BSI Grundschutz für Container und Kubernetes in KRITIS Umgebungen

Zusammenfassung von 10 Jahren Erfahrungen in Kubernetes Sicherheitsworkshops

Thomas Fricke

16. August 2026
FROSCON

APP.4.4 Kubernetes

- Kubernetes De-Facto-Standard
- Orchestrierung von Containern
- Public und Private Cloud
- IoT und andere Anwendungsfälle

1.2. Zielsetzung

Ziel dieses Bausteins ist der Schutz von Informationen, die in Kubernetes-Clustern verarbeitet, angeboten oder darüber übertragen werden.

1.3.Abgrenzung und Modellierung

Der Baustein APP.4.4 *Kubernetes* ist immer zusammen mit dem Baustein SYS.1.6 *Containerisierung* anzuwenden. Dabei ist es bezogen auf den Fokus des vorliegenden Bausteins nicht relevant, welche Container-Runtime im Einsatz ist oder welche zusätzlichen Anwendungen Teil der Control Plane sind.

Grundsätzliche Anforderungen

Der Baustein enthält grundsätzliche Anforderungen zur Einrichtung, zum Betrieb und zur Orchestrierung mit Kubernetes sowie zur spezialisierten Infrastruktur, die zum Betrieb notwendig ist. Letzteres beinhaltet Registries, CSI/CNI, Nodes und Automatisierungssoftware, soweit sie direkt mit dem Cluster interagieren. Die Anforderungen für diese Anwendungen beziehen sich zumeist auf die Schnittstellen, enthalten aber auch Anforderungen, die den Betrieb dieser Anwendungen betreffen, sofern sie direkt die Sicherheit des Clusters berühren. Weitere im Kubernetes-Umfeld übliche Dienste, wie z.B. Automatisierung für CI/CD-Pipelines und Codeverwaltung in z.B. Git, behandelt der Baustein nicht in der Tiefe. > Der Baustein modelliert umfassend einen Cluster. Die Anwendungen der Control Plane, Dienste zur Automatisierung und die Nodes, sind hier als eine Gruppe zu sehen und zu behandeln. > Sicherheitsanforderungen für die in Kubernetes-Clustern betriebenen Dienste, wie z.B. Webserver (APP.3.2 *Webserver*) oder E-Mail-Server (siehe APP.5.3 *Allgemeiner E-Mail-Client und -Server*), sind Gegenstand eigener Bausteine.

2.Gefährdungslage

Da IT-Grundschutz-Bausteine nicht auf individuelle Informationsverbünde eingehen können, werden zur Darstellung der Gefährdungslage typische Szenarien zugrunde gelegt. Die folgenden spezifischen Bedrohungen und Schwachstellen sind für den Baustein APP.4.4 *Kubernetes* von besonderer Bedeutung.

2.1.Mangelhafte Authentisierung und Autorisierung in der Control Plane

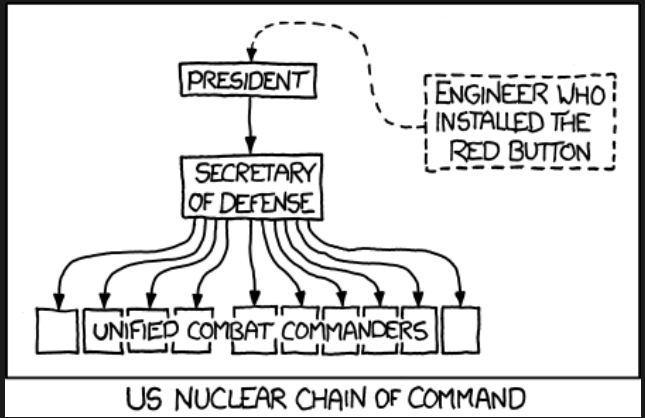
Um Runtimes, Nodes und auch Kubernetes selbst zu verwalten, benötigen die Administratoren und die toolgestützte Bereitstellung administrative Zugänge. Diese Zugänge sind entweder als Unix-Sockets oder Netzports ausgeführt. Mechanismen zur Authentisierung und Verschlüsselung der
administrativen Zugänge sind häufig vorhanden, aber nicht bei allen Produkten standardmäßig aktiviert.

