ebene als auch auf Namespace-Ebene bereitstellen.	Die Serviceaccounts müssen mit den geringst möglichen Privilegien ausgestattet werden. Service-Accounts auf Cluster-Ebene dürfen nicht für die Kommunikation zwischen Namespaces genutzt werden.			

APP.4.4.A10 Absicherung von Prozessen der Automatisierung (S)

Alle Prozesse der Automatisierungssoftware, wie CI/CD und deren Pipelines, SOLLTEN nur mit unbedingt notwendigen Rechten arbeiten. Wenn unterschiedliche Benutzergruppen über die Automatisierungssoftware die Konfiguration verändern oder Pods starten können, SOLLTE dies für jede Gruppe durch eigene Prozesse durchgeführt werden, die nur die für die jeweilige Benutzergruppe notwendigen Rechte besitzen.

[pw]Alt:Archiv: SYS.1.6.A22 Anmerkungen: Vorher war von Hilfsprozessen die Rede. Ansonsten einige Umformulierungen, ggfs. noch in Archiv: SYS.1.6.A20 schauen

Absicherung von Hilfsprozessen der Automatisierung

Alle Prozesse der Automatisierungssoftware SOLLTEN nur mit minimalen Rechten arbeiten. Wenn unterschiedliche Benutzergruppen über die Automatisierungssoftware die Konfiguration verändern können, SOLLTE dies für jede Gruppe durch eigene Prozesse durchgeführt werden, die nur die für die jeweilige Benutzergruppe notwendigen Rechte besitzen.
Anmerkung: Es existiert eine teilweise Überlappung mit A20 im Bereich der Absicherung von Automatisierungsprozessen.
Grundsätzlich wird bei den nachfolgenden Anforderungen darauf abgestellt, dass die Automatisierungssoftware nur über dedizierte Accounts auf die Containerumgebung bzw. deren API zugreifen darf.

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	darauf achten, dass der Build-Lauf für die erstellte Software nur mit den unbedingt notwendigen Rechten erfolgen kann.	Ist dann primär relevant, wenn der Softwareentwickler bzw. der Softwareentwicklungsprozess als Teil des Gesamtsystems/der Gesamtinfrastruktur betrachtet wird. Eventuell sind vertragliche Regelungen inkl. Auditierung erforderlich.	x	x	P
Softwarelieferant	muss	ein geeignetes Rollen- und Berechtigungsmodell definieren.		x	x	
Softwarelieferant	muss	die Software so bereitstellen, dass diese automatisiert deployed werden kann. Dieses muss nur mit unbedingt notwendigen Rechten umgesetzt werden.		x	x	
Softwarelieferant	muss	den Zugriff der Automatisierungssoftware ausschließlich über einen dediziert für den vorgesehenen Zweck bereitgestellten Account realisieren. Die bereitgestellten Accounts dürfen nur die Rechte erhalten, die unbedingt für seine Aufgabenstellung erforderlich sind.	Die Hoheit über das Deployment muss beim Softwarebetreiber liegen. Es müssen unterschiedliche Pipelines für das Deployment in DEV / Test und PROD eingesetzt werden.	x	x	
Softwarelieferant	sollte	die Auditierfähigkeit der Umgebung sicherstellen.		x	x	
Softwarebetreiber	muss	die Hoheit über das Deployment der Softwarelösung erhalten.	Alle Prozesse zum Deployment müssen durch den Softwarebetreiber initiiert werden.	x		
Softwarebetreiber	muss	dem Softwarelieferanten einen für den Zweck der Steuerung der Automatisierungssoftware geeigneten Account in der produktiven Umgebung verweigern.	Zielsetzung: Der Zugriff des Softwarelieferanten auf die produktive Umgebung muss verhindert werden.	x	x	
Softwarebetreiber	muss	ein geeignetes Rollen- und Berechtigungsmodell definieren.		x		
Softwarebetreiber	muss	im Bedarfsfall dem Softwarelieferanten für den jeweiligen Zweck geeignete (nur notwendige Berechtigungen[RBAC]) Accounts für die Automatisierungssoftware bereitstellen.		x	x	
Softwarebetreiber	sollte	die Auditierfähigkeit der Umgebung sicherstellen.		x		
Plattformbetreiber	muss	ein geeignetes Rollen- und Berechtigungsmodell definieren.		x		
Plattformbetreiber	muss	dem Softwarebetreiber für den jeweiligen Zweck geeignete (nur notwendige Berechtigungen[RBAC]) Accounts für die Automatisierungssoftware bereitstellen.		x		
Plattformbetreiber	muss	dafür sorgen, dass Software für Automatisierung der Plattform-Konfiguration nur mit unbedingt notwendigen Rechten betrieben wird.		x		
Plattformbetreiber	sollte	die Auditierfähigkeit der Umgebung sicherstellen.		x		

APP.4.4.A11 Überwachung der Container (S)

In Pods SOLLTE jeder Container einen Health Check für den Start und den Betrieb („readiness" und „liveness") definieren. Diese Checks SOLLTEN Auskunft über die Verfügbarkeit der im Pod ausgeführten Software geben. Die Checks SOLLTEN fehlschlagen, wenn die überwachte Software ihre Aufgaben nicht ordnungsgemäß wahrnehmen kann. Für jede dieser Kontrollen SOLLTE eine dem im Pod betriebenen Dienst angemessene Zeitspanne definieren.

