Editorial

Kubernetes: Der Heilsbringer für Containerprobleme?

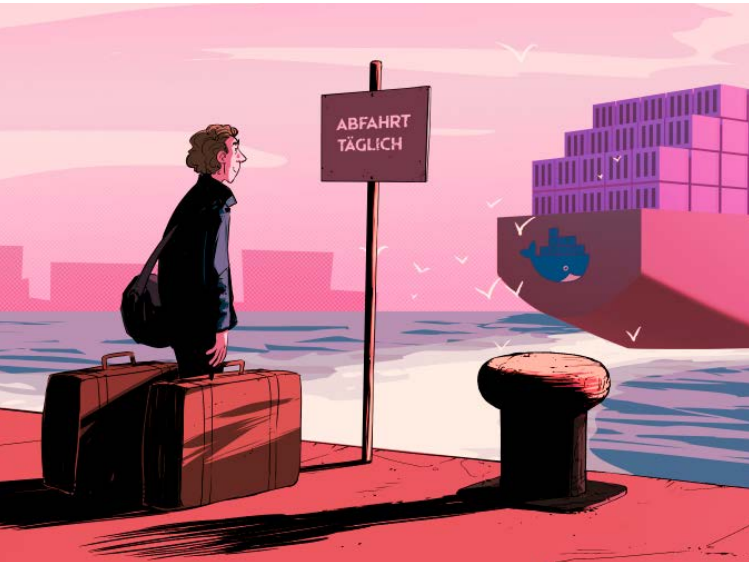
Es ist ein Running Gag in Teilen der IT geworden: Virtualisierung löst unsere Probleme mit Computern. Container lösen unsere Probleme mit Virtualisierung. Und wer löst unsere Probleme mit Containern? Kubernetes vielleicht? Genau mit diesem Ziel ist der Containerorchestrator im Jahr 2015 angetreten, um Admins beim Steuern riesiger Containerumgebungen in Cloudrechenzentren etwa bei Google, AWS & Co. zu unterstützen. Doch diese Lage hat sich gewandelt, mittlerweile hilft Kubernetes auch Mittelständlern.

Wenn Sie vor der Herausforderung stehen, die Containeraufgaben Ihrer Organisation mit Docker, Podman oder direkt mit Kubernetes zu lösen, haben Sie das richtige Heft in der Hand. Es richtet sich vornehmlich an alle, die schon mit Containern arbeiten und Erfahrungen gesammelt haben, an Admins wie Entwickler gleichermaßen. Sie erfahren Schritt für Schritt, wie Sie Ihren ersten Kubernetes-Cluster einrichten und mit YAML-Dokumenten konfigurieren, was es mit Pods, Services, IngressRoutes und PersistentVolumeClaims auf sich hat. Wir reichen Ihnen das komplette Handwerkszeug, um eine containerisierte Umgebung, die bisher in Docker oder Podman lief, in Kubernetes zu überführen. Weiter geht es mit erprobten Strategien aus der Praxis für Storage und einen automatisierten Containerbetrieb auf Basis eines Git-Repos.

All Ihre Probleme mit Computern im Allgemeinen und Containern im Speziellen werden Sie damit nicht lösen, aber mit Kubernetes haben Sie Zugriff auf ein mächtiges Werkzeug. Das Wissen dazu ist die Eintrittskarte in ein überwältigendes Open-Source-Ökosystem etwa für Monitoring über redundante Datenbanken und automatische Skalierung bei Lastspitzen. Fast jede Hürde, auf die Sie in Ihrer Kubernetes-Karriere stoßen, hat schon jemand anderes vor Ihnen genommen. Und in den meisten Fällen gibt es ein Open-Source-Projekt, das Ihnen herüberhilft.

Jan Mahn

Inhalt



CONTAINER MIT DOCKER UND PODMAN

Docker und Podman sind auf Entwicklerrechnern und kleinen Servern zu Hause und die Grundlage für jedes Container-Projekt. Der Ein- und Umstieg ist einfach; bei Security und Netzwerkverbindungen gilt es, auch unbekanntere Funktionen zu entdecken.

