Whitepaper für die Erstellung von Containern

IG Betrieb von Containern

VERSION 2.1.0

15. Mai 2024

IG BvC

Contents

1 Ziel des Dokuments

Mit diesem Dokument beschreibt die IG Betrieb von Containern grundsätzliche Hinweise zur Entwicklung von Containerlösungen. Dieses Dokument richtet sich an Softwarelieferanten.

2 Grundsätze für die Entwicklung von Containerlösungen

Die IG Betrieb von Containern (IG BvC) ist ein Zusammenschluss von Datenzentralen, Softwarelieferanten und Organisationen der Öffentlichen Verwaltung.

Ziel der IG BvC ist die Schaffung von Standards für das Deployment von Fachverfahren in Container-Umgebungen sowie für den Betrieb von Container-Plattformen zur Herstellung einer Kompatibilität zwischen den Rechenzentren. Die Schaffung einer herstellerunabhängigen Systematik ist das Grundverständnis der IG BvC.

Die Schaffung und Umsetzung von konkreten Möglichkeiten der Koppelung und gemeinsamen Nutzung von Container-Plattformen zwischen den Rechenzentren von Bund, Ländern und Kommunen ist ein wesentlicher Grundsatz für die Zusammenarbeit.

Die Ausarbeitungen der AG Cloud und Digitale Souveränität werden bei der Ausarbeitung der Standards genutzt. Im Gegenzug werden die Ausarbeitungen der IG BvC für die Nutzung durch die AG bereitgestellt.

Als grundlegender Standard für die Betrachtungen wird Kubernetes gesetzt. Sämtliche Betrachtungen werden plattformunabhängig ausgeführt um eine Abhängigkeit von Produkten und Herstellern weitestgehend auszuschließen.

Die folgenden beiden Bausteine des BSI-Grundschutzes (Edition 2023) bilden die Grundlage

- SYS.1.6 Containerisierung
- APP.4.4 Kubernetes

3 Nomenklatur

Die Verwendung und Bedeutung der modalen Hilfsverben (alternativ Modalverben) ist in der DIN-Norm 820-2 oder RFC2119 geregelt. Ergänzend zu diesen Regeln werden die Modalverben bzw. Schlüsselwörter „MUSS", „SOLL", „IST ZU", „DARF NICHT", „SOLLTE", „SOLLTE NICHT", „KANN" und „DARF" wie folgt verwendet:

- „**MUSS**", „**IST ZU**", „**DARF NUR**" weisen auf eine absolut zu erfüllende Anforderung hin (uneingeschränkte Anforderung).
- „**DARF NICHT**", „**DARF KEIN**" ist die Negierung einer „MUSS"-Anforderung und stellt ein Verbot dar (uneingeschränktes Verbot).
- „**SOLLTE**", „**SOLL**" oder das entsprechende Adjektiv „EMPFOHLEN" bedeuten, dass eine Anforderung normalerweise erfüllt werden muss, es aber Gründe geben kann, dies doch nicht zu tun. Dies muss aber sorgfältig abgewogen und stichhaltig begründet werden.
- „**SOLLTE NICHT**" oder das korrespondierende Adjektiv „NICHT EMPFOHLEN" sind die entsprechenden Negierungen und bedeuten, dass etwas normalerweise nicht getan werden sollte, es aber Gründe gibt, dies doch zu tun. Dies muss aber sorgfältig abgewogen und stichhaltig begründet werden.
- „**KANN**", „**DARF**" zeigt eine Option an.

Folgende Definitionen für Rollen werden in der IG Betrieb von Containern verwendet:

- Der **Softwarelieferant** ist eine Organisation (juristische Person, Community), welche Softwarereleases bereitstellt.
  Wenn möglich setzt sie die Anforderungen des Software- und Plattformbetreibers um.
- Der **Plattformbetreiber** betreibt die IT-Infrastruktur Kontext IaaS und PaaS für die Containerumgebung im Rechenzentrum und stellt Mittel zur manuellen und/oder automatischen Orchestrierung bereit.
- Der **Softwarebetreiber** verantwortet den Betrieb einer Anwendung / eines Services entsprechend vertraglicher Verpflichtungen gegenüber Kunde/Auftraggeber und managed die Orchestrierung von Containern.
  Wenn möglich, stimmt er die Anforderungen an den Betrieb der Software mit dem Softwarelieferanten ab.
  Es ist das Bindeglied zwischen Plattformbetreiber und Softwarelieferant.

4 Anforderungen an die Entwicklung von Containern

4.1 Implementierung/Containererstellung

4.1.1 Datenspeicherung

Der Softwarelieferant MUSS...