Auf Basis dieser Checks SOLLTE Kubernetes die Pods löschen oder neu starten.

[pw]Alt:Archiv SYS.1.6.A27 Anmerkungen: Begriffe wurden ersetzt, Anforderungen konkretisiert und letzter Satz ist neu bzw. letzter Satz bei (alt) ist weggefallen.

Jedes Image SOLLTE einen Health-Check für den Start und den Betrieb („readiness" und „liveness") definieren. Diese Checks SOLLTEN Auskunft über die Verfügbarkeit der Anwendung im Container geben. Sie SOLLTEN fehlschlagen, wenn die Anwendung nicht in der Lage ist, ihre Aufgaben ordnungsgemäß wahrzunehmen.

Der Container-Dienst oder die Cluster-Betriebssoftware SOLLTEN diese Checks überwachen und Container, bei denen die Checks fehlschlagen, beenden und durch neue Instanzen ersetzen.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	Per Policy prüfbar
Softwarelieferant	sollte	einen Health-Check für den Start und den Betrieb („readiness" und „liveness"), ggf. nach den Vorgaben des Plattformbetreibers, definieren. Beide Checks müssen für die Anwendung relevante Funktionen prüfen und als Ergebnis zurückliefern.	Ausnahme: Init-Container: https://kubernetes.io/docs/concepts/workloads/pods/init-containers/	*	*	P
Softwarelieferant	kann	Start-up-Checks für den Start der Container implementieren.	https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/#define-startup-probes	*	*	o
Softwarebetreiber	sollte	Vorgaben für "readiness-", "liveness-" und "startup"-Checks mit dem Softwarelieferanten und dem Plattformbetreiber abstimmen.	"readiness-", "liveness-" und "startup"-Checks sind nativer Bestandteil von Kubernetes			
Softwarebetreiber	muss	die vom Softwarelieferant beschriebenen "readiness-", "liveness-" und "startup"-Checks in der Konfiguration umsetzen.		*		012

APP.4.4.A12 Absicherung der Infrastruktur-Anwendungen (S)

Sofern eine eigene Registry für Images oder eine Software zur Automatisierung, zur Verwaltung des Festspeichers, zur Speicherung von Konfigurationsdateien oder ähnliches im Einsatz ist, SOLLTE deren Absicherung mindestens betrachten:

· Verwendung von personenbezogenen und Service-Accounts für den Zugang,

· verschlüsselte Kommunikation auf allen Netzports,

· minimale Vergabe der Berechtigungen an Benutzer und Service Accounts,

· Protokollierung der Veränderungen und

· regelmäßige Datensicherung.

[pw]Alt: Archiv SYS.1.6.A28 Anmerkungen: Verschlüsselte Kommunikation zusätzlich gefordert, ansonsten noch einige kleine Umformulierungen/Konkretisierungen, auch im ersten Satz

Sofern eine eigene Registry für Images eingesetzt wird, SOLLTE diese ausreichend abgesichert sein.

Dabei SOLLTEN beachtet werden:

- Verwendung von personenbezogenen und Service-Accounts für den Zugang,
- minimale Vergabe der Berechtigungen,
- Anbindung der Software für die Überwachung der Images auf Verwundbarkeiten,
- Protokollierung der Veränderungen der Images und
- die Datensicherung.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant		keine Anforderungen erkennbar				p
Softwarebetreiber	muss	dem Plattformbetreiber die von ihm erforderlichen Accounts und deren Berechtigungen auf benötigten Infrastruktur-Anwendungen mitteilen. Hierbei sind nur die zwingend erforderlichen Berechtigungen anzufordern (Least-Privilege).		x		
Plattformbetreiber	muss	personenbezogene Accounts und Service-Accounts mit minimalen Rechten bereitstellen (Least-Privilege). Hierbei ist mindestens zwischen Lese- und Schreibberechtigungen zu unterscheiden.		x		
Plattformbetreiber	muss	für die Bereitstellung von personenbezogenen Accounts und Service-Accounts ein Rollen-basiertes Zugriffsmanagement (RBAC) bereitstellen.				
Plattformbetreiber	sollte	zum Zugriffsmanagement einen zentralen Verzeichnisdienst nutzen.		x		
Plattformbetreiber	muss	bei einer Registry jede Bereitstellung und Änderung von Images überwachen und protokollieren.		x		
Plattformbetreiber	muss	sicherstellen, dass eine bestehende Versionierung der Lieferung nicht verändert werden kann.		x		
Plattformbetreiber	muss	jede Änderung an der Infrastruktur-Anwendung überwachen und protokollieren.		x		
Plattformbetreiber	muss	eine Registry nutzen, die eine standardisierte Schnittstelle zur Einbindung verschiedener Scanner zur Erkennung von Schwachstellen und Schadcode bereitstellt.		x		
Plattformbetreiber	muss	die regelmäßige Sicherung und die Möglichkeit der Wiederherstellung der Infrastruktur-Anwendungen und deren Inhalte sicherstellen.				
Plattformbetreiber	muss	die Infrastruktur-Anwendungen mit einer definierten Verfügbarkeit und einem definierten Vertraulichkeitsniveau bereitstellen.				p?
Plattformbetreiber	kann	eine Replikation von Images in andere vertrauenswürdige Registries ermöglichen.				
Plattformbetreiber	sollte	die verschlüsselte Kommunikation auf allen Netzports sicherstellen.				p