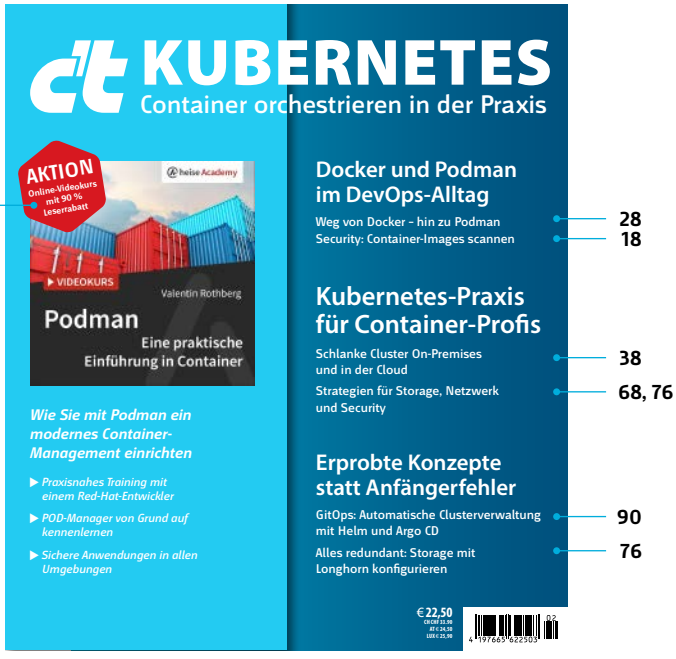
DER KUBERNETES-LERNPFAD

Sobald die Anforderungen steigen, reichen Docker und Podman nicht aus – das Wissen können Admins und Entwickler in die Kubernetes-Welt mitnehmen und steigen somit Stück für Stück in die Technik ein, die auch Dienste von Weltkonzernen betreibt.

KONZEPTE FÜR FORTGESCHRITTENE

Der erste Cluster läuft, die ersten Container sind umgezogen – Zeit für einen Blick auf erprobte Konzepte, die den Admin-Alltag leichter machen. Und auf Open-Source-Software, die alltägliche Probleme aus der Welt schafft.

ZUM HEFT



heise Academy-Aktion:

Schneller Einstieg in Podman

Das Open-Source-Tool Podman verspricht hohe Sicherheit und mehr Zugriffsmöglichkeiten für das moderne Container-Management. Im Videokurs der heise Academy unternehmen Sie eine spannende Tour durch die Grundlagen dieser vollwertigen Engine – mit der Expertise der Entwickler.

Von **Markus Richter**

Sie beschleunigen die Entwicklung von Anwendungen und vereinfachen sie zugleich: Container sind zu Recht einer der großen Trends der professionellen IT. Engineer Teams von Red Hat haben dafür zusammen mit der Open Source Community den POD-Manager Podman auf der Basis von Docker entwickelt und eine Lösung geschaffen, mit der sich das gesamte Container-IT-Ökosystem verwalten lässt. Im Unterschied zu anderen Tools wird Podman ohne Daemons ausgeführt, die eine Sicherheitsschwachstelle darstellen können. Auch auf Root-Rechte verzichtet man bei Podman.

Die heise Academy hat für ihren Einführungsvideokurs zu Podman Valentin Rothberg gewinnen können, der als Software Engineer bei Red Hat unmittelbar mit der Materie vertraut ist. Mit dem Kauf dieses c't-Sonderheftes können Sie gegen eine Schutzgebühr von 4,95 Euro statt regulär 49 Euro an dem Videokurs teilnehmen. Der Trainer stellt darin die wichtigsten Basics von Linux-Containern und Container-Images vor. Wie sehen Container-Prozesse auf dem Linux-System aus? Woraus besteht ein Container-Image? Für die Arbeit mit lokalen Daten und Verzeichnissen lernen Sie, wie Sie mit Volumes und Mounts umgehen, insbesondere als Rootless-Nutzer. Mit Podman können Sie bequem auch auf dem Mac- oder Windows-Rechner mit Containern arbeiten.

Academy-Trainer Valentin Rothberg ist in Red Hat's „Container Runtimes Team" beschäftigt und kennt sich umfassend mit Container-Werkzeugen

Über heise.de/s/VxVd erhalten Sie diesen Videokurs mit dem Rabattcode [redacted] einmalig für nur 4,95 Euro, statt für 49 Euro*.

*Preis- und andere Irrtümer vorbehalten. Das Angebot ist gültig bis zum 31.12.2023 (Stand: Juni 2023).

Das lernen Sie im Videokurs

- Einen Linux-Container verwenden
- Die Engine Podman verstehen
- Container mit Podman ausführen und verwalten
- Arbeiten mit Volumes und Mounts
- Rootless-Container nutzen

und ihren Technologien aus. Er hat zu vielen anderen Projekten in der Container-Landschaft beigetragen, darunter Kubernetes, Linux-Kernel, Moby, Google Cloud und container-diff. Vor seiner Tätigkeit in der Industrie war Valentin Rothberg in der Forschung und akademischen Lehre von Betriebssystemen tätig.

