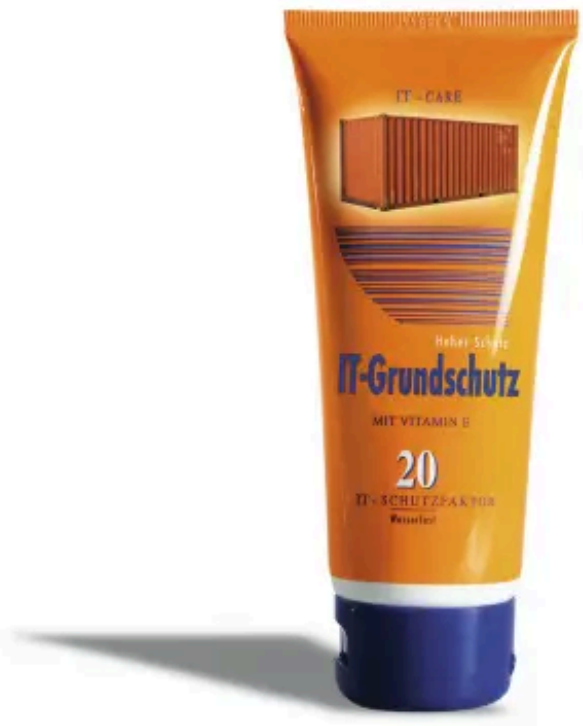
MAGAZIN FÜR PROFESSIONELLE
INFORMATIONSTECHNIK

Container-Sicherheit: Entwurf des BSI-Bausteins für das Grundschutz-Kompendium

19.11.2018 09:54 Uhr Christoph Puppe



Mit der Verbreitung von Containern gerät deren Sicherheit in den Fokus. Das BSI erweitert daher sein IT-Grundschutz-Kompendium um einen Container-Baustein.

Behörden und Firmen in Deutschland setzen Container inzwischen strategisch ein. Das BSI hat daher den Entwurf eines Bausteins für das Grundschutz-Kompendium veröffentlicht. Dieser gibt den Nutzern einen Leitfaden an die Hand, wie sie Container sicher und zuverlässig einsetzen. Nutzer des BSI-Kompendiums können den Entwurf kommentieren, bevor ihn die nächste Version aufnimmt und er für Zertifizierungen relevant wird.

Wie alle Bausteine seit der **Novellierung des IT-Grundschutzes [1]** im August 2017 gliedert sich dieser Baustein in die Stufen Basis, Standard und Erhöht. Er enthält auf jeder Stufe Anforderungen, die das BSI für den Betrieb von Containern für notwendig erachtet. Zu jedem Baustein gehört außerdem eine Kreuzreferenz, die zeigt, welche Anforderung welches Risiko abdeckt. Zu Beginn einer Zertifizierung im Rahmen des IT-Grundschutzes etwa steht eine Risikoanalyse, aus der sich ergibt, welche Anforderungen zu erfüllen sind.

IX-TRACT

● Das IT-Grundschutz-Kompendium des BSI enthält Anforderungen an die Sicherheit von IT-Komponenten. Bundesbehörden müssen diese einhalten – und somit auch deren Auftragnehmer.

● Wegen des wachsenden Interesses hat das BSI einen Grundschutz-Baustein für den Umgang mit Software in Containern entworfen.

● Der Container-Baustein ist derzeit ein Entwurf, also nicht verpflichtend, soll aber in die nächste Version des Grundschutz-Kompendiums aufgenommen werden. Nutzer sind aufgerufen, ihn zu kommentieren.

[2]

Bei der Novellierung hat das BSI das Vorgehen zur Risikoanalyse stark vereinfacht. Der Gefährdungskatalog enthält 47 Gefährdungen, die für alle IT-Landschaften zu beachten sind. Dies reicht von physischen Risiken wie Hochwasser über Fehler der Anwender bis hin zum Angriff. Außerdem enthält jeder Baustein spezifische Gefährdungen. Bei Containern sind dies vor allem Schwachstellen in Images, ungesicherte administrative Zugänge, potenzielle Schwachstellen der Orchestrierung, fehlende Persistenz und die Verwaltung von Zugangsdaten. Ein weiteres wichtiges Thema ist die Separierung der Anwendungen und eventuell der Mandanten.