Wenn Unbefugte auf das Datennetz oder auf die Nodes zugreifen, können sie über ungeschützte administrative Zugänge Befehle ausführen, die der Verfügbarkeit, Vertraulichkeit und Integrität der verarbeiteten Daten schaden können.

Kommentar zu 2.1: viele ControlPlanes

- ArgoCD ..
- Kubernetes
- Netzwerk
- Storage
- Loadbalancer
- Secrets
- Cluster

Besser: Controlplanes identifizieren und definieren



2.2. Vertraulichkeitsverlust von Zugangsdaten

Oft benötigen Pods Zugangsdaten (Access Token) für Kubernetes. Über einen Angriff auf den Pod können diese Zugangsdaten in unbefugte Hände gelangen. Mit diesen Zugangsdaten ist es dem Angreifer möglich, mit der Control Plane authentifiziert zu interagieren und, sofern die Berechtigungen ausreichen, auch Veränderungen an der Orchestrierung vorzunehmen.

Besser:

- Nachweisen durch GitOPS: kein Zugriff auf Passworte
- RBAC auf Kubernetes
  - Secrets
  - Pod/Exec

2.3.Ressourcenkonflikte auf Nodes

Einzelne Pods können den Node oder auch die Orchestrierung überlasten und so die Verfügbarkeit aller anderen Pods auf dem Node oder auch den Betrieb des Nodes selbst gefährden.

Limits setzen auf

- CPU
- Memory
- Network

2.4.Unautorisierte Änderungen an Clustern

Die Automatisierung mit CI/CD und die daraus folgende Notwendigkeit, den Werkzeugen privilegierte Zugangsberechtigungen zu erteilen, birgt das Risiko, dass nicht autorisierte Änderungen an Clustern erfolgen. So kann z.B. ein Entwickler eine neue Version einer Anwendung auf dem Cluster aufbringen, die nicht ausreichend getestet ist oder die den Freigabeprozess nicht durchlaufen hat. Auch ist es bei Fehlern in den Berechtigungen auf der CI/CD-Umgebung möglich, dass Schadsoftware in die Cluster eindringen und dort Daten auslesen, löschen oder verändern kann.

Keine Diskussionen wenn GitOps

2.5.Unberechtigte Kommunikation

Alle Pods in einem Cluster sind grundsätzlich in der Lage, miteinander, mit den Nodes im eigenen Cluster sowie beliebigen anderen IT-Systemen zu kommunizieren. Sofern diese Kommunikation nicht eingeschränkt ist, kann eine Schadsoftware oder ein Angreifer dies ausnutzen, um z.B. die Control Plane, andere Pods oder die Nodes anzugreifen.

Auch besteht die Gefahr, dass Pods im Cluster unerwünscht von außen erreichbar sind. So kann ein Angriff gegen Dienste, die eigentlich nur innerhalb des Clusters erreichbar sein sollten, von außen erfolgen. Diese Gefährdung wird durch die geringere Aufmerksamkeit verschlimmert, die internen Diensten oft entgegen gebracht wird. Wird z.B. eine Verwundbarkeit auf einem nur intern eingesetzten Dienst toleriert und ist dieser auch von außen erreichbar, gefährdet dies den gesamten Cluster erheblich.

Massnahmen gegen 2.5.Unberechtigte Kommunikation

- NetworkPolicies
- Istio
- Nicht herumliegenlassen
  - Tools (curl, wget)
  - ServiceAccountToken
  - !!! Auch nicht das Token des Cloud Accounts !!!

3.Anforderungen

Im Folgenden sind die spezifischen Anforderungen des Bausteins APP.4.4 *Kubernetes* aufgeführt. Der ISB ist dafür zuständig, dass alle Anforderungen gemäß dem festgelegten Sicherheitskonzept erfüllt und überprüft werden. Bei strategischen Entscheidungen ist der ISB stets einzubeziehen.

Im IT-Grundschutz-Kompendium sind darüber hinaus weitere Rollen definiert. Sie sollten besetzt werden, insofern dies sinnvoll und angemessen ist.

Zuständigkeiten	**Rollen**
Grundsätzlich zuständig	IT-Betrieb
Weitere Zuständigkeiten	keine

Genau eine Rolle sollte *Grundsätzlich zuständig* sein.

Darüber hinaus kann es noch *Weitere Zuständigkeiten* geben. Falls eine dieser weiteren Rollen für die Erfüllung einer Anforderung vorrangig zuständig ist, dann wird diese Rolle hinter der Überschrift der Anforderung in eckigen Klammern aufgeführt.