Die heise Academy – IT-Weiterbildung neu gedacht

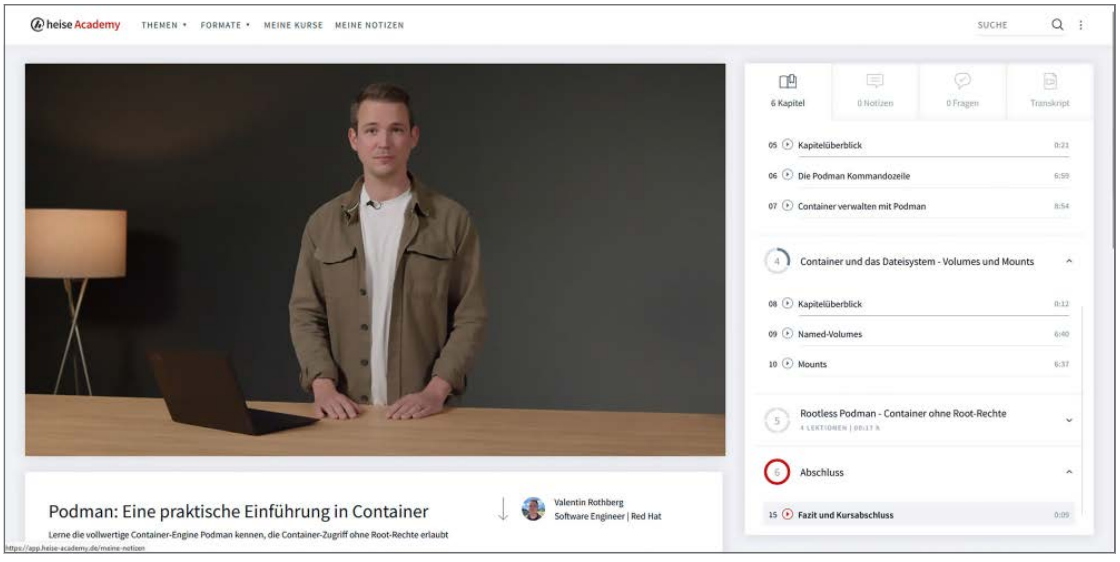
Dieser Videokurs gehört zum Angebot der heise Academy, die sämtliche Weiterbildungen der heise-Verlagsgruppe unter einem Dach bündelt. Interessierte IT-Professionals und Unternehmen finden auf www.heise-academy.de zeitgemäße und maßgeschneiderte Wissensangebote. Damit können Sie Ihre Skills vertiefen, neue Schwerpunkte in Ihrer Arbeit setzen, Ihre Karriere voranbringen und mit Spaß lernen.

Mit dem Launch der neuen Seite sind ab sofort alle Weiterbildungsangebote übersichtlich und strukturiert auf einer Plattform vereint. Finden Sie Ihren passenden On-Demand-Kurs für das selbstbestimmte Lernen, oder besuchen Sie von führenden IT-Experten durchgeführte Live-Events wie Webinare, Konferenzen, Schulungen und Workshops.

Von Netzwerken und Systemen über IT-Projektmanagement, Softwareentwicklung, Data Science und IT-Security bis hin zu Web- und Cloud-Technologien bietet die heise Academy jede Menge geballtes Fachwissen für die professionelle Anwendung. Dabei steht die Wissensvermittlung durch ausgewählte Experten im Mittelpunkt. Mehr als 100 erfahrene Trainer kommen aus dem gesamten deutschsprachigen Raum und bedienen unterschiedliche Schwerpunkte mit starkem Praxisbezug.

Buchen Sie einzelne Events oder Kurse oder treten Sie dem Campus mit dem Academy Pass bei. Diese Lern-Flatrate beinhaltet mehr als 100 Videokurse – darunter auch weitere zum Thema Container, Kubernetes und Podman – sowie eine Vielzahl an Webinaren. Die Videokurse schauen Sie in einem eigens entwickelten Player, der eine komfortable Benutzeroberfläche bietet. Durchsuchen Sie den Kurs mithilfe einer Volltextsuche nach Stichworten, hinterlegen Sie persönliche Notizen und stellen Sie Fragen an die Trainer. So ist neben den Vorteilen der On-Demand-Nutzung die Möglichkeit zur Interaktion mit den Experten gegeben. Der Campus bietet zudem mehr als 100 Live-Webinare jährlich, die immer am Puls der Zeit sind. Hier wird über Trendthemen diskutiert, an praktischen Fallbeispielen geübt und eine Lösung für jedes IT-Problem angeboten.

Alle Infos zu den Angeboten finden Sie unter www.heise-academy.de. (anm) c't



Red-Hat-Entwickler Valentin Rothberg zeigt im Videokurs die Grundlagen von Podman.