Dieser Artikel zählt nicht alle Anforderungen auf oder erklärt diese. Er zeigt die wichtigsten Risiken die relevanten Anforderungen aufzeigen, um einen Einblick in die Herangehensweise und den Umfang zu geben.

Basis der Absicherung bilden die Container-Hosts. Deswegen sind immer die Bausteine "Allgemeiner Server" und "Server unter Linux" anzuwenden. Zusätzlich enthält der Baustein die Anforderungen

"SYS.1.6.A3 Härtung des Host-Systems" und "SYS.1.6.A4 Härtung der Software im Container" in der Basisstufe.

Prozesse in eingeschränkter Umgebung

An der Frage, welche Separierung angemessen ist, scheiden sich die Geister. Im Prinzip handelt es sich bei Containern nur um chroot-Umgebungen, die man bei Linux seit den 1990ern kennt, oder Jails, die BSD Anfang des Jahrtausends einführte. Die Großrechner-Welt arbeitete schon immer mit Containern. Letztlich sind sie wie die Paravirtualisierung, die vor 15 Jahren aufkam, nur Prozesse in einer eingeschränkten Umgebung – vergleichbar mit der Sandbox von Java. Der Vergleich hinkt etwas, da bei Containern die Abschottung über Cgroups und Namespaces des Kernels geschieht. Dennoch ist er absichtlich gewählt, denn besonders diese Sandbox war eher ein Sieb, zumindest was die Zahl der Schwachstellen über die Jahre angeht. Und auch ein Linux-Kernel beinhaltet Fehler.

Spätestens seit Rowhammer, **Meltdown und Spectre [3]** in all ihren Varianten sollte jedem klar sein, dass geteilte Hardware immer ein Sicherheitsrisiko birgt. Auch Ausbrüche aus Virtualisierung und Sandboxen gibt es ausreichend. Das soll nicht heißen, dass Container keine Separierung von Anwendungen ermöglichen. Aber die Entscheidung, welches Level von Separierung ausreicht, muss bei der Planung immer im Hinblick auf die Sicherheitsanforderungen von Daten und Anwendungen erfolgen. Der Baustein folgt diesem Prinzip und enthält daher die Basisanforderungen "SYS.1.6.A1 Planung des Container-Einsatzes" und "SYS.1.6.A2 Planung der Separierung", die konkret sowohl Container und Virtualisierung als auch unterschiedliche Hardware aufführen.

Der frei einsehbare Baustein [4] kann dem Anwender die Entscheidung nicht abnehmen, sondern nur Hilfestellung bieten und klar aufzeigen, dass Container Virtualisierung und physische Trennung nicht ersetzen können. Ein moderner, agiler und kostengünstiger Betrieb von Anwendungen ist nur so lange kostengünstig, wie die Daten nicht plötzlich im Internet stehen und die Kosten für die Ausfälle und Entschädigungen die gesparten Betriebskosten schnell um ein Mehrfaches übersteigen.

Ein anderer Aspekt der Separierung ist Datenverschlüsselung. Gespeicherte Daten kann das Betriebssystem des Container-Hosts meist schnell und zuverlässig verschlüsseln. Aber es hat die Schlüssel während dessen Betrieb im Zugriff, was ein Angreifer ausnutzen kann. Während des Transports bietet sich die Verschlüsselung der virtuellen Netzwerke an, die die Container miteinander verbinden. Hierfür enthält der Baustein die Standardanforderung "SYS.1.6.A14 Verschlüsselung der Netzkommunikation". Die Netzwerke der unterschiedlichen Anwendungen lassen sich so gut trennen und auch vor einem Angreifer im LAN schützen.

Die Netzwerke für Administration und Anwendungen sind natürlich immer voneinander zu separieren – "SYS.1.6.A9 Separierung der Netze". Dass dies technisch nicht immer einfach ist, zeigt die Praxis. Viele Orchestrierungen halten die Container zwar am Laufen, sichern aber den Zugang zu den Netzen des Hosts nicht ausreichend ab. Dass vor dem Start nicht bekannt ist, auf welchem Knoten ein Container startet, erschwert zudem die Zuordnung zu einem bestimmten Interface.