- die Nutzung von temporärem Speicher so vorsehen, dass ein Restart des Containers und ein Wechsel der Nodes zur Laufzeit möglich ist. (SYS.1.6.A9 S)
- notwendige persistente Volumen und deren Nutzung (z. B. RWM, RWO, RW/RO) im Deployment Descriptor beschreiben.[1] (SYS.1.6.A19 S)
- sicherstellen, dass keine Zugangsdaten (z.B. Passworte, geheime/private Schlüssel, API-Keys, Schlüssel für symmetrische Verschlüsselungen) in Container-Images bzw. in Konfigurationsdateien gespeichert werden.[2] (SYS.1.6.A8 B)
- die Software so gestalten, dass diese keine Geheimnisse und Kennwörter als Plain-Text nutzt. (APP.4.4.A2 B)
- für seine Software sicherstellen, dass diese mit minimalen Berechtigungen lauffähig ist.[3] (SYS.1.6.A21 H)
- die Container mit Ressourcenbeschränkungen ausliefern (z.B. Datenmenge und Speicherzugriffe). (SYS.1.6A23 H)
- Daten in eigene Mount-Verzeichnisse schreiben. (SYS.1.6A23 H)
- sicherstellen, dass Daten, die im Container ausschließlich gelesen werden, als Read-Only-Volume in den Container eingebunden werden.[4] (SYS.1.6A23 H)

Der Softwarelieferant SOLL...

- keine lokalen Speicher der Workernodes benutzen.[5] (SYS.1.6.A19 S)
- für seine Software sicherstellen, dass diese nicht in das Root-File-System des Containers schreiben darf. (SYS.1.6A23 H)
- das temporäre Speichern von Daten[6] in dediziert dafür bereitgestellten Volumes (gängigerweise Ephemeral Storage) vornehmen. (SYS.1.6.A23 H)

4.1.2 Dienste und Service-Accounts

Der Softwarelieferant MUSS...

- bei Neuentwicklungen die Software so gestalten, dass jeder Container nur einen Dienst bereitstellt. (SYS.1.6.A11 S)
- für Dienste, die unterschiedlich skalieren oder auf verschiedenen Worker-Nodes betrieben werden sollen, entsprechende Parameter der Deployment Descriptoren vorsehen.[7] (SYS.1.6.A11 S)
- pro Anwendung (mindestens) einen eigenen ServiceAccount bereitstellen, mit dem die Anwendung vollständig betrieben werden kann. (APP.4.4.A9 S)
- die Software so entwickeln, dass die Service-Accounts mit den geringst möglichen Berechtigungen (Least-Privilege) betrieben werden können. (APP.4.4.A9 S)

Der Softwarelieferant SOLL...

- bei der Nutzung von zeitlich wiederkehrenden Aktivitäten auf Systemebene (z.B. CRON-Jobs) die Mechanismen der Container-Plattform zur Ausführung nutzen. (SYS.1.6.A9 S)

[1] z.B. Zielpfad im Containerbetrieb, Größe, Berechtigungen, Dateisystem, Performance. Die Beschreibung sollte in einer formalen Sprache erfolgen. Das Physical Volume Claim soll dynamisch erfolgen. Bitte mögliche Einschränkungen im Rechenzentrum beachten.

[2] Anstatt statische Secrets auszuliefern können Secrets beim Deployment generiert werden.

[3] Beispielsweise sollen für Anwendungen, die nur Lesezugriff benötigen, nur Nutzer verwendet werden, die keinen Schreibzugriff haben.

[4] Ein technischer Ansatz kann die Pod-Security-Policy "ReadOnlyRootFileSystem" sein.

[5] Es sollte nur persistent Storage genutzt werden. "Sollte" bietet Ausnahmen für flüchtige Anwendungsfälle [virenscan].

[6] bspw. Daten in temp-Verzeichnissen, wie File-Uploads, etc.

[7] Für Helm Charts ist die Parametrisierung über die values.yaml vorzusehen.

- die Container so auslegen, dass eine horizontale Skalierung möglich ist. (SYS.1.6.A9 S)
- die Software so gestalten, dass jeder Container nur einen Dienst bereitstellt.[8] (SYS.1.6.A11 S)
- die Software so gestalten, dass Pods nicht den default-Service-Account benötigen.[9] (APP.4.4.A9 S)

4.1.3 Monitoring

Der Softwarelieferant SOLL...