Container verstehen und loslegen

Ab 2013 hat Docker die Containertechnik salonfähig gemacht. Von Kritikern erst als kurzfristiger Hype verschrien, sind Container mit Docker, Podman und Kubernetes wenig später zur etablierten Technik geworden. Für den Einstieg ist es lange noch nicht zu spät: wie Sie von Containern profitieren und Software mit und ohne Docker betreiben.

Von **Jan Mahn**

Bild: Albert Hulm

Technische Neuerungen spalten oft die Gemüter: Auf der einen Seite gibt es die Early Adopter, die alles Neue sofort anfassen und ausprobieren müssen und mit dem Risiko leben, wenig später ein totes Pferd reiten zu müssen. Auf der anderen Seite stehen die Skeptiker, die sich das bunte Treiben lieber von außen anschauen und darauf warten, dass sich eine gewisse Stabilität einstellt – oft ist das eine gute Strategie, weil viele Hype-Themen nach wenigen Jahren still und leise in der Versenkung verschwinden. Auch bei Docker, eine Software, die 2013 erstmals erschien, war Vorsicht durchaus angebracht. Dennoch handelten viele die Technik eines kleinen Open-Source-Start-ups schnell als Quasi-Industriestandard.

Viele Baustellen im Docker-Unterbau haben sich rasanter verändert, als die Entwickler ihre Dokus und die Vertriebler ihre Prospekte anpassen konnten. Nicht nur die Open-Source-Software Docker selbst, auch das Geschäftsmodell der Docker Inc. hat einige Kurswechsel hinter sich.

Wer bisher abgewartet hat, konnte sich einige Sackgassen und Irrwege ersparen. Nach fast 10 Jahren auf dem Markt zeichnet sich aber ab: Container bleiben uns erhalten, ob mit oder ohne Docker. Und die Zeit der großen Umbrüche ist vorbei. Auf den Maschinen von Entwicklern laufen Docker oder Podman, um Images zu erproben, zu entwickeln. Auch in kleinen Produktivumgebungen sind Docker und Podman zu Hause.

Sobald die Anforderungen größer werden, kommen die Container in einen Kubernetes-Cluster. Der Umstieg ist vergleichsweise einfach, weil dieselben Images zum Einsatz kommen.

Seit etwa 2016 berichten wir in c't regelmäßig über Container-Technik im Allgemeinen und Docker im Speziellen, veröffentlichen Artikel mit Grundlagen und stellen viele Projekte auf Container-Basis vor, die Docker-Grundwissen voraussetzen. Häufig erreicht uns daher die Frage, welche schon veröffentlichten Artikel man lesen muss, um schnell ins mit Linux-Unterbau – Windows-Container auf Windows-Servern sind eine eigene Baustelle. Wenn Sie Docker den Rücken kehren wollen und stattdessen einen Blick auf den Open-Source-Marktmitbewerber Podman werfen wollen, werden Sie im Artikel „Von Docker Desktop auf Podman wechseln“ auf Seite 28 fündig. Fast alles, was unter Docker funktioniert, klappt auch mit Podman.

Was habe ich davon?

Als reiner Desktop-PC-Anwender (sei es unter Linux, Windows oder macOS) haben Sie nichts verpasst, wenn die Dockerei bisher an Ihnen vorbeiging. Docker ist eine Software, die für Admins und Entwickler gemacht wurde. Sinnvolles Einsatzgebiet sind Serverdienste, aber auch für Kommandozeilenwerkzeuge können Container durchaus nützlich sein. Mit etwas Bastelei kann man theoretisch auch grafische Anwendungen im Container betreiben, erste Wahl für Desktop-Software sind die Container aber nicht.

Um den Nutzen von Containern und die Funktionsweise zu verstehen, ohnt ein Blick auf die Arbeitsschritte, die ohne nötig sind, bis eine Serversoftware funktioniert. Anschauliches Beispiel ist die populäre Bloganwendung WordPress auf einem Linux-Server. Damit WordPress läuft, braucht man einen Ordner mit den WordPress-Dateien selbst, eine PHP-Engine, einen Webserver (Apache, Nginx …) und eine SQL-Datenbank (MariaDB oder MySQL).

