Whitepaper für die Erstellung von Containern

VERSION:

1.0

STATUS:

fertiggestellt

AUTOR*IN:

IG Betrieb von Containern

STAND:

Juni 2021

IG BvC

INHALT

1 ZIEL DES DOKUMENTES 3

2 GRUNDSÄTZE FÜR DIE ENTWICKLUNG VON CONTAINERLÖSUNGEN 4

3 NOMENKLATUR 5

4 ANFORDERUNGEN AN DIE ENTWICKLUNG VON CONTAINERN 6

4.1 Programmierung/Containererstellung 6

4.1.1 Datenspeicherung 6

4.1.2 Dienste und Service-Accounts 6

4.1.3 Monitoring Betriebsstatus 7

4.1.4 Basisimages 7

4.1.5 Kommunikationsverbindungen 8

4.2 Konfiguration/Administration 8

4.3 Protokollierung 8

4.4 Fehlertoleranz 9

4.5 Update- und Patchmanagement 9

5 DOKUMENTATION 10

5.1 Dokumentation des Containers 10

5.2 Speicher und/oder Datenbanken 10

5.3 Konnektierte Netze 11

5.4 Berechtigungen 11

5.5 Lizenzen 12

6 LIEFERUNG 13

1 ZIEL DES DOKUMENTES

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.

Die Ausarbeitungen des BSI-Bausteins SYS.1.6 Container (Community Draft) werden als Grundlage genutzt. Als grundlegender Standard für die Betrachtungen wird Kubernetes gesetzt. Sämtliche Betrachtungen werden plattformunabhängig ausgeführt um eine Abhängigkeit von Produkten und Herstellern weitestgehend auszuschließen.

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 Service 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 Programmierung/Containererstellung

4.1.1 Datenspeicherung

Der Softwarelieferant MUSS…

- sicherstellen, dass Nutzdaten nicht innerhalb des Containers gespeichert werden (SYS.1.6.A9).
- sicherstellen, dass keine Zugangsdaten (z.B. Passworte, geheime/private Schlüssel, API-Keys) in Container-Images gespeichert werden (SYS.1.6.A10).
- das temporäre Speichern von Daten in einem Container begründen und die Daten in einem besonders dafür abgegrenzten Bereich zwischenspeichern (SYS.1.6.A15).

Der Softwarelieferant SOLL…

- die Speicherung temporärer Daten innerhalb von Containern vermeiden (SYS.1.6.A15).
- eine Verschlüsselung auf Dienste-Ebene oder Applikationsebene anbieten oder nutzen (SYS.1.6.A35).

Der Softwarelieferant KANN…

- für das persistente Speichern von Nutzdaten externe Datenbanken, persistente Volumes oder andere externe Speicher (z. B. S3) nutzen (SYS.1.6.A9).

Umsetzungshinweise:
Nutzung von ConfigMaps und Secrets.
Secrets sollen so gestaltet sein, dass sie vom Dateisystem lesbar sein
Hinweis zu (SYS.1.6.A9): Der Softwarelieferant / Softwarebetreiber braucht Informationen des Plattformbetreibers über die Möglichkeit der persistenten Speicherung von Nutzdaten.

4.1.2 Dienste und Service-Accounts

Der Softwarelieferant MUSS…

- die Anwendungssoftware gestalten, dass keine erweiterten Privilegien benötigt werden (kein "privileged"-Mode für den laufenden Container) (SYS.1.6.A21).
- die Services so entwickeln, dass die Serviceaccounts mit den geringst möglichen Berechtigungen betrieben werden können (SYS.1.6.A25).
- die Software so gestalten, dass die genutzten User- und Group-ID's keine oder nur minimale Berechtigungen auf die System- und Datenbereiche des Hosts erfordern (SYS.1.6.A26).

Der Softwarelieferant SOLL…

- die Software so gestalten, dass jeder Container nur einen Dienst bereitstellt (SYS.1.6.A12).