APP.4.4.A13 Automatisierte Auditierung der Konfiguration (H)

Es SOLLTE ein automatisches Audit der Einstellungen der Nodes, von Kubernetes und der Pods der Anwendungen gegen eine definierte Liste der erlaubten Einstellungen und gegen standardisierte Benchmarks erfolgen.

Kubernetes SOLLTE die aufgestellten Regeln im Cluster durch Anbindung geeigneter Werkzeuge durchsetzen.

~~*-> Entsprechung in SYS.1.6.A29 Automatisierte Auditierung von Containern (H)*~~

~~Die gesamte Software in den Images SOLLTE automatisiert katalogisiert werden. Sie SOLLTE mindestens täglich mit aktualisierten Datenbanken über bekannte Verwundbarkeiten abgeglichen werden. Auch die Einstellungen des Containers selbst sowie die des betriebenen Anwendungsdienstes SOLLTEN automatisiert mit einer Liste der erlaubten Einstellungen abgeglichen werden. Nur Container, die geeignet überprüft wurden, SOLLTEN für den Einsatz im Produktivbetrieb freigegeben werden. Die Orchestrierung SOLLTE diese Prozesse automatisieren.~~

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	die Deployments entsprechend der Richtlinien des Softwarebetreibers ausrichten.	Governikus: Hier ist eine Standardisierung der Richtlinien wünschenswert. Es ist sonst zu befürchten, dass jeder Softwarebetreiber eigene Richtlinien aufstellt und daher Softwarebetreiber-spezifischen Deployments bereitgestellt werden müssen.		x	---
Softwarebetreiber	muss	Richtlinien für die Einstellungen von Deployments definieren und dem Softwarelieferanten bekannt geben.			x	
Softwarebetreiber	muss	Konfiguration der Deployments gegen ein definiertes Regelwerk prüfen und durchsetzen.	z.B. Kyverno oder OPA für die Prüfung der Deployments	x		
Softwarebetreiber	muss	die durch ihn betriebenen Workloads laufend auf Richtlinienverstöße überwachen, melden und ggf. unterbinden.	z.B. falco (Aqua-Security) Die Festlegung einer laufenden Überwachung, Meldung und Einschränkung von Regelverstößen impliziert ein hinreichendes Maß an Automatisierung.			
Plattformbetreiber	muss	Richtlinien für die Einstellungen von Nodes und Kubernets-Cluster definieren und dem Softwarebetreiber bekannt geben.				
Plattformbetreiber	muss	seine Plattform mittels Benchmarks in regelmäßigen Abständen, nach Änderungen an der Plattform und situationsabhängig gegen ein definiertes Regelwerk prüfen.	z.B. Kube-Bench mit CIS-Benchmarks			
Plattformbetreiber	muss	die auf seiner Plattform betriebenen Workloads laufend auf Richtlinienverstöße überwachen, melden und ggf. unterbinden. 07.09.2022: sollte in der großen Runde diskutiert werden.... Redundanz mit Softwarebetreiber	z.B. falco (Aqua-Security) Die Festlegung einer laufenden Überwachung, Meldung und Einschränkung von Regelverstößen impliziert ein hinreichendes Maß an Automatisierung.			

APP.4.4.A14 Verwendung dedizierter Nodes (H)

In einem Kubernetes-Cluster SOLLTEN die Nodes dedizierte Aufgaben zugewiesen bekommen und jeweils nur Pods betreiben, welche der jeweiligen Aufgabe zugeordnet sind.

Bastion Nodes SOLLTEN alle ein- und ausgehenden Datenverbindungen der Anwendungen zu anderen Netzen übernehmen.

Management Nodes SOLLTEN die Pods der Control Plane betreiben und sie SOLLTEN nur die Datenverbindungen der Control Plane übernehmen.

Sofern eingesetzt, SOLLTEN Speicher-Nodes nur die Pods der Festspeicherdienste im Cluster betreiben.

-> Entsprechung in SYS.1.6.A19Verwendung vorgelagerter Ein- und Ausgangssysteme (S)

Sofern mehrere Container-Hosts in einem Verbund arbeiten („Cluster"), SOLLTEN dedizierte Container-Hosts die Ein- und Ausgabe zu anderen Netzen übernehmen. Die anderen Container-Hosts SOLLTEN NICHT aus anderen Netzen außer dem Administrationsnetz erreichbar sein.