Auf einem nackten Linux-Server würde man zur Installation damit beginnen, die Komponenten über den Paketmanager herunterzuladen, etwa per `apt install` in Debian, Mint oder Ubuntu. Anschließend konfiguriert man alle Bausteine einzeln in ihren Konfigurationsdateien (die unter Linux meist im Ordner /etc. liegen) und verdrahtet dann die Komponenten miteinander: Der Webserver muss mit der PHP-Engine sprechen und ihm Dateien mit der Endung .php zum Parsen vorwerfen, der PHP-Code muss die Datenbank erreichen und wissen, wo er

Lesen Sie mehr in c't Kubernetes 2023

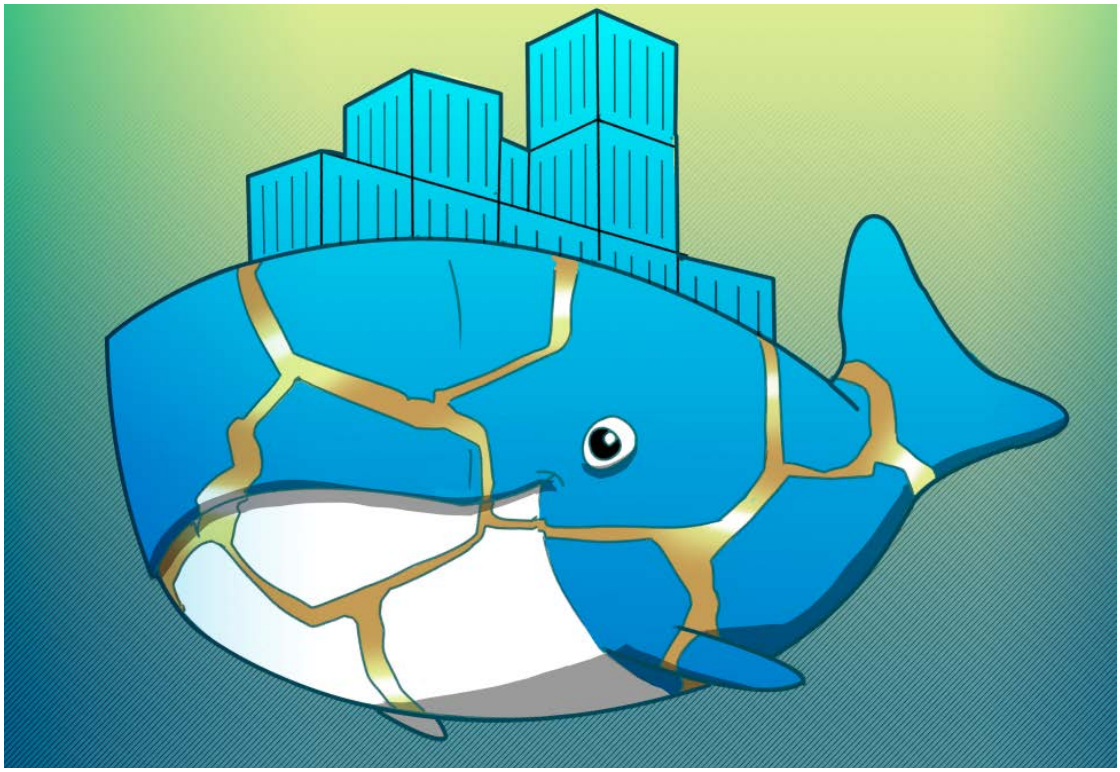


Bild: Thorsten Hübner

So harmonieren Docker und IPv6

Globale IPv4-Adressen sind knapp. Deshalb sollten heute im Netzwerk angebotene Dienste selbstverständlich per IPv6 erreichbar sein. Doch die verbreitete Container-Umgebung Docker erschwert es, IPv6 sinnvoll einzusetzen. Unser Artikel sortiert die Einzelteile, um sie besser zusammenzufügen.

Von **Peter Siering**

Dass Docker und IPv6 fremdeln, wird an vielen Stellen deutlich. Sichtbar zu Tage tritt es, wenn man die Log-Daten von Containern studiert: Während dort die IPv4-Adressen der anfragenden Clients auftauchen, findet sich für Anfragen von IPv6-Clients darin nur die IPv4-Adresse der internen Netzwerkschnittstelle „docker0“. Um dieses Problem zu lösen und für weitere präpariert zu sein, hilft es, die Docker-Netzwerkmöglichkeiten zu rekapitulieren.

Docker kennt grundsätzlich mehrere Techniken, um Container mit dem lokalen Netz zu verbinden. Dieser Artikel betrachtet den meistgenutzten Typ „Bridge“ und den produktiven Betrieb auf einem Linux-Server. Hier helfen spezielle Mechanismen, um die Container vom Netzwerk des Hosts zu separieren, um ihnen untereinander die Kommunikation zu erlauben und um Zugriffe von außen auf Dienste in den Containern zu realisieren.

Andere Netzwerktypen kommen bei besonderen Wünschen zum Einsatz: der Typ „Overlay“, wenn Container über mehrere Hosts verteilt erreichbar sein sollen, die Typen „Host“ und „macvlan“, wenn ein Container direkt am Netzwerk des Hosts lauschen soll, etwa um dort Broadcasts zu empfangen oder zu senden, und der Typ „ipvlan“, um komplexe virtuelle Netzwerke zu bauen. Alle Typen funktionieren grundsätzlich sowohl mit IPv4 als auch mit IPv6.