Containerisierung ist für viele Anwendungen sinnvoll, da sie gegenüber einem regulären Betrieb direkt auf dem Server Vorteile bietet. Über die Namespaces und die Cgroups lässt sich das Sicherheitsniveau steigern. Auch besitzen Container eine Schnittstelle mit der sie die Rechte über SELinux, SecComp und Apparmor einschränken können. Der größte Vorteil der Container liegt allerdings in der Automatisierung. Deskriptive Konfiguration und Einbindung in automatisierte Prozesse zur Softwareverteilung können die Sicherheit der Anwendungen erhöhen. So sind viele Image-Sammlungen mittlerweile mit einem Schwachstellenmanagement ausgestattet, das die Images automatisch auf sicherheitsrelevante Fehler in der enthaltenen Software untersucht.

Schwankende Image-Qualität

Üblicherweise lädt man Images von einer zentralen Sammlung. Docker bietet selbst Docker-Hub an. GitHub, GitLab und weitere Anbieter haben ebenfalls solche Sammlungen erstellt. Die Qualität dieser Images schwankt allerdings sehr. Die Anforderung "SYS.1.6.A7 Verwendung sicherer Images" definiert, worauf ein Nutzer achten muss. Da sie in den Basisanforderungen enthalten ist, muss jeder sie beachten und umsetzen. Die Standardanforderungen "SYS.1.6.A12 Freigabe von Images" und "SYS.1.6.A13 Updates von Containern" regeln den Umgang mit der Freigabe. Für erhöhten Schutzbedarf gibt es die Anforderungen "SYS.1.6.A21 Automatisierte Auditierung" und "SYS.1.6.A22 Eigene Trusted Registry".

Cloud-Infrastruktur-Software beinhaltet im Lieferzustand häufig keine Authentifizierung oder gar rollenbasierte Autorisierung. Dies ist in gut gesicherten Netzen vielleicht praktisch bei der Verwaltung großer Umgebungen, aber die Verteidigung gegen die Angreifer sollte immer durch möglichst viele Schichten geschehen. Daher enthält der Baustein die Basisanforderung "SYS.1.6.A11 Administrativer Fernzugriff auf Container" sowie die Standardanforderungen "SYS.1.6.A15 Identitätsmanagement der Administratoren" und "SYS.1.6.A19 Planung der Verwaltung und Orchestrierung", damit kein ungesicherter Socket Befehle von nicht authentifizierten oder autorisierten Personen annimmt.

Für den Baustein Container relevante Gefährdungen

G 0.14	Ausspähen von Informationen (Spionage)
G 0.19	Offenlegung schützenswerter Informationen
G 0.20	Informationen oder Produkte aus unzuverlÃ¤ssiger Quelle
G 0.21	Manipulation von Hard- oder Software
G 0.23	Unbefugtes Eindringen in IT-Systeme
G 0.24	Zerstörung von Geräten oder Datenträgern
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
G 0.45	Datenverlust
G 0.46	Integritätsverlust schützenswerter Informationen

Benutzermanagement ist auch für die Anwendungen im Container relevant, denn diese laufen unter einer Benutzerkennung des Hosts. Die Rechte hängen wie bei Prozessen außerhalb von Containern von den Rechten dieses Benutzers ab. **Docker [5]** & Co. versprechen, dass ein Prozess im Container nicht auf die Ressourcen außerhalb seines Namensraums zugreifen darf. Aber wie bei jeder Sandbox ist es nur eine Frage der Zeit, bis jemand eine Möglichkeit zum Ausbruch findet – sofern das nicht schon passiert ist. Daher fordert der Baustein als Standardanforderung in "SYS.1.6.A16 Accounts der Anwendungsdienste" und "SYS.1.6.A17 Container-Ausführung ohne privilegierten Account" die für die Absicherung von Diensten unter Unix üblichen Sicherheitsvorkehrungen. Bei erhöhtem Schutzbedarf sind auch die Anforderungen "SYS.1.6.A23 Reduzierte Rechte" und "SYS.1.6.A24 Erstellung erweiterter Richtlinien für Container" zu beachten. Sie fordern, die Rechte der Anwendungen im Container auf Syscalls, Netzwerke und Dateisysteme zu reduzieren.