Annahmen:

- Eingabe meint Ingress, also eingehenden Netzwerktraffic.
- Ausgabe meint Egress, also ausgehenden Netzwerktraffic.

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	die eingehenden und ausgehenden Kommnunikationsbeziehungen beschreiben und für den Softwarebetreiber dokumentieren.	Auch die Kommunikation innerhalb der Anwendung muss bekannt und beschrieben sein.		x	P (Infra-Nodes und dedizierte Ingress)
Softwarelieferant	muss	die eingehende und ausgehende Kommunikation über Kubernetes-Serviceressourcen umsetzen.			x	
Softwarelieferant	muss	das vom Softwarebetreiber benötigte Verteilungsschema für die Pods unterstützen.	z. B. Aufteilung der Pods entsprechend der Aufgaben		x	
Softwarebetreiber	muss	das Verteilungsschema für die Pods definieren. Die Nodes müssen mindestens nach folgenden Aufgaben unterschieden und getrennt werden: - Bastion-Nodes zur Realisierung von ingress und egress - Anwendungs-Nodes zum Betrieb der Pods für die Anwendungen - Speicher-Nodes zur Bereitstellung der Speicherlösungen - Management-Nodes für den Cluster		x	x	
Softwarebetreiber	muss	durch entsprechende Konfiguration dedizierte Container-Hosts eines Clusters ausschließlich für die notwendige Ein- und Ausgabe von Daten/Kommunikation zu anderen Netzen nutzen.	Bastion-Nodes			
Softwarebetreiber	muss	separate Nodes für den Betrieb von Speicherlösungen planen.	z. B. DBMS oder S3-Storage			
Softwarebetreiber	sollte	separate Nodes für verschiedene Speicherlösungen planen.				
Plattformbetreiber	muss	muss die Nodes entsprechend dem Verteilschema des Softwarebetreibers bereitstellen.				
Plattformbetreiber	kann	zusätzlich externe Systeme (z.B. Load-Balancer) für die Ein- und Ausgabe von Daten zu anderen Netzen festlegen und bereitstellen.	Primärer Anwendungsfall ist die Schaffung der Hochverfügbarkeit über mehrere ingress-Controller und Clustergrenzen hinweg.			
Plattformbetreiber	kann	für Ein- und Ausgaben zusätzliche dedizierte Container-Hosts bereitstellen, wenn dies besondere Protokolle oder Kommunikationsbeziehungen erfordern.	Die Bereitstellung dedizierter Hosts für bestimmte Außenkommunikation kann erforderlich werden, wenn die Kommunikationsverbindung einer besonderen Isolation unterliegen soll. Hierbei erfolgt nur Trennung der Kommunikation in an deren Netze. Eine erforderliche Trennung entsprechend einem Zonen-Modell bleibt hiervon unberührt. Es darf durch einen Kubernetes-Cluster keine Überbrückung unterschiedlicher Sicherheitszonen erfolgen.			
Plattformbetreiber	muss	Pods der Festspeicherdienste im Cluster ausschließlich auf Speicher-Nodes betreiben.	Annahme: Speicher-Nodes sind spezielle Worker Nodes, die Speicher als Persistenten Storage zur Verfügung stellen.			
Plattformbetreiber	muss	die Nutzung der Mangagement Nodes für den Betrieb von Anwendungsworkloads unterbinden.	Damit werden die Datenverbindungen der Control Plane automatisch von den Worker Nodes getrennt.			
Plattformbetreiber	muss	dem Softwarebetreiber einen verbindlichen Servicekatalog mit den Plattformfunktionen anbieten.	Die Zielstellung ist die konkrete Bereitstellung und Beschreibung der Möglichkeiten der Plattform.			

Plattformbetreiber sollte	dem Softwarebetreiber Vorgaben über die dedizierte Konfiguration (z.B. Definition von Services, Vorgaben zu Routing, Ingress- und Egress-Kommunikationen) machen.	
Softwarebetreiber muss	dem Softwarelieferanten Vorgaben zu den Kommunikationsverbindungen machen und hierbei die Anforderungen des Plattformbetreibers berücksichtigen.	x
Softwarelieferant muss	die Vorgaben des Softwarebetreibers zu den Kommunikationsverbindungen bei der Erstellung und Bereitstellung der Software berücksichtigen.	x

APP.4.4.A15 Trennung von Anwendungen auf Node- und Cluster-Ebene (H)

Anwendungen mit einem sehr hohen Schutzbedarf SOLLTEN jeweils eigene Kubernetes-Cluster oder dedizierte Nodes nutzen, die nicht für andere Anwendungen bereitstehen.