Bockige Brücke

Für die meisten Anwendungsfälle genügt ein Bridge-Netzwerk. In einer regulären Installation kümmert sich Docker auf Anforderung darum, dass im Container laufende Netzwerkdienste von außen über die IP-Adresse des Docker-Hosts erreichbar sein sollen. Das kann man sich wie eine Portfreigabe beim DSL-Router vorstellen. Das Image deklariert dazu, welche Ports es exponiert, beim Starten des Containers weist Docker ihnen Ports des Hosts zu, einem simplen Webserver etwa Port 80.

Auf dem Docker-Host auf Port 80 eingehende Netzwerkzugriffe leitet der dann an den Container weiter. Der Host kann dabei weltweit erreichbare globale IP-Adressen besitzen, etwa als Mietserver eines Cloud-Hosters. Der Container hat standardmäßig nur eine private IPv4-Adresse, die Docker dem gängigen für die lokale Nutzung reservierten Adressraum entnimmt (etwa 172.18.0.0/16).

Ob die externen Zugriffe auf IPv4- oder IPv6-Adressen des Hosts erfolgen, spielt keine Rolle. Es lohnt aber ein genauer Blick hinter die Kulissen: Auf einer Linux-basierten Installation sowohl mit IPv4- als auch mit IPv6-Adresse startet Docker für jedes Protokoll und jeden exponierten Port einen eigenen Prozess namens docker-proxy, der Zugriffe auf den IPv4- oder IPv6-Port des Hosts an den IPv4-Port des Containers weiterreicht. Pro Port laufen

Docker IPv4 versus IPv6

Die Standardinstallation von Docker auf Linux-Hosts sieht unterschiedliche Wege für Netzwerkzugriffe auf Container in einem Bridge-Netzwerk vor. IPv4-Zugriffe (gelb) landen über von Docker eingerichtete Firewallregeln im Container, IPv6-Zugriffe (grün) hingegen fließen über einen speziellen docker-proxy-Prozess. Erst nach dem Aktivieren experimenteller Funktionen verwendet Docker für IPv6 ebenfalls Firewallregeln (grün gepunktet).

Der Lernpfad zum Kubernetes-Kenner

Kubernetes-Experten sind gefragt und viele Docker-Nutzer würden die andere Seite des Container-Universums gern mal kennenlernen – wäre das Ökosystem nicht so groß und undurchsichtig. Mit unserer ausführlichen Praxis-Reihe gelingt der Einstieg: Der erste Teil zeigt, wie Sie aus drei Linux-Servern einen Cluster bauen.