So notwendig, wie die Verwaltung der Container durch die Orchestrierung besonders in größeren Umgebungen ist, so groß sind die damit verbundenen Risiken. Zumeist gibt es zusätzliche administrative Zugänge, Agenten, Daemons, insgesamt Millionen Codezeilen, die Schwachstellen enthalten können. Hier sind die Anforderungen für die Absicherung der administrativen Zugänge und die Anforderungen für die Separierung der Netze auch auf die Software zur Orchestrierung anzuwenden. Derzeit gibt es keinen spezifischen Baustein für die Orchestrierung, aber der Container-Baustein enthält die Standardanforderung "SYS.1.6.A19 Planung der Verwaltung und Orchestrierung", die die wichtigsten Aspekte anreißt und in den Umsetzungshinweisen genauer ausführen wird.

Dokumente sollten nichtflüchtig sein

Daten dauerhaft zu speichern, ist in einer Umgebung mit flüchtigen Anwendungen – die Container können jederzeit starten, stoppen und in der Zahl variieren –, ein besonderer Aspekt. In diesem unterscheiden sich Container- und Nicht-Container-Welt stark. Der Baustein beinhaltet dafür die Basisanforderungen "SYS.1.6.A5 Persistenz von Protokollierungsdaten", "SYS.1.6.A6 Persistenz von Nutzdaten" und "SYS.1.6.A10 Einbinden von Volumes". Die Anforderungen sind wie alle unspezifisch für die eingesetzte Technik, erst die Umsetzungshinweise gehen auf diese ein.

Die meisten Anwendungen, sei es ein Microservice oder ein Monolith, benötigen Zugangsdaten, etwa für Datenbanken oder andere Dienste, von denen sie Daten beziehen oder an die sie senden. Besonders die Microservices verlangen eine solide Authentifizierung und Autorisierung zwischen den einzelnen Diensten. Wenn eine Anwendung Dienste über öffentliche Netze einbindet, wird die Authentifizierung noch wichtiger. Denn dann fällt der Schutz eines Perimeters weg und der Dienst ist aus dem gesamten Internet erreichbar und damit angreifbar.

Hier bieten alle Hersteller von Containern und Orchestrierung Werkzeuge an, mit denen sich diese Zugangsdaten spezifisch für jede Umgebung verwalten lassen. Ein Passwort für die Datenbank muss für die Entwicklung ein anderes sein als für Staging und Produktion. Der Baustein enthält dazu die Basisanforderung "SYS.1.6.A8 Speicherung von Zugangsdaten".

Leider werden jeden Tag noch Anwendungen und Seiten gehackt, weil Programmierer die Zugangsdaten im Code speichern und diese mit dem Code zum Beispiel auf GitHub landen. Für Software in der Cloud ist das Thema daher von besonderem Interesse und die Entwicklung oder der Betrieb von Anwendungen sollte das Thema frühzeitig beachten und absichern.

Fazit

Der Baustein-Draft gibt einen Ausblick, was der zukünftig für die Zertifizierung von Unternehmen und die Bundesbehörden relevante Baustein Container enthalten wird. In seiner jetzigen Form ist er noch nicht bindend, da es nur ein Entwurf zur Kommentierung ist. Wer heute schon seinen Betrieb mit Containern am Grundschutz ausrichten will, hat zumindest einen Einstieg und eine Vorlage für einen gemäß Standard zu erstellenden Baustein.

Christoph Puppe ist Security Consultant bei www.sva.de, Audit-Teamleiter, IS-Revisor, Mitautor der Grundschutzkataloge und ehemaliger Penetrationstester. Der Autor hat den Baustein initiiert und am Draft maßgeblich mitgewirkt. ()

URL dieses Artikels:

https://www.heise.de/-4220176

Links in diesem Artikel:

[1] https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzAbout/IT-Grundschutz-Modernisierung/itgrundschutz_modernisierung_node.html

[2] https://www.heise.de/ix/

[3] https://www.heise.de/thema/Meltdown-und-Spectre

[4] https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-Grundschutz-Modernisierung/BS_Container.html

[5] https://www.heise.de/thema/Docker