3.1.Basis-Anforderungen

Die folgenden Anforderungen MÜSSEN für diesen Baustein vorrangig erfüllt werden.

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.

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

Voraussetzung

- Threat Analyse
- Welche bekannten/unbekannten Anwendungen habe ich im Cluster
- Schutzbedarf

Daraus leitet sich Separierung ab

- Netz
- Nodes
- Hypervisor
- Unikernel

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.

Eigenes, grosses Thema!

- Softwarelieferkette
- SBOMs
- Übergabepunkte (Harbor Registries)
  - Containerscanner
  - Vulnerability Reports
  - SIEM
- **Zulieferer**

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.

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

- Konzept
- s.o. Controlplane
- Kubescape scannt das!

  inclusive Azure Cloud Account Token!

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.

- Linux Container
- Windows Container

  WSL

  OpenShift mit virtuellen Windows Nodes ???

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.

APP.4.4.A5 Datensicherung im Cluster (B)

Wie immer

- Festspeicher (Persistent Volumes)
- Datenbanken
- alle Infrastrukturanwendungen

  die zum Betrieb des Clusters und der darin befindlichen Dienste notwendig sind und die Datenhaltung der Code und Image Registries.

GitOps regelt das

- 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*

3.2.Standard-Anforderungen

Gemeinsam mit den Basis-Anforderungen entsprechen die folgenden Anforderungen dem Stand der Technik für diesen Baustein. Sie SOLLTEN grundsätzlich erfüllt werden.

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.

Das ist tatsächlich der Fall.

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.

Geht am Anfang immer schief!

- Network Control Plane
- Network Policies
- Service Meshs

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.

GITOPS!

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.

- Planung!
- Kubescape scannt das

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.

Here be large Dragons!

- eigenes grosses Thema!
- Planung!
- Kubescape scannt das

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.

- Eingebaut, einfach nutzen!
- Health und Live nicht mischen
- Abhängigkeiten von Datenbanken berücksichtigen
- nicht die falsche Applikation killen
- Caches!

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.

Harbour

- Vulnerability Scanner
- Reports an Monitoring
- **nicht in meine Email**

3.3.Anforderungen bei erhöhtem Schutzbedarf

Im Folgenden sind für diesen Baustein exemplarische Vorschläge für Anforderungen aufgeführt, die über dasjenige Schutzniveau hinausgehen, das dem Stand der Technik entspricht. Die Vorschläge SOLLTEN bei erhöhtem Schutzbedarf in Betracht gezogen werden. Die konkrete Festlegung erfolgt im Rahmen einer individuellen Risikoanalyse.

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.

- Kubescape
- Kubebench
- Werbung: Mitigant

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.

- Bastion Nodes könnten die Performance beinträchtigen
- Management Nodes
- Storage extern
  - Rados
  - Ceph

Nur bei hohen Datenbank Anforderungen lokaler Storage

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.

- Kleine Cluster sind ineffektiv
- Edge Cluster

  K0s

  K3s

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.

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.

- Trusted Boot
- Immutable Images
- Spezielle Distributionen

  Anthos

  K0s

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.

- Netzwerkkonzept

  NetworkPolicies

  Service Meshs

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.

- Grosses Thema: Effektivität
- Cold/Hot Standby
- 3 Zonen/Region
- Sync muss getestet werden
- Architektur von OpenDesk

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.

Backup von etcd

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.

- 24h?
- andere runtimes

  virtualisierung

  ...

4.Weiterführende Informationen

4.1.Wissenswertes

Weiterführende Informationen zu Gefährdungen und Sicherheitsmaßnahmen im Bereich Container finden sich unter anderem in folgenden Veröffentlichungen:

- NIST 800-190
- CIS Benchmark Kubernetes
- CNCF – Cloud Native Computing Foundation

5.Anlage: Kreuzreferenztabelle zu elementaren Gefährdungen

Aus jeder Anforderung (A) in diesem Baustein können Sicherheitsmaßnahmen abgeleitet werden. Die Umsetzung dieser Maßnahmen wirkt denjenigen elementaren Gefährdungen (G0) entgegen, die für das Thema bzw. Zielobjekt relevant sind. In der Kreuzreferenztabelle (KRT) zu diesem Baustein sind jeder Anforderung die entsprechenden elementaren Gefährdungen zugeordnet.