- Liveness- und Readiness-Checks gemäß der Kubernetes-Dokumentation[10] bereitstellen. Die Checks, insbes. die Readiness-Checks, sind anwendungsspezifisch zu betrachten.[11] (APP.4.4.A11 S)
- die forensische Analyse unterstützen, indem u.a. die eingehenden und ausgehenden Requests, sowie Konfigurationsänderungen geloggt werden. (SYS.1.6.A22 H)
- ein für seine Software erwartbares Verhalten (Systemaufrufe, Kommunikationsbeziehungen usw.) definieren, beschreiben und dem Softwarebetreiber vorlegen.[12] (SYS.1.6.A24 H)

Der Softwarelieferant KANN...

- Start-up-Checks für den Start der Container implementieren.[13] (APP.4.4.A11 S)

4.1.4 Images

Der Softwarelieferant MUSS...

- sicherstellen, dass gelieferte Artefakte, insb. Images, dem IT-Grundschutz entsprechen. (SYS.1.6.A10 S)
- das Image mit Metainformation versehen und signieren.[14] (SYS.1.6.A12 S)
- bewerten und sicherstellen, dass sämtliche Bestandteile seiner Anwendungsimages aus vertrauenswürdigen Quellen stammen. Dabei kann auf vom Plattformbetreiber freigegebene Base-Images aufgesetzt werden. (SYS.1.6.A6 S)
- dem Softwarebetreiber Container Images mit allen zur Runtime benötigten Modulen bereitstellen. Ein Nachladen von Modulen DARF NICHT erfolgen. (SYS.1.6.A6 S)
- regelmäßig Images auf Schwachstellen und Schadcode prüfen[15] und bei Bedarf geeignete Updates zur Verfügung stellen.[16] (SYS.1.6.A6 S)

Der Softwarebetreiber MUSS...

- mit dem Softwarelieferanten die Basis für gelieferte Images abstimmen. (SYS.1.6.A9 S)
- Vorgaben für möglichst minimale Images festlegen und veröffentlichen. (SYS1.6.A6 S)

Der Plattformbetreiber MUSS ...

- Richtlinien für die Bereitstellung von Images dem Softwarelieferanten zur Verfügung stellen (SYS1.6.A6 S)
- Images unmittelbar bei der Bereitstellung in geeigneter Weise auf Schadcode und Schwachstellen prüfen. (SYS1.6.A6 S)

Der Plattformbetreiber SOLL ...

- automatisierte Policies implementieren, die die Herkunft, Vertrauenswürdigkeit und Integrität der Images prüfen und durchsetzen. (SYS1.6.A6 S)

[8] Die Einstufung mit "soll" ist nur aufgrund der Übergangsphase aus der "klassischen" in die Containerwelt vorgenommen worden. Es wird ein "muss" angestrebt.

[9] Bei OpenShift sind die Rechte des "default" Service Account bereits maximal eingeschränkt.

[10] siehe https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/

[11] siehe einschlägige Best-Practices, bspw. https://cloud.google.com/blog/products/containers-kubernetes/kubernetes-best-practices-setting-up-health-checks-with-readiness-and-liveness-probes

[12] Hinweis: Wenn der Softwarelieferant das normale Verhalten -wie oben angesprochen- nicht angeben kann, dann muss der Softwarebetreiber das normale Verhalten durch Erprobung feststellen.

[13] Hinweis: https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/#define-startup-probes

[14] Es muss eine Liste von Metadaten die mindestens vorhanden sein müssen existieren (Versionierung und Tagging von Images).

[15] Gängig sind in diesem Zusammenhang Security Scanner der Image Registry.

[16] Welche Reaktionszeiten angemessen sind, ist vertraglich zu definieren.

4.1.5 Kommunikationsverbindungen

Der Softwarelieferant MUSS...

- die Software so bereitstellen, dass keine Fernwartung möglich ist.[17] (SYS.1.6.A16 S)
- Zugriffe und Berechtigungen auf Ressourcen durch seine Software auf die technisch notwendigen Zugriffe beschränken. (APP.4.4.A21 H und SYS.1.6.A21 H)
- die Vorgaben und Rahmenbedingungen des Software- und des Plattformbetreibers für die Kommunikationsbeziehungen berücksichtigen und seine Implementierung entsprechend auslegen.[18] (SYS.1.6.A5 B)
- die eingehende und ausgehende Kommunikation der Anwendung über Kubernetes-Serviceressourcen (Ingress und Egress) umsetzen.
- 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. (APP.4.4.A18)

Der Softwarelieferant SOLL...