- sicherstellen, dass zusammenhängende Dienste durch das Deployment bzw. die Orchestrierung mehrerer Container als Gruppe bereitgestellt werden können und geeignete Deployment-Pakete liefern (SYS.1.6.A12).
- Software so bereitstellen, dass kein Fernzugriff bei der Implementierung und beim Betrieb erforderlich ist (SYS.1.6.A23).
- Containerimages so entwickeln, dass sie mit einer möglichst geringen Anzahl von Serviceaccounts unter Wahrung des Least-Privilage-Ansatzes betrieben werden können (SYS.1.6.A25).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A21): Es muss eine beliebige hohe UID genutzt werden können, welche dem Container beim Starten übergeben werden muss. Siehe auch https://docs.openshift.com/container-platform/4.2/openshift_images/create-images.html , Punkt "Support arbitrary user ids"
Hinweis zu (SYS.1.6.A23): Software zum administrativen Zugriff wie zum Beispiel ssh, telnet sollten in den Images nicht vorhanden und auch nicht erforderlich sein.

4.1.3 Monitoring Betriebsstatus

Der Softwarelieferant SOLL…

- 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. (SYS.1.6.A27).

Der Softwarelieferant KANN…

- Start-up-Checks für den Start der Container implementieren (SYS.1.6.A27).

4.1.4 Basisimages

Der Softwarelieferant SOLL[1]…

- Basisimages ausschließlich aus der vom Plattformbetreiber bereitgestellten Registry oder aus einer durch den Plattformbetreiber genehmigten vertrauenswürdigen Registry (z.B. https://catalog.hersteller.com oder "basisimages.alle-bundesländer.de) beziehen (SYS.1.6.A30).

Umsetzungshinweise:
Hinweis bzgl. offener finaler Abstimmung
Die zu nutzenden Basisimages werden zwischen Softwarelieferanten, Softwarebetreiber und Plattformbetreiber unter Berücksichtigung eines sinnvollen Verhältnisses zwischen Sicherheit und Aufwand abgestimmt.

[1] *Ziel ist die Definition einer Vorgabe mit Klassifizierung „MUSS“*

4.1.5 Kommunikationsverbindungen

Der Softwarelieferant MUSS…

- eine Möglichkeit anbieten, so dass alle Kommunikations-Endpunkte der Komponenten des Fachverfahrens mit einem verschlüsselnden Protokoll nach Stand der Technik kommunizieren können (SYS.1.6.A36).

Der Softwarelieferant SOLL…

- eine zertifikatsbasierte Authentifizierung an allen selbst implementierten Kommunikations-Endpunkten der Komponenten des Fachverfahrens anbieten (SYS.1.6.A36).

Umsetzungshinweise:
Optimal ist die direkte Umsetzung mittels eines verschlüsselnden Protokolls in der Softwarelösung. Alternativ kann ein ergänzendes Produkt (z. B. Sidecar) im Pod genutzt werden. Bei der Gestaltung der Lösung muss die Aktualisierung der Zertifikate (möglichst automatisiert) berücksichtigt werden, z. B. Ablage in Secrets.

4.2 Konfiguration/Administration

Der Softwarelieferant MUSS…

- die Konfiguration eines Containers über Container-externe Mittel ermöglichen (SYS.1.6.A7).
- für Service-Accounts seiner Applikation ein Manifest oder eine Definitionsdatei sowohl auf Cluster-Ebene als auch auf Namespace-Ebene bereitstellen (SYS.1.6.25).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A7): Die Konfiguration eines Containers darf nicht im Container selbst erfolgen. Der Softwarebetreiber muss die Konfigurationsdaten für Container über ENV-Variablen oder Dateien auf Volumes bereitstellen sowie die Qualität der Konfiguration sicherstellen können.

4.3 Protokollierung

Der Softwarelieferant MUSS…

- alle Protokolldaten der Anwendung im Container über die Standardausgabe ausgeben (SYS.1.6.A8).
- alle sicherheitsrelevanten Ereignisse protokollieren (SYS.1.6.A8).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A8): Log-Daten MÜSSEN über STDOUT / STDERR ausgegeben werden. Die Ablage von Log-Daten DARF NICHT in persistenten Volumens erfolgen, die im Container der Anwendung gemountet sind. Die Logdaten SOLLEN unverzüglich einem Log-Agenten zugeführt und möglichst revisionssicher speicherbar sein. Die Vorgaben für sicherheitsrelevante Ereignisse SOLLEN im Sicherheitskonzept beschrieben werden.