Anhand der KRT lässt sich ermitteln, welche elementaren Gefährdungen durch welche Anforderungen abgedeckt sind. Die Buchstaben in der zweiten Spalte zeigen an, welche Grundwerte der
Informationssicherheit durch die Anforderung vorrangig geschützt werden. Diese Grundwerte sind Confidentiality (C) für Vertraulichkeit, Integrity (I) für Integrität sowie Availability (A) für Verfügbarkeit.

Elementaren Gefährdungen

Die folgenden elementaren Gefährdungen sind für den Baustein APP.4.4 *Kubernetes* von Bedeutung:

- G 0.14 Ausspähen von Informationen Spionage
- G 0.19 Offenlegung schützenswerter Informationen
- G 0.20 Informationen oder Produkte aus unzuverlässiger Quelle
- G 0.21 Manipulation von Hard- oder Software
- G 0.23 Unbefugtes Eindringen in IT-Systeme
- G 0.25 Ausfall von Geräten oder Systemen
- G 0.27 Ressourcenmangel
- G 0.28 Software-Schwachstellen oder -Fehler
- G 0.30 Unberechtigte Nutzung oder Administration von Geräten und Systemen
- G 0.37 Abstreiten von Handlungen
- G 0.39 Schadprogramme
- APP.4.4 Kubernetes G 0.45 Datenverlust
- G 0.46 Integritätsverlust schützenswerter Informationen

2.1.Schwachstellen oder Schadsoftware in Images

Container werden vorrangig auf Basis von vorgefertigten Images erstellt, die selbst erstellt, aber auch häufig aus dem Internet bezogen werden. Des Weiteren wird Software durch Hersteller zunehmend in Form von Images ausgeliefert. Diese Images kann der IT-Betrieb auch für die Erstellung seiner eigenen Images nutzen, indem er Software oder Konfigurationen ergänzt, verändert oder auch entfernt.

Die in den Images enthaltene Software könnte verwundbar und die aus dem Image gestarteten Container könnten dadurch angreifbar sein. Solche Schwachstellen können dem IT-Betrieb auch häufig nicht bekannt sein, da die in den Images enthaltene Software oft nicht in der eigenen Software-Verwaltung erfasst ist. Der IT-Betrieb muss sich grundsätzlich darauf verlassen, dass Updates über den Prozess der Image-Erstellung verfügbar gemacht werden. Von außen betrachtet, ist häufig nur aufwendig zu ermitteln, welche Software-Pakete in diesen Images vorhanden sind.

Zudem könnten Images absichtlich integrierte Schadsoftware enthalten, wie z. B. Ransomware oder Kryptominer. Da ein einzelnes Image oft in einer großen Anzahl von Containern ausgebracht wird, kann der resultierende Schaden immens sein.

siehe Kubernetes

- **2.2.Administrative Zugänge ohne Absicherung**
- **2.3.Ressourcenkonflikte auf dem Host**
- **2.4.Unberechtigte Kommunikation**

2.5.Ausbruch aus dem Container auf das Host-System

Sollte ein Angreifer in der Lage sein, im Container eigenen Code auszuführen, kann er möglicherweise die Isolation des Containers gegenüber anderen Containern bzw. dem Host überwinden und auf andere Container, das Host-System oder die Infrastruktur zugreifen. Es wird auch von einem „Container-Ausbruch" gesprochen. Dieser Angriff kann z. B. über Schwachstellen in Prozessoren, im Betriebssystem-Kernel oder in lokal angebotenen Infrastruktur-Diensten wie z. B. DNS oder SSH erfolgen.

Somit könnte ein Angreifer die Kontrolle über das Host-System oder andere Systeme aus der Infrastruktur übernehmen. Es droht der Verlust der Vertraulichkeit, Integrität und Verfügbarkeit aller auf diesem Host ausgeführten Container sowie auf dem Host selbst, falls der Angreifer dort ebenfalls erhöhte Privilegien erlangen kann.

Details

Container Breakouts

- Docker Socket im Container gemounted

  Geht auch mit Containerd

  Hyped AI

  Gefährliches Halbwissen

- Kernel

  Copy Fail

  Einzige Abhilfe Kernel Update ASAP

  K0S !