~~-> keine ENtsprechung in SYS 1.6 alt~~

siehe auch SYS.1.6.A26 Weitergehende Isolation und Kapselung von Containern (H)

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	sollte		Kein ToDo für den Softwarelieferanten			---
Softwarebetreiber	sollte	Anwendungen mit einem sehr hohen Schutzbedarf auf jeweils dedizierten und speziell gehärteten Kubernetes-Clustern betreiben.		x		
Softwarebetreiber	muss	für den Fall, dass kein dedizierter Cluster betrieben wird, Anwendungen mit einem sehr hohen Schutzbedarf auf jeweils dedizierten Worker-Nodes betreiben, wenn sich verschiedene Anwendungen einen Kubernetes-Clustern teilen.		x		
Plattformbetreiber	sollte	für den Betrieb von Anwendungen mit einem sehr hohen Schutzbedarf dedizierte Kubernetes-Cluster bereitstellen.				
Plattformbetreiber	muss	für den Betrieb von Anwendungen mit einem sehr hohen Schutzbedarf dedizierte Nodes bereitstellen, die von keinen anderen Anwendungen genutzt werden.				

APP.4.4.A16 Verwendung von Operatoren (H)

Die Automatisierung von Betriebsaufgaben in Operatoren SOLLTE bei besonders kritischen Anwendungen und den Programmen der Control Plane zum Einsatz kommen.

-> keine Entsprechung in SYS 1.6 alt

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	sollte	bei besonders kritischen Anwendungen Operatoren zur Automatisierung von Betriebsaufgaben liefern.	ggf. nicht nur für kritische Anwendungen: z.B. einfachere Verwaltung von Upgrades	x	x	---
Softwarebetreiber	sollte	den Einsatz von Operatoren unterstützen		x		
Plattformbetreiber	sollte	Operatoren zur Automatisierung in Programmen der Control Plane einsetzen		x		

APP.4.4.A17 Attestierung von Nodes (H)

Nodes SOLLTEN eine kryptografisch und möglichst mit einem TPM verifizierte gesicherte Zustandsmeldung an die Control Plane senden. Die Control Plane SOLLTE NUR Nodes in den Cluster aufnehmen, die erfolgreich ihre Unversehrtheit nachweisen konnten.

(kh) Neue Anforderung

Anmerkung: Keine Referenz gefunden. Die Anforderung ist sehr K8S Cluster spezifisch. Unklar wie ein Node seine Unversehrtheit gegenüber der Management Plane nachweisen soll bzw. was Unversehrtheit in dem Kontext ist.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	sollte		keine Anforderung für den Softwarelieferanten erkennbar			---
Softwarebetreiber	muss		keine Anforderung für den Softwarebetreiber erkennbar			
Plattformbetreiber	sollte	nur Nodes nur mit einem TPM-Chip einsetzen, welche eine *verifizierte* gesicherte Zustandsmeldung an die Control Plane senden können.				
Plattformbetreiber	muss	geeignete Werkzeuge zur Überprüfung der Nodes auf Unversehrtheit zur Verfügung stellen.	Thema für die große Runde			

APP.4.4.A18 Verwendung von Mikro-Segmentierung (H)

Die Pods SOLLTEN auch innerhalb eines Kubernetes-Namespace nur über die notwendigen Netzports miteinander kommunizieren können. Es SOLLTEN Regeln innerhalb des CNI existieren, die alle bis auf die für den Betrieb notwendigen Netzverbindungen innerhalb des Kubernetes-Namespace unterbinden. Diese Regeln SOLLTEN Quelle und Ziel der Verbindungen genau definieren und dafür mindestens eines der folgenden Kriterien nutzen: Service-Name, Metadaten („Labels"), die Kubernetes Service Accounts oder zertifikatsbasierte Authentifizierung.

Alle Kriterien, die als Bezeichnung für diese Verbindung dienen, SOLLTEN so abgesichert sein, dass sie nur von berechtigten Personen und Verwaltungs-Diensten verändert werden können.

(kh) Alt: SYS.1.6.A33

Die Container SOLLTEN nur über die notwendigen Netzports miteinander kommunizieren können. Es SOLLTEN innerhalb der virtuellen Netze Regeln existieren, die alle bis auf die für den Betrieb notwendigen Netzverbindungen unterbinden. Die Regeln SOLLTEN Quelle und Ziel der Verbindungen genau definieren und dafür mindestens die Service-Namen, Meta-Daten („Labels") oder die Service-Accounts verwenden. Sofern möglich SOLLTEN die Regeln eine zertifikatsbasierte Authentifizierung vorschreiben und nur die in den Zertifikaten hinterlegten Identitäten für die Definition der erlaubten Verbindungen nutzen.

Alle Kriterien, die als Bezeichnung für diese Verbindung dienen, SOLLTEN so abgesichert sein, dass nur berechtigte Personen und Verwaltungs-Dienste diese Kriterien setzen dürfen.

Anmerkungen: Die Anforderung wurde fuer K8S umformuliert und erweitert (Namespaces, CNI).