- die zertifikatsbasierte Authentifizierung unterstützen und Kommunikationsverbindungen nur für die in den Zertifikaten hinterlegten Identitäten erlauben.[19] (APP.4.4.A18)

4.1.6 Systemarchitektur

Der Softwarelieferant MUSS...

- Software-Produkte so entwickeln, dass ein Re-Deployment auf einem "Restore-Cluster" jederzeit möglich ist (APP.4.4.A5 B)

Der Softwarelieferant SOLL...

- die Anwendung so bereitstellen, dass die lt. Schutzbedarf erforderliche Anwendung von technischen Schutzmaßnahmen wie z.B. die Verwendung von Betriebssystem-Mechanismen, die Isolation und Kapselung der Anwendung und auch die Netzwerktrennung durch die Software-Architektur unterstützt und nicht verhindert wird.[20] (SYS.1.6.A3 B, APP.4.4.A4 B)
- 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.[21] (APP.4.4.A5 B)

Der 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. (APP.4.4.A4 B)
- der Plattformbetreiber MUSS technische Maßnahmen anbieten, die eine Isolation und Kapselung containerisierter IT-Systeme sowie die Trennung dieser IT-Systeme in unterschiedliche virtuelle Netze ermöglichen. Dabei muss die Netzwerkarchitektur (physische, virtuelle und Overlay-Netze) so gestaltet

[17] Die Container Images DÜRFEN KEINE Komponenten zur Fernwartung (zum Beispiel ssh, telnet) enthalten.

[18] Bei Anbietern für Standardsoftware sollte der Plattformbetreiber prüfen, ob seine Anforderungen dem Marktangebot gerecht werden.

[19] 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.

[20] durch Plattform vorgegebene Isolationsmechanismen wie z.B. SE Linux und CGroups; plattformspezifische Isolationsmechanismen wie z.B. Namespaces, Pod Security Admission und Network Policies

[21] 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

werden, dass die Anforderungen an einen sicheren Netzwerkbetrieb lt. BSI-Grundschutz NET.1 und NET.3 erfüllt sind. (SYS.1.6.A3 B)

4.2 Konfiguration/Administration

Der Softwarelieferant MUSS...

- Konfigurationsanpassungen so ausführen, dass keine manuellen Anpassungen in laufenden Containern erfolgen. (SYS.1.6.A9 S)
- die Software so gestalten, dass die Container-Runtime und alle instanziierten Container nur von einem nicht-privilegierten System-Account ausgeführt werden, der keine erweiterten Rechte für den Container-Dienst bzw. das Betriebssystem des Host-Systems benötigt oder diese Rechte erlangen kann. Ausnahmen hiervon sind Container, welche Aufgaben des Host-Systems übernehmen. [22] (SYS.1.6.A17 S)
- die Privilegien für Container auf das erforderliche Minimum begrenzen. (SYS.1.6.A17 S)

- die Informationen zur Einschränkung der Nutzung von erweiterten Privilegien dem Softwarelieferanten zur Verfügung stellen. (SYS.1.6.A17 S)
- die Software so gestalten, dass die genutzten User- und Group-ID's keine Berechtigungen auf die System- und Datenbereiche des Hosts erfordern. (SYS.1.6.A18 S)
- die erforderlichen User- und Group-ID's und deren Berechtigungen benennen (z.B. in den Konfigurationsdateien). (SYS.1.6.A18 S)
- die mitgelieferte Konfiguration der Container versioniert bereitstellen und den Bezug zu Imageversionen sicherstellen. [23] (SYS.1.6.A20 S)
- das vom Softwarebetreiber benötigte Verteilungsschema für die Pods unterstützen. [24] (APP.4.4.A14 H)
- notwendige Berechtigungen im Deployment beschreiben. Diese müssen minimal gewählt werden. (APP.1.6.A19 S)
- den Abschluss eines Init-Container sicherstellen. (APP.4.4.A6 S)
- den Init-Container eindeutig von der Applikation logisch trennen. (APP.4.4.A6 S)

Der Softwarelieferant SOLL...

- das automatische Management seiner Software durch die Bereitstellung von geeigneten Administrationswerkzeugen unterstützen.[25] (SYS.1.6.A2 B)
- Init-Container für Konfigurationsanpassungen nutzen. (SYS.1.6.A9 S)
- die Deployments entsprechend der Richtlinien des Softwarebetreibers ausrichten.[26] (APP.4.4.A13 H)
- bei besonders kritischen Anwendungen Operatoren zur Automatisierung von Betriebsaufgaben liefern. (APP.4.4.A16 H)

Der Softwarebetreiber MUSS...