2.6.Datenverluste durch das Container-Management

Im Rahmen der Verwaltung von Containern können diese ausgeschaltet werden, ohne darin gerade ausgeführter Software die Gelegenheit zu geben, z. B. aktuelle Schreibprozesse abzuschließen (nicht ordnungsgemäßes Herunterfahren). Sollten zu diesen Zeitpunkten Daten durch den Container verarbeitet werden, sind alle diese Daten verloren. Auch Daten, die persistent im Container gespeichert sind, können so dauerhaft verloren gehen.

- Nichts besonderes
- herunterfahren

2.7. Vertraulichkeitsverlust von Zugangsdaten

Die Prinzipien des Aufbaus und der Erstellung von Images für Container setzen voraus, dass Zugangsdaten, z. B. für Datenbanken, im Container verfügbar sind. Über die Images selbst, die Skripte zur Erstellung der Images oder die Versionskontrolle der Skripte können solche Zugangsdaten in unbefugte Hände gelangen.

Oft werden Zugangsdaten auch zum Erstellungszeitpunkt des Containers als Umgebungsvariable verfügbar gemacht. Hier droht ebenfalls der Vertraulichkeitsverlust dieser Daten.

- Saubere Trennung von und Secrets Applikationen
- wie in jedem Config Management

2.8.Ungeordnete Bereitstellung und Verteilung von Images

Im Gegensatz zu klassischen Installationen durch den IT-Betrieb, wo die Kontrolle über die ausgebrachten Anwendungen, Komponenten und Diensten vollständig beim IT-Betrieb selbst liegt, geht diese bei z. B. bei Automatisierung durch CI/CD in Container-Umgebungen verloren. Vielmehr stellt der IT-Betrieb nur eine Plattform bereit, in die Entwickler direkt ihre Anwendungen inklusive sämtlicher Abhängigkeiten einbringen und jederzeit verändern

- wird gerne übersehen
- benötigt viel Disk Space!
- aufbewahren mit definierten Fristen

3.Anforderungen

- **3.1.Basis-Anforderungen** s.o
- **SYS.1.6.A1 Planung des Container-Einsatzes (B)**

  Bevor Container eingesetzt werden, MUSS zunächst das Ziel des Container-Einsatzes (z. B. Skalierung, Verfügbarkeit, Wegwerf-Container zur Sicherheit oder CI/CD) festgelegt werden, damit alle sicherheitsrelevanten Aspekte der Installation, des Betriebs und der Außerbetriebnahme geplant werden können. Bei der Planung SOLLTE auch der Betriebsaufwand berücksichtigt werden, der durch Container-Einsatz oder Mischbetrieb entsteht. Die Planung MUSS angemessen dokumentiert werden.

- Lifecyle
- Aufwand
- Vermeidet Mischbetrieb

SYS.1.6.A2 Planung der Verwaltung von Containern (B)

Die Verwaltung der Container DARF NUR nach einer geeigneten Planung erfolgen. Diese Planung MUSS den gesamten Lebenszyklus von Inbetrieb- bis Außerbetriebnahme inklusive Betrieb und Updates umfassen. Bei der Planung der Verwaltung MUSS berücksichtigt werden, dass der Ersteller eines Containers aufgrund der Auswirkungen auf den Betrieb in Teilen wie ein Administrator zu betrachten ist.

Start, Stopp und Überwachung der Container MUSS über die eingesetzte Verwaltungssoftware erfolgen.

- System Container mit Privilegien

  oft sehr schlampig
- Anwendungen brauchen sehr selten Privilegien

wie Kubernetes

- **SYS.1.6.A3 Sicherer Einsatz containerisierter IT-Systeme (B)**
- **SYS.1.6.A4 Planung der Bereitstellung und Verteilung von Images (B)**

  Der Prozess zur Bereitstellung und Verteilung von Images MUSS geplant und angemessen dokumentiert werden.
- **SYS.1.6.A5 Separierung der Administrations- und Zugangsnetze bei Containern (B)**

  ... Es SOLLTEN nur die für den Betrieb notwendigen Kommunikationsbeziehungen erlaubt werden.

SYS.1.6.A6 Verwendung sicherer Images (B)

Es MUSS sichergestellt sein, dass sämtliche verwendeten Images nur aus vertrauenswürdigen Quellen stammen. Der Ersteller der Images MUSS eindeutig identifizierbar sein.