Von **Jan Mahn**

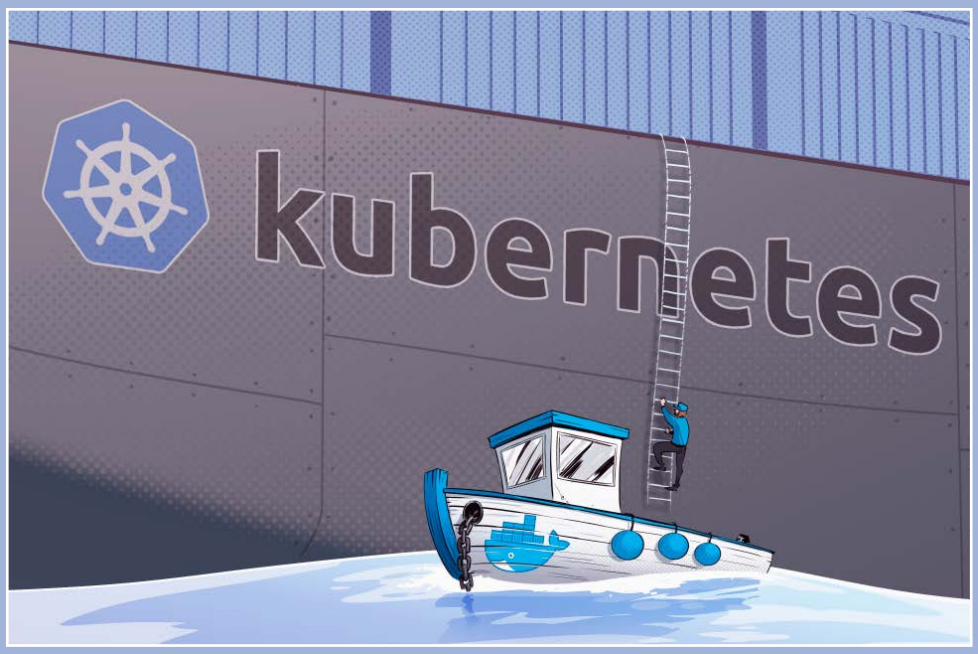


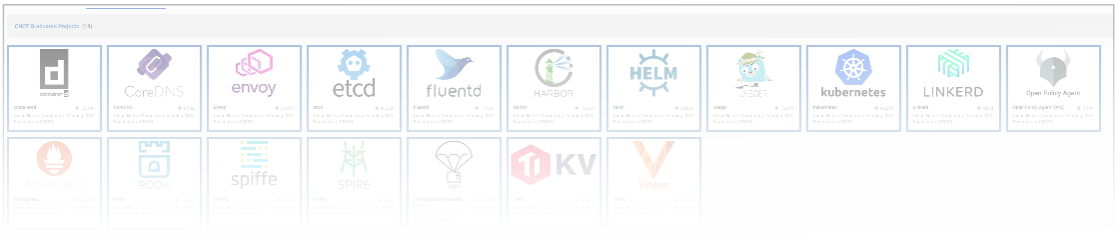
Bild: Albert Hulm

Das Softwareprojekt ist zu groß geworden für einen einzigen Docker-Server, Ihre Chefs erwarten von Ihnen jetzt Kubernetes-Erfahrung oder Sie wollen aus eigenem Antrieb verstehen, wie man seine Container mit der Software betreibt, die auch Schwergewichte wie Netflix, Spotify und Banken im Einsatz haben. Gründe, sich heute an den Einstieg in Kubernetes zu wagen, gibt es viele – Voraussetzung ist lediglich ein souveräner Umgang mit Docker oder einer anderen Container-Umgebung wie Podman. Wenn Sie noch nicht überzeugt sind, warum Sie Kubernetes brauchen und lernen sollten, finden Sie Argumente im Kasten „Darum Kubernetes“ auf Seite 40. Aber ohne, dass man einige Monate lang Container betrieben, Abbilder heruntergeladen und eigene gebaut hat, sollte man die Finger von Kubernetes lassen; Frust wäre garantiert. Eine Einführung in Docker und den aktuellen Stand lesen Sie im Artikel „Container verstehen und loslegen“ auf Seite 8.

Auch für erfahrene Docker-Nutzer führt der naheliegendste Weg in die Kubernetes-Welt leider schnell in eine Sackgasse. Beim ersten Blick auf die offizielle Kubernetes-Dokumentation wird man ziemlich zuverlässig erschlagen. Die liegt unter docs.kubernetes.io und wird später ein zuverlässiger Begleiter. Wie Sie vielleicht schon mitbekommen haben, stammt Kubernetes ursprünglich aus dem Hause Google und wird jetzt als branchenübergreifendes Open-Source-Projekt entwickelt. Daher arbeitet ein Team aus Dokumentationsprofis daran, die Texte auf dem aktuellen Stand zu halten und leistet gute Arbeit. Die Doku verrät jedes Detail und ist ein unverzichtbares Nachschlagewerk, denn auswendig lernen kann niemand alle Funktionen von Kubernetes. Für Einsteiger ist dieses Werk jedoch keine Empfehlung – das liegt auch daran, dass Sie neben Kubernetes auch gleich ein ganzes Ökosystem aus Open-Source-Projekten kennenlernen müssen, die im Zusammenspiel mit Kubernetes funktionieren. Und oft gibt es auch mehrere Projekte, die dasselbe Problem lösen. Die Kubernetes-Doku allein enthält also nur einen Teil der Wahrheit. Im Ökosystem gibt es aber so viele Pfade und Verzweigungen, dass man sich allzu leicht verlaufen kann.

Eigene Erfahrungen statt Theorie

Diese Artikelreihe möchte einen möglichen Weg durch das Profi-Container-Dickicht aufzeigen. Nicht den einzigen Weg und sicher nicht den besten Weg für alle erdenklichen Umgebungen, aber einen, der sich für Docker-Kenner bewährt hat. Wo es angebracht ist, erhalten Sie Hinweise auf alternative Routen. Im Mittelpunkt steht das Ausprobieren und Nachbauen: Anhand einer Anwendung, die aus einer Ein-Server-Docker-Umgebung in die Kubernetes-Welt umziehen soll, lernen Sie Kubernetes-Konzepte, Werkzeuge aus dem Ökosystem und erprobte Lösungsansätze kennen. Auf dem Weg verinnerlichen Sie Begriffe und Kommandozeilenbefehle ganz automatisch.