- dem Softwarelieferanten geeignete Möglichkeiten zum Geheimnisaustausch bieten. (SYS.1.6.A20 S)

Der Plattformbetreiber MUSS...

- geeignete Möglichkeiten zum mandantenfähigen Geheimnismanagement bereitstellen. (SYS.1.6.A20 S)

Beispiele für Aufgaben des Hostsystems sind: i) ingress und egress sind Bastion-Node-Aufgaben, ii) Bereitstellung von Speichersystemen sind Speicher-Node-Aufgaben

[22] kein "privileged"-Mode für die Container-Runtime und den laufenden Containern Beispiel für Ausnahmen sind: ingress, egress und Infrastrukturcontainer

[23] Entweder z.B. durch Label in der YAML oder als git-repo.

[24] z.B. Aufteilung der Pods entsprechend der Aufgaben. Der Softwarebetreiber muss das Verteilungsschema für die Pods definieren. Die Nodes müssen mindestens nach folgenden Aufgaben unterschieden und getrennt werden: i) Bastion-Nodes zur Realisierung von ingress und egress, ii) Anwendungs-Nodes zum Betrieb der Pods für die Anwendungen, iii) Speicher-Nodes zur Bereitstellung der Speicherlösungen, iv) Management-Nodes für den Cluster

[25] Werkzeuge können Deployment-Scripte, Pipeline-Scripte für CI/CD, Kubernetes Operatoren sein.

[26] Softwarebetreiber muss Richtlinien für die Einstellungen von Deployments definieren und dem Softwarelieferanten bekannt geben.

4.3 Protokollierung

Der Softwarelieferant MUSS...

- alle Protokolldaten der Anwendung im Container über die Standardausgabe ausgeben.[27] (SYS.1.6.A7 B)

4.4 Fehlertoleranz

Der Softwarelieferant MUSS...

- eine Fehlertoleranz für die Anwendung für den Wiederanlauf nach einem ungeordneten Abbruch der Datenverarbeitung sicherstellen.[28] (APP.4.4.A19 H)

Der Softwarelieferant SOLL...

- die Anwendung Stateless gestalten, um die Mittel der Plattform bei einer Umsetzung von Hochverfügbarkeit nutzen zu können.[29] (APP.4.4.A19 H)
- Container stateless gestalten und eine transaktionsoriente Funktionsweise der Anwendung sicherstellen.[30] (SYS.1.6.A9 S)

Der Softwarelieferant KANN...

- Start-up-Checks für den Start der Container implementieren.[31] (APP.4.4.A11 S)

4.5 Update- und Patchmanagement

Der Softwarelieferant MUSS...

- ein Patchmanagement gewährleisten. (SYS.1.6.A10 S)
- sicherstellen, dass Updates von Software nicht im laufenden Container installiert werden. Bei persistenten Containern SOLLTE geprüft werden, ob in (absolut seltenen) Ausnahmefällen ein Update des jeweiligen Containers geeigneter ist, als den Container vollständig neu zu provisionieren.[32] (SYS.1.6.A14 S)
- bei sicherheitsrelevanten und featurebasierten Updates der Anwendungen oder der Standardimages neue Images erstellen und eindeutig versioniert innerhalb der vereinbarten Zeiträume / SLAs zur Verfügung stellen. (SYS.1.6.A14 S)
- das Rechte- und Rollenkonzept für die Verwaltung des Sourcecodes umsetzen.[33] (APP.4.4.A2 B)

Der Softwarelieferant SOLL...

- sicherstellen, dass die Software in der Lage ist, den Service auch während Updates entsprechend der definierten Anforderungen aufrechtzuerhalten.[34] (SYS.1.6.A14 S)

Der Softwarebetreiber MUSS...

- sicherstellen, dass die Software neu angelieferter Images auf Funktionionalität und Schadcode geprüft und innerhalb der vereinbarten Zeitfenster / SLAs für den produktiven Container-Betrieb bereitgestellt werden. (SYS.1.6.A14 S)

[27] Log-Daten MÜSSEN über STDOUT / STDERR ausgegeben werden.
[28] 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.
[29] Die Optionen hinsichtlich Zustandslosigkeit sind für Bestandsanwendungen beschränkt und lassen sich ggf. nicht umsetzen.
[30] Die transaktionsorientierte Funktionsweise soll sicherstellen, dass alle Verarbeitungen korrekt abgeschlossen werden können und keine inkonsistenten Zustände entstehen, selbst wenn Container neu deployed werden.
[31] Siehe https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/#define-startup-probes
[32] Es dürfen keine Tools enthalten sein, welche nicht für den Betrieb der Applikation notwendig sind (Paketmanager, YUM, APT, APK usw.).
[33] siehe auch CON.8
[34] d.h. Clusterfähigkeit der Software