Die Quelle MUSS danach ausgewählt werden, dass der Ersteller des Images die enthaltene Software regelmäßig auf Sicherheitsprobleme prüft, diese behebt und dokumentiert sowie dies seinen Kunden zusichert.

Die verwendete Version von Basis-Images DARF NICHT abgekündigt („deprecated") sein. Es MÜSSEN eindeutige Versionsnummern angegeben sein. Wenn ein Image mit einer neueren Versionsnummer verfügbar ist, MUSS im Rahmen des Patch- und Änderungsmanagement geprüft werden, ob und wie dieses ausgerollt werden kann.

Problemfeld

Problemfeld

- Softwareentwicklung

  viele Diskussionen mit Teams und Zulieferern

- Minimale Container

  sind manchmal nicht minimal (UBI)

- Zendis

  Gehärtete Container-Images für die Öffentliche Verwaltung: Secure Government Container Initiative (SGCI)

- selber härten

  Mein Container Hardening

  geht auch automatisch mit eBPF, aber das braucht noch Sponsoring

SYS.1.6.A7 Persistenz von Protokollierungsdaten der Container (B)

Die Speicherung der Protokollierungsdaten der Container MUSS außerhalb des Containers, mindestens auf dem Container-Host, erfolgen.

- ordentliches Logging

  Elastic, OpenSearch, Splunk

- BAFIN Anforderung

SYS.1.6.A8 Sichere Speicherung von Zugangsdaten bei Containern (B)

Zugangsdaten MÜSSEN so gespeichert und verwaltet werden, dass nur berechtigte Personen und Container darauf zugreifen können. Insbesondere MUSS sichergestellt sein, dass Zugangsdaten nur an besonders geschützten Orten und nicht in den Images liegen. Die von der Verwaltungssoftware des Container-Dienstes bereitgestellten Verwaltungsmechanismen für Zugangsdaten SOLLTEN eingesetzt werden.

Mindestens die folgenden Zugangsdaten MÜSSEN sicher gespeichert werden:

- Passwörter jeglicher Accounts,
- API-Keys für von der Anwendung genutzte Dienste,
- Schlüssel für symmetrische Verschlüsselungen sowie
- private Schlüssel bei Public-Key-Authentisierung.

siehe Kubernetes

3.2.Standard-Anforderungen

Gemeinsam mit den Basis-Anforderungen entsprechen die folgenden Anforderungen dem Stand der Technik für diesen Baustein. Sie SOLLTEN grundsätzlich erfüllt werden.

SYS.1.6.A9 Eignung für Container-Betrieb (S)

Die Anwendung bzw. der Dienst, der im Container betrieben werden soll, SOLLTE für den Container-Betrieb geeignet sein. Dabei SOLLTE berücksichtigt werden, dass Container häufiger für die darin ausgeführte Anwendung unvorhergesehen beendet werden können. Die Ergebnisse der Prüfung nach SYS.1.6.A3 *Sicherer Einsatz containerisierter IT-Systeme* SOLLTE nachvollziehbar dokumentiert werden.

- Fast alles ist geeignet
- Probleme macht Posix Filesystem

  Mailer (OpenExchange)

SYS.1.6.A10 Richtlinie für Images und Container-Betrieb (S)

Es SOLLTE eine Richtlinie erstellt und angewendet werden, die die Anforderungen an den Betrieb der Container und die erlaubten Images festlegt. Die Richtlinie SOLLTE auch Anforderungen an den Betrieb und die Bereitstellung der Images enthalten.

SYS.1.6.A11 Nur ein Dienst pro Container (S)

Jeder Container SOLLTE jeweils nur einen Dienst bereitstellen.

SYS.1.6.A12 Verteilung sicherer Images (S)

Es SOLLTE angemessen dokumentiert werden, welche Quellen für Images als vertrauenswürdig klassifiziert wurden und warum. Zusätzlich SOLLTE der Prozess angemessen dokumentiert werden, wie Images bzw. die im Image enthaltenen Softwarebestandteile aus vertrauenswürdigen Quellen bezogen und schließlich für den produktiven Betrieb bereitgestellt werden.

Die verwendeten Images SOLLTEN über Metadaten verfügen, die die Funktion und die Historie des Images nachvollziehbar machen. Digitale Signaturen SOLLTEN jedes Image gegen Veränderung absichern.