4.4 Fehlertoleranz

Der Softwarelieferant MUSS…

- die Container des Fachverfahrens so bauen, dass diese neustartfähig sind bzw. bei einem Rescheduling automatisch neu starten und alle abhängigen Dienste ebenfalls dies unterstützen (SYS.1.6.A3).

Der Softwarelieferant SOLL…

- seine Anwendung so programmieren, dass die Bestätigung für die Datenannahme erst zurückgemeldet wird, wenn die Daten auf den externen Speicher geschrieben wurden (SYS.1.6.A9).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A3): Zielstellung: Unterstützung der Scheduling-Funktion der Plattform, z.B. müssen die Dienste in den Containern nach einem Rescheduling wieder funktionsfähig sein, ein Webserver muss automatisch seine Datenbank neu verbinden usw.
Hinweis zu (SYS.1.6.A9): Innerhalb der containerisierten Anwendung sollen keine dauerhaft benötigten Nutzdaten zwischengespeichert werden. Der Ausfall eines Containers soll keine Datenverluste verursachen oder ein vollständiges Rollback sicherstellen, z. B. durch Transaktionsorientierung.

4.5 Update- und Patchmanagement

Der Softwarelieferant MUSS…

- sicherstellen, dass Updates von Software nicht im laufenden Container installiert werden (SYS.1.6.A14).

Umsetzungshinweise:
Hinweis zu (SYS.1.6. A14): Updates von Software führen zu neuen Imageversionen, welche den Release-Prozess durchlaufen müssen. Es müssen Mechanismen wie z.B. Recreate- oder Rolling-Updates unterstützt werden.

5 DOKUMENTATION

5.1 Dokumentation des Containers

Der Softwarelieferant MUSS…

- die Möglichkeiten zur Konfiguration eines Containers beschreiben (SYS.1.6.A7).
- den Inhalt der Container ausführlich dokumentieren (SYS.1.6.A7).
- Konfigurationsoptionen für die Protokollierung beschreiben (SYS.1.6.A8).
- die Möglichkeiten und Funktionsweisen zur Skalierung der Dienste beschreiben (SYS.1.6.A16).

Der Softwarelieferant SOLL…

- Build-Spezifikation (bspw. Dockerfile) für Images liefern (SYS.1.6.A13).
- Der Softwarelieferant SOLL ein für seine Software erwartbares Verhalten (Systemaufrufe, Kommunikationsbeziehungen usw.) definieren, beschreiben und dem Softwarebetreiber vorlegen (SYS.1.6.A32).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A7): Inhalt = enthaltene Basisimages, Softwarekomponenten, Frameworks, Bibliotheken, Secrets, Besonderheiten, benötigte Berechtigungen.
Hinweis zu (SYS.1.6.A8): Der Softwarebetreiber soll in die Lage versetzt werden, die Protokollierung einzurichten. Dokumentierte Konfigurationsoptionen durch den Softwarelieferanten ermöglichen dies dem Softwarebetreiber.
Hinweis zu (SYS.1.6.A16): Skalierung der Dienste: 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.
Hinweis zu (SYS.1.6.A13):
- Statische Analyse auf Schwachstellen und Schadsoftware ermöglichen
- Patchen von Basisimages durch Betreiber ermöglichen

5.2 Speicher und/oder Datenbanken

Der Softwarelieferant MUSS…

- Anforderungen an die zum Betrieb erforderlichen persistenten Speicher oder Datenbanken beschreiben (SYS.1.6.A9).
- Mindestanforderungen (Requests) und Begrenzungen (Limits) für CPU, RAM sowie temporären Speicher für jeden Container definieren. Er muss ebenfalls die Anforderungen und die Größe des benötigten persistenten Speichers definiert und die Informationen dem Softwarebetreiber mitgeteilt werden. (SYS.1.6.A16).
- notwendige persistente Volumen und deren Nutzung (RW/RO) im Deployment beschreiben (SYS.1.6.A33).

Umsetzungshinweise:
Die Beschreibung im Kontext fachlicher Anforderungen kann nur individuell erfolgen.