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL		per Policy prüfbar
Softwarelieferant	sollte	die zertifikatsbasierte Authentifizierung unterstützen und Kommunikationsverbindungen nur für die in den Zertifikaten hinterlegten Identitäten erlauben.	z. B. mittels mTLS Es sollte mindestens eine Serverauthentifizierung umgesetzt werden. Eine Client-Authentifizierung sollte möglich werden. Die Umsetzung kann auch über einen Service-Mesh (Bereitstellung durch Plattformbetreiber notwendig) umgesetzt werden.		x		P
Softwarelieferant	muss	die notwendigen Kommunikationsbeziehungen (auch innerhalb von Namespaces) zwischen den Komponenten der Anwendung (z.B. Containern und Pods), sowie mit Komponenten außerhalb der Container-Plattform in geeigneter Weise dokumentieren.	Es muss mindestens eines der folgenden Kriterien genutzt werden: Service-Name, Metadaten („Labels"), die Kubernetes Service Accounts oder zertifikatsbasierte Authentifizierung		x		
Softwarebetreiber	sollte	die vom Softwarelieferanten beschriebene zertifikatsbasierte Authentifizierung umsetzen.	z. B. mittels mTLS Die Umsetzung kann auch über einen Service-Mesh (Bereitstellung durch Plattformbetreiber notwendig) umgesetzt werden.	x	x		
Softwarebetreiber	muss	die vom Softwarelieferanten beschriebenen Regeln prüfen, ob diese nur die erforderlichen Kommunikationsbeziehungen zwischen den Komponenten der Anwendung (z.B. Containern und Pods) sowie mit Komponenten außerhalb der Container-Plattform zulassen.	z. B. Prüfung der Manifeste	x	x		o
Softwarebetreiber	kann	für die Definition von Quellen und Zielen der Kommunikation außerhalb der Container-Platform IP-Adressen und DNS-Namen (z.B. Services vom Typ ExternalName) verwenden.	Beispiel: Angabe von Datenbanken / Diensten, die nicht in der Platform laufen. Hinweis: Services vom Typ ExternalName https://kubernetes.io/docs/concepts/services-networking/service/#externalname	x		x	
Softwarebetreiber	muss	die zur Definition der Kommunikationsbeziehungen verwendeten Bezeichner (z.B. Labels, Serviceaccounts) so absichern, dass nur berechtige Personen und Verwaltungs-Dienste diese setzen dürfen.	Bezeichner= Labels, Serviceaccounts Ziel: Nur berechtigte Personen sollen "bewusst" Änderungen an den Labels durchführen dürfen; Geignetes Rollen/Rechtemodell ist zu implementieren.	x			
Plattformbetreiber	muss	die Absicherung der zur Definition der Kommunikationsbeziehungen verwendeten Bezeichner (z.B. Labels, Serviceaccounts) ermöglichen.	Geignetes Rollen/Rechtemodell ist zu implementieren. Autorisierung / Authentifizierung	x			
Plattformbetreiber	muss	ein Regelwerk implementieren, das alle Kommunikationen ohne explizite Allow-Regel unterbindet.	Ein Default-Deny Regelwerk muss implementiert sein.				o
Plattformbetreiber	muss	ein Container Network Interface implementieren, das den Einsatz von Filterregeln bis zum einzelnen Container ermöglicht.					

APP.4.4.A19 Hochverfügbarkeit von Kubernetes (H)

Der Betrieb SOLLTE so aufgebaut sein, dass bei Ausfall eines Standortes die Cluster und damit die Anwendungen in den Pods entweder ohne Unterbrechung weiterlaufen oder in kurzer Zeit an einem anderen Standort neu anlaufen können.
Für den Wiederanlauf SOLLTEN alle notwendigen Konfigurationsdateien, Images, Nutzdaten, Netzverbindungen und sonstige für den Betrieb benötigten Ressourcen inklusive der zum Betrieb nötigen Hardware bereits an diesem Standort verfügbar sein.
Für den unterbrechungsfreien Betrieb des Clusters SOLLTEN die Control Plane von Kubernetes, die Infrastruktur-Anwendungen der Cluster sowie die Pods der Anwendungen anhand von Standort-Daten der Nodes über mehrere Brandabschnitte so verteilt werden, dass der Ausfall eines Brandabschnitts nicht zum Ausfall der Anwendung führt.

(kh) Alt: SYS.1.6.A34

Der Containerbetrieb SOLLTE so aufgebaut sein, dass bei Ausfall eines Rechenzentrums die Anwendungen in den Containern in kurzer Zeit an einem anderen Standort neu anlaufen können. Dafür SOLLTEN alle notwendigen Konfigurationsdateien, Images, Nutzdaten, Netzverbindungen und sonstige für den Betrieb benötigten Ressourcen inklusive der zum Betrieb nötigen Hardware an diesem Standort verfügbar sein.

Anmerkung: Erweitert um den Begriff 'Cluster' - gemeint ist vermutlich der Neustart von Pods auf anderen Nodes im selben Cluster. Neue Anforderung zu der Clusteraufteilung in 'Brandabscnitte'. Ich denke damit sind auf neudeutsch 'Avalibility Zones (AZ)' gemeint.