SYS.1.6.A13 Freigabe von Images (S)

Alle Images für den produktiven Betrieb SOLLTEN wie Softwareprodukte einen Test- und Freigabeprozess gemäß des Bausteins OPS.1.1.6 *Software-Test und Freigaben* durchlaufen.

SYS.1.6.A14 Aktualisierung von Images (S)

Bei der Erstellung des Konzeptes für das Patch- und Änderungsmanagement gemäß OPS.1.1.3 *Patch- und Änderungsmanagement* SOLLTE entschieden werden, wann und wie die Updates der Images oder der betriebenen Software bzw. des betriebenen Dienstes ausgerollt werden. Bei persistenten Containern SOLLTE geprüft werden, ob in Ausnahmefällen ein Update des jeweiligen Containers geeigneter ist, als den Container vollständig neu zu provisionieren.

SYS.1.6.A15 Limitierung der Ressourcen pro Container (S)

Für jeden Container SOLLTEN Ressourcen auf dem Host-System, wie CPU, flüchtiger und persistenter Speicher sowie Netzbandbreite, angemessen reserviert und limitiert werden. Es SOLLTE definiert und dokumentiert sein, wie das System im Fall einer Überschreitung dieser Limitierungen reagiert.

SYS.1.6.A16 Administrativer Fernzugriff auf Container (S)

Administrative Zugriffe von einem Container auf den Container-Host und umgekehrt SOLLTEN prinzipiell wie administrative Fernzugriffe betrachtet werden. Aus einem Container SOLLTEN KEINE administrativen Fernzugriffe auf den Container-Host erfolgen. Applikations-Container SOLLTEN keine Fernwartungszugänge enthalten. Administrative Zugriffe auf Applikations-Container SOLLTEN immer über die Container-Runtime erfolgen.

- Kein SSH

  ansible
- Administrations Container sind hoch privilegiert

SYS.1.6.A17 Ausführung von Containern ohne Privilegien (S)

Die Container-Runtime und alle instanziierten Container SOLLTEN nur von einem nicht-privilegierten System-Account ausgeführt werden, der keine erweiterten Rechte für den Container-Dienst bzw. das Betriebssystem des Host-Systems verfügt oder diese Rechte erlangen kann. Die Container-Runtime SOLLTE durch zusätzliche Maßnahmen gekapselt werden, etwa durch Verwendung der Virtualisierungs-Erweiterungen von CPUs.

Sofern Container ausnahmsweise Aufgaben des Host-Systems übernehmen sollen, SOLLTEN die Privilegien auf dem Host-System auf das erforderliche Minimum begrenzt werden. Ausnahmen SOLLTEN angemessen dokumentiert werden.

Schlimmes Gegenbeispiel von

Hacking OpenShift

SYS.1.6.A18 Accounts der Anwendungsdienste (S)

Die System-Accounts innerhalb eines Containers SOLLTEN keine Berechtigungen auf dem Host-System haben. Wo aus betrieblichen Gründen diese Berechtigung notwendig ist, SOLLTE diese nur für unbedingt notwendige Daten und Systemzugriffe gelten. Der Account im Container, der für diesen Datenaustausch notwendig ist, SOLLTE im Host-System bekannt sein.

SYS.1.6.A19 Einbinden von Datenspeichern in Container (S)

Die Container SOLLTEN NUR auf die für den Betrieb notwendigen Massenspeicher und Verzeichnisse zugreifen können. Nur wenn Berechtigungen benötigt werden, SOLLTEN diese explizit vergeben werden. Sofern die Container-Runtime für einen Container lokalen Speicher einbindet, SOLLTEN die Zugriffsrechte im Dateisystem auf den Service-Account des Containers eingeschränkt sein. Werden Netzspeicher verwendet, so SOLLTEN die Berechtigungen auf dem Netzspeicher selbst gesetzt werden.

SYS.1.6.A20 Absicherung von Konfigurationsdaten (S)

Die Beschreibung der Container-Konfigurationsdaten SOLLTE versioniert erfolgen. Änderungen SOLLTEN nachvollziehbar dokumentiert sein.

GITOPS!

3.3.Anforderungen bei erhöhtem Schutzbedarf

Im Folgenden sind für diesen Baustein exemplarische Vorschläge für Anforderungen aufgeführt, die über dasjenige Schutzniveau hinausgehen, das dem Stand der Technik entspricht. Die Vorschläge SOLLTEN bei erhöhtem Schutzbedarf in Betracht gezogen werden. Die konkrete Festlegung erfolgt im Rahmen einer individuellen Risikoanalyse.