Tatsächliche Anforderungen und Begrenzungen werden in Abstimmung von Softwarelieferant und Softwarebetreiber erarbeitet.
Hinweis zu (SYS.1.6.A9): Insbesondere Geschwindigkeit, Zugriffstyp (RWO, ROM, RWX) sowie unterstützte Datenbanksysteme (z.B. DB2, Oracle, usw.)
Hinweis zu (SYS.1.6.A16): Nach dem Prinzip pro Container einen Dienst. Deckt mit ab: will in der Lage sein, Resourcen zu begrenzen. Ggf., je nach Szenario feingranular pro Dienst
Unter "flüchtigen Speicher" wird "ephemeral storage" verstanden, dieser ist limitierbar: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#setting-requests-and-limits-for-local-ephemeral-storage You can use ephemeral-storage for managing local ephemeral storage. Each Container of a Pod can specify one or more of the following:

- spec.containers[].resources.limits.ephemeral-storage

- spec.containers[].resources.requests.ephemeral-storage

- Requests und Limits gibt es nicht nur auf Container-Ebene, sondern auch auf Pod-Ebene

5.3 Konnektierte Netze

Der Softwarelieferant MUSS…

- alle erforderlichen Kommunikationsverbindungen wenn möglich inklusive Zielhosts, Kommunikationsprotokolle und Abhängigkeiten (z. B. spezielle oder feste Ports) beschreiben, die für Inbetriebnahme und Betrieb erforderlich sind (SYS.1.6.A18).
- die eingehenden und ausgehenden Kommunikationsbeziehungen zu anderen Systemen beschreiben und für den Softwarebetreiber offenlegen (SYS.1.6.A19).
- die notwendigen Kommunikationsbeziehungen zwischen den Komponenten der Anwendung (z.B. Containern und Pods), sowie mit Komponenten außerhalb der Container-Plattform in geeigneter Weise deklarieren (SYS.1.6.A33).

Der Softwarelieferant SOLL…

- Anforderungen (Requests) und Begrenzungen (Limits) für das Netzwerk definieren und dem Softwarebetreiber mitteilen (SYS.1.6.A16).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A33): Deklaration meint hier neben einer Dokumentation der Beziehungen auch eine Definition bspw. als Network-Policy.

5.4 Berechtigungen

Der Softwarelieferant MUSS…

- begründen, für welche Zwecke technische User (z.B.: Kubernetes ServiceAccounts) Berechtigungen erfordern, welche über die Standardberechtigungen hinausgehen (SYS.1.6.A24).
- die erforderlichen User- und Group-ID's und deren Berechtigungen benennen (z.B. in den Konfigurationsdateien) (SYS.1.6.A26).

5.5 Lizenzen

Der Softwarelieferant MUSS…

- eine vollständige Liste erforderlicher Lizenzen angeben (SYS.1.6.A13).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A13): Es muss eine Prüfung der Lizenzverwendung ermöglicht werden.

6 LIEFERUNG

Der Softwarelieferant MUSS…

- Die Images, Deploymentpakete und Build-Spezifikationen über einen sicheren Weg an das abgestimmte Repository anliefern oder zur Abholung bereitstellen (SYS.1.6.A6).
- bei sicherheitsrelevanten und featurebasierten Updates der Anwendungen neue Images erstellen und eindeutig versioniert innerhalb der vereinbarten Zeiträume / SLAs zur Verfügung stellen (SYS.1.6.A14).
- Images dem Softwarebetreiber zugänglich machen und über eine durch den Plattformbetreiber genehmigte vertrauenswürdige Registry bereitstellen. Der schreibende Zugang muss authentisiert erfolgen (SYS.1.6.A30).

Der Softwarelieferant SOLL…

- bei Feststellung einer Verwundbarkeit ein neues Image in einer vereinbarten Zeit (z.B. SLA) mit entsprechenden Updates zur Verfügung stellen (SYS.1.6.A29).
- ein neues Image ohne Schwachstellen bereitstellen, sofern Schwachstellen im durch ihn erstellten Image erkannte Angriffe ermöglicht haben (SYS.1.6.A32).

Der Softwarelieferant KANN…

- die zu liefernden Images bereits vor der finalen Lieferung mit dem Prüfstandard des Softwarebetreibers überprüfen (SYS.1.6.A14).

Umsetzungshinweise:
Hinweis zu (SYS.1.6.A14): 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.

ÄNDERUNGSKONTROLLE

Version	Datum	Bearbeiter	Status, Änderungsgrund	Seiten
1.0	17.06.2021		Erstversion	Alle