Der Softwarebetreiber SOLL... - die benötigten SLAs zur Aktualisierung der Images mit dem Softwarelieferanten vereinbaren. (SYS.1.6.A14 S) - sicherstellen, dass keine veralteten Images zum Einsatz kommen und sicher stellen, dass neue Images regelmäßig vom Softwarelieferanten in die eigene Registry übernommen werden. (SYS.1.6.A14 S) - gemäß OPS.1.1.3 Patch- und Änderungsmanagement entscheiden, wann und wie die Updates der Images oder der betriebenen Software bzw. des betriebenen Dienstes ausgerollt werden. (SYS.1.6.A14 S)

5 Dokumentation

5.1 Dokumentation des Containers

Der Softwarelieferant MUSS...

- die Richtlinien und Vorgaben zum sicheren Betrieb dem Softwarelieferanten bereitstellen. (SYS.1.6.A10 S)
- dokumentieren, welche Quellen für Images verwendet wurden. (SYS.1.6.A12 S)
- beschreiben, welche Quellen für Bestandteile des Image verwendet wurden, inklusive eines ggf. genutzten Basis-Images[35] (SYS.1.6.A12 S)
- Dimensionierungen (Requests) und Begrenzungen (Limits) für CPU, RAM sowie temporären Speicher für jeden Container auf den von ihm angenommenen Nutzerprofil empfehlen.[36] (SYS.1.6.A15 S)
- die Möglichkeiten und Funktionsweisen zur Skalierung der Dienste beschreiben.[37] (SYS.1.6.A15 S)
- Konfigurationsoptionen für die Protokollierung beschreiben.[38] (SYS.1.6.A7 B)
- den Einsatz eines Init-Containers dokumentieren.[39] (APP.4.4.A6 S)
- die Ressourcenanforderungen an die Init-Container beschreiben und dem Softwarebetreiber mitteilen. (APP.4.4.A6 S)

Der Softwarelieferant SOLL...

- die Umsetzung der Richtlinien und Vorgaben mit dem Softwarelieferanten vertraglich festhalten.[40] (SYS.1.6.A10 S)
- Handlungsempfehlungen bereitstellen, für den Fall, dass ein Container an seine Ressourcengrenzen kommt. (SYS.1.6.A15 S)
- die Möglichkeiten der Hochverfügbarkeit darstellen (SYS.1.6.A25 H)

Der Softwarelieferant KANN...

- dem Softwarebetreiber ein Klassifikationssystem für die Erstellung einer Whitelist für vertrauenswürdige Quellen vorschlagen. (SYS.1.6.A12 S)

Der Softwarebetreiber MUSS...

- die Klassifikation von vertrauenswürdigen Quellen vornehmen und dem Softwarelieferant mitteilen. (SYS.1.6.A12 S)
- die Klassifizierung der vertrauenswürdigen Quellen begründen. (SYS.1.6.A12 S)
- die angegebenen Begrenzungen der Ressourcen (Limits) durch entsprechende Konfiguration der Container umsetzen. (SYS.1.6.A15 S)

5.2 Dokumentation von Anforderungen an persistenten Speicher

Der Softwarelieferant MUSS...

- die Dimensionierungen und die Größe des benötigten persistenten Speichers empfehlen. Die Informationen müssen dem Softwarebetreiber mitgeteilt werden. [41] (SYS.1.6.A15 S)

5.3 Kommunikationsbeziehungen dokumentieren

Der Softwarelieferant MUSS...

[35] Dies dient der Verdeutlichung des konzeptionellen Aufbaus der Software (z.B. zur Beurteilung hinsichtlich der verwendeten Lizenzen).

[36] Der Softwarelieferant muss Defaultwerte für die Limits liefern, die dann ggf. vom Softwarebetreiber angepasst werden können.

[37] z.B. kubernetes pod hpa - https://kubernetes.io/de/docs/tasks/run-application/horizontal-pod-autoscale/ - Anmerkung: Limitierung muss auch berücksichtigt werden, wenn der Dienst skaliert werden soll.

[38] Der Softwarebetreiber soll in die Lage versetzt werden, die Protokollierung , bspw. Log-Level, -Format oder -Kategorien, einzurichten.

[39] Es müssen die Berechtigungen dargestellt werden. Er muss die notwendigen Ressourcen dokumentieren und eine Empfehlung bereitstellen.