SYS.1.6.A21 Erweiterte Sicherheitsrichtlinien (H)

Erweiterte Richtlinien SOLLTEN die Berechtigungen der Container einschränken. Mandatory Access Control (MAC) oder eine vergleichbare Technik SOLLTE diese Richtlinien erzwingen. Die Richtlinien SOLLTEN mindestens folgende Zugriffe einschränken:

- eingehende und ausgehende Netzverbindungen,
- Dateisystem-Zugriffe und
- Kernel-Anfragen (Syscalls).

Die Runtime SOLLTE die Container so starten, dass der Kernel des Host-Systems alle nicht von der Richtlinie erlaubten Aktivitäten der Container verhindert (z. B. durch die Einrichtung lokaler Paketfilter oder durch Entzug von Berechtigungen) oder zumindest Verstöße geeignet meldet.

Applikationen, die Ihr immer behandeln solltet wie ein CVE von 10:

- ffmpeg
- clamav
- pdfutils
- alles, was nur in C geschrieben ist
- und Libs in C verwendet

SYS.1.6.A22 Vorsorge für Untersuchungen (H)

Um Container im Bedarfsfall für eine spätere Untersuchung verfügbar zu haben, SOLLTE ein Abbild des Zustands nach festgelegten Regeln erstellt werden.

SYS.1.6.A23 Unveränderlichkeit der Container (H)

Container SOLLTEN ihr Dateisystem während der Laufzeit nicht verändern können.
Dateisysteme SOLLTEN nicht mit Schreibrechten eingebunden sein.

Kein Config Management im Container

- Ansible
- Chef
- Puppet
- ...

SYS.1.6.A24 Hostbasierte Angriffserkennung (H)

Das Verhalten der Container und der darin betriebenen Anwendungen bzw. Diensten SOLLTE überwacht werden. Abweichungen von einem normalen Verhalten SOLLTEN bemerkt und gemeldet werden. Die Meldungen SOLLTEN im zentralen Prozess zur Behandlung von Sicherheitsvorfällen angemessen behandelt werden.

Das zu überwachenden Verhalten SOLLTE mindestens umfassen:

- Netzverbindungen,
- erstellte Prozesse,
- Dateisystem-Zugriffe und Kernel-Anfragen (Syscalls).
- **SYS.1.6.A25 Hochverfügbarkeit von containerisierten Anwendungen (H)** Bei hohen Verfügbarkeitsanforderungen der containerisierten Anwendungen SOLLTE entschieden werden, auf welcher Ebene die Verfügbarkeit realisiert werden soll (z. B. redundant auf der Ebene des Hosts).

SYS.1.6.A24 Hostbasierte Angriffserkennung (H)

Aufwändig

- Ebpf
- SELinux
- APPArmor

Kann von Kubernetes erzwungen werden

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

Wird eine weitergehende Isolation und Kapselung von Containern benötigt, dann SOLLTEN folgende Maßnahmen nach steigender Wirksamkeit geprüft werden:

- feste Zuordnung von Containern zu Container-Hosts,
- Ausführung der einzelnen Container und/oder des Container-Hosts mit Hypervisoren,
- feste Zuordnung eines einzelnen Containers zu einem einzelnen Container-Host.

CPU Pinning um Sidechannel Attacks zu unterbinden?

Was betreibt Ihr da? Public Cloud?

Zusammenfassung

- Alle Checks lassen sich automatisieren
  - Kubescape, Kubebench (1 Minute)
  - Grype, Trivy (1 Stunde)

    kann auch mal ein Report mit 2000 Seiten CVEs werden
- Aufwand ist der Abgleich mit Grundschutz
  - Sicherheitsstufe
  - Bedrohungsanalyse
  - Liebe zum Detail
- Workshops für die Basics
  - 5 Tage (2+3)
  - Referenzen: HPA, BSI

Fragen und ein paar Antworten

Slides: https://thomasfricke.de/froscon2026.pdf

Mail: froscon2026@thomasfricke.de

Mastodon: @thomasfricke@23.social

LinkedIn: https://www.linkedin.com/in/thomas-fricke-9840a21/