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	muss	eine Fehlertoleranz für die Anwendung für den Wiederanlauf nach einem ungeordneten Abbruch der Datenverarbeitung sicherstellen.	Mögliche Maßnahmen: atomare Gestaltung Keine persistente Daten im Container Synchrone Transaktionen falls mehrere Datenbanken, Webservices usw. genutzt werden. Mechanismus zur Prüfung der Konsistenz der Daten, ggf. Mittel zur Wiederherstellung der Konsistenz		x	---
Softwarelieferant	muss	die Anwendung ~~Lösung~~ so gestalten, dass eine Hochverfügbarkeit umgesetzt werden kann.	Mögliche Maßnahmen: stateless-Anwendungen		x	
Softwarebetreiber	muss	definieren, wie viele Ressourcen der Betrieb seiner Software in den abgesetzten Standorten benötigt.	Es ist denkbar, dass der Notbetrieb geringere Anforderungen als der Primärbetrieb benötigt. Beispiele für Ressourcen sind: CPU, RAM, Festplatte, IO-Leistung			
Softwarebetreiber	muss	ein Konzept zum Wiederanlauf oder kontinuierlichen Betrieb an einem anderen Standort erstellen und dieses regelmäßig verproben.	Konzept sollte folgendes beinhalten: alle notwendigen Konfigurationsdateien, Images, Nutzdaten, Netzverbindungen und sonstige für den Betrieb benötigten Ressourcen inklusive der zum Betrieb nötigen Hardware bereits an diesem Standort	x		
Softwarebetreiber	muss	RTO (Recovery Time Objective) und RPO (Recovery Point Objective) definieren und dem Plattformbetreiber mitteilen.	Folgende Punkte sollten bedacht werden: • Definition OLA (Operational Level Agreement) und/oder SLA (Service Level Agreement) zwischen Softwarebetreiber und Plattformbetreiber. • Plattformbetreiber muss die Anforderungen an seine Plattform erhalten. • Umsetzung kann auch eine Beratung und Anpassung der RPO/RTO oder der Anwendung sein. • Der Softwarebetreiber muss den Plattformbetreiber entsprechend seiner Anforderungen auswählen.			
Softwarebetreiber	muss	seine Kubernetes-Objekt Definitionen (z.B. ConfigMap, Service, Deployment...) auch außerhalb des Clusters an einem anderen Standort speichern.	z.B. Versioniert in einem Source Code Management System (Git, SVN, ...).	x		
Softwarebetreiber	muss	seine Datenspeicher so wählen, dass die Nutzdaten am Wiederanlaufort entsprechend RTO und RPO wiederhergestellt werden können.	Plattformbetreiber stellt diese bereit/bindet diese an. Datenspeicher umfasst jegliche Speicherart, sei es Block, File, Object oder auch Datenbanken.			
Softwarebetreiber	muss	sicherstellen, dass die Images georedundant gespeichert sind.		x		
Softwarebetreiber	soll	das Deployment seiner Container-Anwendung automatisiert haben.	• Bspw. durch Nutzung parametrisierter CI/CD Pipelines • Das Deployment der Anwendung am zweiten Standort sollte berücksichtigt werden.	x		

Softwarebetreiber soll	deklarative Methoden zur Automatisierung verwenden.	Die Automation soll den gewünschten Zustand beschreiben und nicht imperative Veränderungen, die auf einem bekannten Zustand aufsetzen erfordern. Dies ermöglicht den Austausch von Clustern/Infrastruktur über Parametrisierung im Falle von Standort- oder Clusterausfällen. Nutzung von YAML anstatt Kubectl	x
Softwarebetreiber kann	Aktiv-Aktiv oder Aktiv-Passiv Architekturen verwenden		x
Softwarebetreiber soll	Vorgaben zur Umsetzung der Verteilung der die Pods der Anwendungen anhand von Standort-Daten der Nodes umsetzen.		
Plattformbetreiber soll	die Control Plane und Workernodes sinnvoll auf unterschiedliche Brandabschnitte verteilen.		
Plattformbetreiber muss	ein Konzept zum Wiederanlauf oder kontinuierlichen Betrieb der Plattform an einem anderen Standort erstellen und dieses regelmäßig verproben.	Hinweis: Nicht nur die Plattfporm betrachten, auch die Datenübertragung beachten.	
Plattformbetreiber muss	Infrastrukturen mit ausreichender Kapazität an mehr als einem Standort vorhalten.	Die Planung die für Standort A existiert (inkl. variabler Anteile) muss in den anderen Standorten aufgenommen werden können.	
Plattformbetreiber sollte	die Standorte für die Bereitstellung aller für den Betrieb von Containern erforderlichen Komponenten mit ausreichender Entfernung zueinander wählen (Geo-Redundanz).		
Plattformbetreiber muss	Image-Registries und Datenspeicher geo-redundant bereitstellen.	Datenspeicher umfasst jegliche Speicherart, sei es Block-, File-, Objectspeicher oder auch Datenbanken. Die Anbindung besagt lediglich, dass der Datenspeicher aus der Plattform zugreifbar sein muss, nicht dass es nativ in Kubernetes integriert sein muss (bspw. als StorageClass).	x
Plattformbetreiber muss	die Erreichbarkeit der Registries an verschiedenen Standorten gewährleisten.	Dies vereinfacht den Softwarebetreibern die Erfüllung der Anforderung die Images georedundant bereitzustellen.	
Plattformbetreiber soll	den Zugriff auf Datenspeicher fehlertolerant gestalten.	Es ist eine netzwerkseitige Redundanz zu beachten. Für die Inbetriebnahme am abgesetzten Standort sollen die Konfigurationsänderungen minimiert (optimal: keine) werden.	
Plattformbetreiber soll	Backup-Dienste für Manifeste und die Cluster-Konfiguration anbieten.		
Plattformbetreiber kann	Dienste anbieten, die einen Aktiv-Aktiv Betrieb über mehrere Standorte hinweg ermöglichen.	Dies kann beispielsweise eine Loadbalancer-Komponente sein, die im Netzbereich eine Lastverteilung über die verschiedenen Kubernetes-Cluster/Standorte ermöglicht.	
Plattformbetreiber kann	die fehlertolerante Gestaltung der Plattform durch verschiedene Kubernetes-Cluster in unterschiedlichen Standorten oder als einen über mehrere Standorte gestreckten Kubernetes-Cluster anbieten.		