[40] Insbesondere bei der Lieferung von Images, da hier die Fähigkeit der Anpassung der Images beschränkt wird.

[41] Der Softwarelieferant muss Defaultwerte für die Limits liefern, die dann ggf. vom Softwarebetreiber angepasst werden können.

- alle erforderlichen Kommunikationsverbindungen inklusive Ziele, Kommunikationsprotokolle und Ports beschreiben, die für Inbetriebnahme und Betrieb erforderlich sind. (APP.4.4.A7 S [42], siehe auch SYS.1.6.A5 B, siehe auch APP.4.4.A14 H [43]).
- benötigte Zugriffe und Berechtigungen auf Ressourcen dokumentieren. (APP.4.4.A21 H)
- dem Softwarebetreiber die von Ihm definierten erweiterten Maßnahmen mitteilen, sofern vorhanden. [44] (SYS.1.6.A26 H)

Der Softwarelieferant SOLL...

- Dimensionierungen (Requests) und Begrenzungen (Limits) auf den von ihm angenommenen Nutzerprofil für das Netzwerk definieren und dem Softwarebetreiber mitteilen. (SYS.1.6.A15 S)

Der Softwarelieferant KANN...

- in Abstimmung mit dem Softwarebetreiber die Kommunikationsbeziehungen in einer technisch formalen Form darstellen. (APP.4.4.A7 S [45], siehe auch SYS.1.6.A5 B)

5.4 Berechtigungen

Der Softwarelieferant MUSS...

- darlegen, für welche Zwecke technische User (z.B.: Kubernetes ServiceAccounts) Berechtigungen erfordern[46] [47] (APP.4.4.A3 B)

5.5 Lizenzen

Der Softwarelieferant MUSS...

- dem Softwarebetreiber bekannt geben, welche Lizenzen auf der Container-Plattform für den Betrieb der Software erforderlich sind und diese je nach vertraglicher Vereinbarung auch bereitstellen. (SYS.1.6.A9 S)

[42] Bei klar definierten Zielen sollten FQDN oder IP-Adressen zur Beschreibung genutzt werden.
[43] Auch die Kommunikation innerhalb der Anwendung muss bekannt und beschrieben sein.
[44] Feste Zuordnung von Containern zu Container-Hosts, Ausführung der einzelnen Container und/oder des Container-Hosts mit Hypervisoren und feste Zuordnung eines einzelnen Containers zu einem einzelnen Container-Host.
[45] siehe https://kubernetes.io/docs/concepts/services-networking/network-policies
[46] Der Softwarebetreiber muss die benötigten Accounts und Berechtigungen beim Plattformbetreiber anfordern und sie müssen den Privilegienrichtlinien entsprechen
[47] Der Softwarebetreiber muss anonyme Zugriffe für administrative Handlungen verhindern.

6 Lieferung

Der Softwarelieferant MUSS...

- eine vollständige Software Bill of Materials (SBOM) in einem abgestimmten Format abgeben.[48] (SYS.1.6.A13 S)
- einen definierten Prozess für den Bezug und die Weitergabe von Images etablieren und nachvollziehbar dokumentieren.[49] (SYS.1.6.A4 B)
- einen standardkonformen Übergabepunkt zum Softwarebetreiber verwenden.[50] (SYS.1.6.A4 B)
- die unterschiedlichen Schutzbedarfe bei der Bereitstellung der Lösung unterstützen.[51] (APP.4.4.A2 B)

Der Softwarelieferant SOLL...

- Deployment Manifeste zur Installation bereitstellen. (SYS.1.6.A9 S)
- sicherstellen, dass zusammenhängende Dienste durch das Deployment bzw. die Orchestrierung als Verwaltungseinheiten bereitgestellt werden können und z.B. geeignete Deployment-Artefakte (Manifeste) liefern. (SYS.1.6.A11 S)
- für die durch ihn bereitgestellten Software-Artefakte automatisiert zu verarbeitende Deployment-Anweisungen bereitstellen (z.B. YAML-basierte Deployments). (SYS.1.6.A11 S)
- Build-Spezifikationen (bspw. Dockerfile) für Images liefern.[52] (SYS.1.6.A13 S)
- neben einem Freigabeprozess für die erstellte Software[53] auch einen Freigabeprozess für Images und Deployment Deskriptoren umsetzen. (SYS.1.6.A13 S)
- die Freigabebedingungen in einem definierten Freigabeprozess im Lieferantenkontext prüfen und alle Informationen für den weiteren Freigabeprozess an den Softwarebetreiber übergeben. Zu den übergebenen Informationen gehören auch die Ergebnisse des eigenen Freigabeprozesses. (SYS.1.6.A13 S)

Der Softwarelieferant KANN...