Kubernetes allein ist nicht der Schlüssel zum Erfolg – es ist das

Lesen Sie mehr in c’t Kubernetes 2023

Redundanter Speicher mit Longhorn

In einem Kubernetes-Cluster laufen Anwendungen skalierbar und redundant. Damit auch die anfallenden Daten auf mehreren Servern liegen und der Speicherplatz mit den Anforderungen wächst, braucht man eine Erweiterung wie Longhorn. Sie macht Volumes redundant – eine Backup-Strategie gibt es obendrauf.

Von **Jan Mahn**

Bild: KI Midjourney | Bearbeitung: c't

Mit dem Umstieg von einem einzelnen Docker-Server auf einen Kubernetes-Cluster eröffnen sich schier grenzenlose Möglichkeiten, die eigene Anwendung zu skalieren. Wie Sie diesen Weg beschreiten und vom Docker- zum Kubernetes-Kenner werden, haben wir auf den vorangegangenen Seiten im Kapitel „Der Lernpfad zum Kubernetes-Kenner“ ab S. 38 beschrieben. Wachsen die Anforderungen, kann man mit Kubernetes problemlos Server nachbestellen und in den Cluster aufnehmen, um größeren Lasten zu begegnen. Was mit Kubernetes-Bordmitteln aber nicht mitwächst, ist der Speicherplatz. Muss ein Container etwas auf der Festplatte speichern, geschieht das über mehrere Abstraktionsschichten (VolumeMount, Volume, PersistentVolumeClaim und StorageClass, siehe Artikel „Volumes, Secrets und ConfigMaps“ auf S. 60). Der Prozess, der im Container läuft, bekommt von solchen Details nichts mit – ihm setzt die Container-Runtime ein Dateisystem vor, das er lesen und auf Wunsch auch beschreiben kann.

Ein ganz einfacher Anbieter von Kubernetes-Speicherplatz ist zum Beispiel der LocalPathProvisioner von Rancher (siehe ct.de/w6tu), den die leichtgewichtige Kubernetes-Distribution k3s bereits mitliefert. Der schnappt sich einfach einen Ordner im Dateisystem des Nodes, auf dem der Container läuft und reicht ihn an den Container weiter. Dieses Verhalten entspricht ziemlich genau dem, was Docker-Nutzer von „named volumes“ kennen, also solchen Volumes, die man mit Befehlen wie `docker volume ls` und `docker volume create` verwaltet. Doch in einem Cluster ist das gar nicht mal so praktisch: Liegen die Daten auf einem einzigen Node, wird es zur Qual, den Pod auf einen anderen Node umziehen zu lassen – fällt ein Node mit angehängtem Volume aus, kann Kubernetes ihn nicht woanders unterbringen. Redundant gespeichert wird vom LocalPathProvisioner auch nichts.

Auftritt von Longhorn

ten dann im täglichen Betrieb. Dass Anbieter wie Longhorn eine solche Speicherschicht für Kubernetes bauen können, liegt an einem Konzept namens „Container Storage Interface“ (CSI). Das ist die Kubernetes-Plug-in-Schnittstelle für Speicherplatz (sogenannten Block-Storage). Die wurde von den Kubernetes-Maintainern geschaffen, als sich immer klarer abzeichnete, dass man verschiedene Speicheranbindungen unmöglich im Kubernetes-Code selbst entwickeln kann.

Longhorn installieren Sie am schnellsten über den Kubernetes-Paketmanager Helm, der auf der lokalen Entwicklermaschine eingerichtet ist und dort auf die Kubeconfig-Datei mit den Cluster-Zugangsdaten zugreift (siehe Artikel „Services und Ingress mit Traefik“ auf S. 52). Vor dem Helm-Einsatz müssen Sie die Kubernetes-Welt aber kurz verlassen und auf den Servern zwei Pakete auf Linux-Ebene installieren. Longhorn erwartet, dass iSCSI auf allen Maschinen installiert ist. Unter Debian und Ubuntu erreichen Sie das mit dem folgenden Befehl, direkt auf den Servern ausgeführt:

```
sudo apt install open-iscsi
```

Unter SUSE und openSUSE mit dem Paketmanager Zypper heißt das Paket ebenfalls `open-iscsi`. Die mehrschrittige Anleitung für Server-Distributionen aus der Red-Hat-Großfamilie finden Sie in der Longhorn-Dokumentation (siehe ct.de/w6tu).

Zweite Voraussetzung ist ein NFS-Client. Den bekommen Sie unter Debian und Ubuntu per

```
sudo apt install nfs-common
```

Unter Red Hat heißt das Paket `nfs-utils`, bei SUSE `nfs-client`. Wenn Sie von manuellen Paketinstallationen genervt sind, sollten Sie die Installation der beiden Pakete für all Ihre Kubernetes-Server automatisieren – zum Beispiel mit Ansible oder zumindest per Skript.