APP.4.4.A20 Verschlüsselte Datenhaltung bei Pods (H)

Die Dateisysteme mit den persistenten Daten der Control Plane (hier besonders etcd) und der Anwendungsdienste SOLLTEN verschlüsselt werden.

Definition Anwendungsdienste: siehe Glossar

Anmerkung 08.09.2022 (große Runde): Es wurde entschieden, dass hier nur die Dateisysteme betrachtet werden und nicht die Verschlüsselung der Anwendungsdienste.

Bedingt durch den Fokus auf den Plattformbetrieb (Control Plane & Betrieb der Anwendungsdienste = PaaS) ist für die Softwarelieferanten und -betreiber die Betrachtung eher schwierig.

(kh) Alt: SYS.1.6.A35

Anmerkung: Änderung in Richtung von K8S mit der Management Plane und etcd.

Rolle	Prio	Anforderung	Anmerkungen	CI/CD	Whitepaper SWL	per Policy prüfbar
Plattformbetreiber	muss	mindestens eine Variante für verschlüsselten persistenten Speicher als Anwendungsdienst anbieten.				P
Plattformbetreiber	muss	Dateisysteme mit den persistenten Daten der Control Plane (hier besonders etcd), wenn technisch möglich, verschlüsseln.				
Plattformbetreiber	muss	Verschlüsselungsalgorithmen und Schlüsselmanagement nach dem aktuellen Stand der Technik einsetzen.				o

APP.4.4.A21 Regelmäßiger Restart von Pods (H)

Bei einem erhöhten Risiko für Fremdeinwirkung und einem sehr hohen Schutzbedarf SOLLTEN Pods regelmäßig gestoppt und neu gestartet werden. Kein Pod SOLLTE länger als 24 Stunden laufen. Dabei SOLLTE die Verfügbarkeit der Anwendungen im Pod sichergestellt sein.

Rolle	Prio	Anforderung	Anmerkungen	CICD	Whitepaper SWL	per Policy prüfbar
Softwarelieferant	sollte	Software so bereitstellen, dass >= 2 Replicas verwendet werden können und bei einem rollierenden Restart der Pods die Qualität der Verfügbarkeit nicht beinträchtigt wird.	Bitte die Anforderungen durch den Schutzbedarf beachten. Die Software sollte generell zum Rolling-Update befähigt sein.	x	x	P
Softwarelieferant	sollte	Software so bereitstellen, dass die Pods möglichst schnell starten und für die Pods keine Abhängigkeiten zu flüchtigen Caches bestehen.				
Softwarelieferant	sollte	Software so bereitstellen, dass Benutzer-Sessions bei Pod-Restarts ohne Unterbrechung bestehen bleiben.	am besten stateless, aber auch mittels Distributed Caches oder der Speicherung in der Datenbank realisierbar.			
Softwarebetreiber	muss	Anforderungen für den Softwarelieferanten definieren, dass ein rollierenden Restart realisiert werden kann.				
Softwarebetreiber	sollte	die Software so betreiben, dass ein automatischer Restart der Pods in einem definierten Zeitintervall (<= 24 Stunden) so erfolgen kann, dass Benutzer-Sessions ohne Unterbrechung bestehen bleiben und die Qualität der Verfügbarkeit nicht beinträchtigt wird.				
Softwarebetreiber	muss	den Restart der Pods so durchführen, dass es zu keinen Einschränkungen der Verfügbarkeit kommt.				
Softwarebetreiber	sollte	den Restart der Pods so durchführen, dass es zu keinen betrieblichen Einschränkungen in der Performance kommt.				
Plattformbetreiber	sollte	für die Softwarebetreiber auf der Betriebsplattform Mechanismen und Best Practices für automatisierte Restarts anbieten, die mit einfachem Customizing an die Bedürfnisse des Softwarebetreibers angepasst werden können.	Für den Restart sind verschiedene Ansätze möglich: • alle 12 Stunden den ältesten Pod restarten • in einem Durchlauf alle Pods rollierend alle 24 Stunden restarten • Startup Probe, Liveness probe für die Überwachung nutzen • Schutz vor Endless-Loop für den Restart von Pods • siehe auch Anforderung APP.4.4.A11 Überwachung der Container			