- die zu liefernden Images bereits vor der finalen Lieferung mit dem Prüfstandard des Softwarebetreibers überprüfen. [54] (SYS.1.6.A13 S)

Der Softwarebetreiber MUSS

- die Informationen zum Prozess der Bereitstellung und Verteilung von Images an den Softwarelieferanten übergeben und sich mit diesen abstimmen. (SYS.1.6.A4 B)
- prüfen, dass die durch den Softwarelieferanten gelieferten Software-Artefakte nur einen Dienst je Container oder Verwaltungseinheit bereitstellen und geeignete Deployment-Anweisungen verfügbar sind. (SYS.1.6.A11 S)
- den sicheren Transport der Deployment-Artefakte, insb. Images, unter Berücksichtigung des Bausteins *CON.9 Informationsaustausch* sicherstellen. (SYS.1.6.A12 S)
- die Freigabebedingungen in einem Freigabeprozess gemäß des Bausteines OPS.1.1.6 Software-Test und -Freigaben definieren. (SYS.1.6.A13 S)
- einen definierten Freigabeprozess für die Produktivsetzung der Anwendung / neuer Versionen etablieren. (SYS.1.6.A13 S)

Der Softwarebetreiber SOLL

- bei der Definition des Prozess zur Bereitstellung und Verteilung von Images folgendes berücksichtigen: (SYS.1.6.A4 B)
  - Auflistung erforderlicher Lizenzen

[48] Dokumentation der genutzten Komponenten, Ermöglichung der Prüfung von Lizenzen

[49] Der Softwarebetreiber muss seine Prozess entsprechend anpassen können. Die Nachvollziehbarkeit ist Grundlage für die Prüfung der gelieferten Software.

[50] Bereitstellung in einer Registry nach OCI-Registry-Format.

[51] Gerade beim Test muss die strikte Trennung (Schutzbedarf und Mandant) umgesetzt werden, da hier Code ausgeführt wird.

[52] Dient der Ermöglichung der statischen Analyse auf Schwachstellen und Schadsoftware

[53] Siehe dazu auch Baustein OPS.1.1.6 Software-Test und Freigaben

[54] Das ermöglicht dem Lieferanten, seine eigene Software qualitätszusichern und nicht vom Prüfergebnis des Betreibers "überrascht" zu werden. Das kann z.B. durch eine standardisierte Entwicklungsumgebung realisiert werden, die dem Lieferanten bereitgestellt wird.

  - Bereitstellung von Deployment-Artefakten und Build-Spezifikationen
  - Prüfung auf Schwachstellen (statische und dynamische Tests), Schadsoftware, veraltete Komponenten, Build-Dependencies
  - Prüfsummen, Signaturen und Herkunft
  - Lifecycle und Patchmanagement von Images
  - Nutzung von vertrauenswürdigen Registries
- die Prüfung der durch den Softwarelieferanten bereitgestellten Artefakte und Deployment-Anweisungen automatisiert vornehmen und in seine CI/CD-Prozesse integrieren. (SYS.1.6.A11 S)
- allen Anforderungen entsprechenden Freigabebedingungen für die Inbetriebnahme von Images und Konfigurationen definieren und für den Softwarelieferanten bereitstellen. Hierbei sind die Freigabebedingungen des Plattformbetreibers zu berücksichtigen. (SYS.1.6.A13 S)

7 Anhänge

7.1 Referenzen

Kürzel	Beschreibung
SYS.1.6	Bundesamt für Sicherheit in der Informationstechnik (BSI). *IT-Grundschutz-Baustein SYS.1.6 Containerisierung*, 1. Februar 2022, online verfügbar unter https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-GS-Kompendium_Einzel_PDFs_2022/07_SYS_IT_Systeme/SYS_1_6_Containerisierung_Edition_2022.html
APP.4.4	Bundesamt für Sicherheit in der Informationstechnik (BSI). *IT-Grundschutz-Baustein APP.4.4 Kubernetes*, 1. Februar 2022, online verfügbar unter https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-GS-Kompendium_Einzel_PDFs_2022/06_APP_Anwendungen/APP_4_4_Kubernetes_Edition_2022.html

7.2 Versionshistorie

Version	Datum	Status, Änderungsgrund
1.0	17.06.2021	Erstversion
2.0	15.06.2023	Anpassung auf BSI Grundschutz 2022, Bausteine APP4.4 und SYS1.6
2.1	06.05.2024	CRs der AG Blickwinkel Softwarelieferant