Hilfe bei Microservices:
Kubernetes mit Support

Gut sortiert

Christian Rost, Thorsten Schifferdecker, Christian Schneemann

Kubernetes bereitzustellen, ist nicht trivial. Dabei helfen die vorgefertigten Kubernetes-Distributionen SUSE CaaS Platform 4 und Red Hat OpenShift Container Platform 4 sowie die Werkzeugsammlung von Rancher.

Wer einen Kubernetes-Cluster einrichten will, stemmt eine Mammutaufgabe, für die er die geeigneten Werkzeuge braucht. Deshalb haben die Open-Source-Häuser SUSE und Red Hat vorgefertigte Distributionen zusammengestellt, die alles Notwendige zum Einrichten und Betreiben eines Kubernetes-Clusters enthalten und für die die Anbieter kommerziellen Support leisten. Auch für die Werkzeugsammlung von Rancher zum Aufsetzen und Verwalten von Kubernetes-Clustern ist kommerzieller Support erhältlich.

Mit allen drei lässt sich relativ schnell eine kleine Kubernetes-Umgebung bereitstellen. Auch wenn es in diesem Artikel nur um das Deployment von Kubernetes geht, liefern SUSE und Red Hat jedoch nicht „nur“ reines Kubernetes und eine Workload-Verwaltung, sondern auch Tools für das Development und Lifecycle-Management von Applikationen. Zum Vergleich der Produkte dient eine Standardinstallation.

SUSE CaaS Platform

SUSE bietet mit seiner CaaS Platform eine Cluster-Management-Suite für das Lifecycle-Management auf Basis von Kubernetes (Abbildung 1). Seit September 2019 ist SUSE CaaS Platform in der Version 4 verfügbar. Die Plattform hilft beim Deployment eines Kubernetes-Clusters und versucht, den Aufbau so einfach wie möglich zu gestalten. CaaS Platform 4 hat im Vergleich zur Version 3 einige Veränderungen gebracht. Wurde bei CaaS Platform 3 noch SUSEs MicroOS installiert, lässt sich Version 4 als Produkt auf SUSE Linux Enterprise Server 15 SP 1 einrichten. Die Installation kann manuell erfolgen, mit der Installer-DVD von SUSE Linux Enterprise 15 Service Pack 1, per AutoYaST oder über Images in virtuellen oder Cloud-Umgebungen.

SUSE stellt auch leicht anzupassende Terraform-Beispiele zur Verfügung, mit denen sich schnell und einfach kleine Testumgebungen starten lassen. Neben einer aktuellen Kubernetes-Version, Release Kubernetes 1.16.2 seit CaaSP 4.1 wird auch die CRI-O-Containerlaufzeitumgebung installiert.

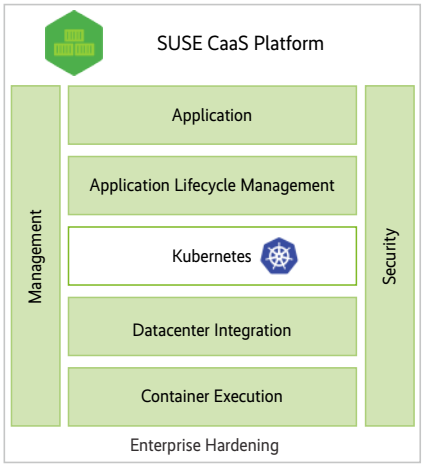
Die Stiefel schnüren

Nachdem die Clusterknoten installiert sind, erfolgen die Clusterkonfiguration und das Zuweisen der Rollen. Ersteres geschieht mit skuba, dem von SUSE entwickelten Kommandozeilentool zur einfachen Installation und Wartung von Kubernetes-Clustern. Es basiert auf kubeadm und ist in Go geschrieben. Zu den zu konfigurierenden Clustersystemen benötigt es lediglich eine SSH-Verbindung. Die Konfiguration ist im Dateisystem in einer Ordnerstruktur abgelegt. Dadurch lassen sich von einem Adminsystem aus mehrere Cluster verwalten (Listing 1, siehe ix.de/z1cj).

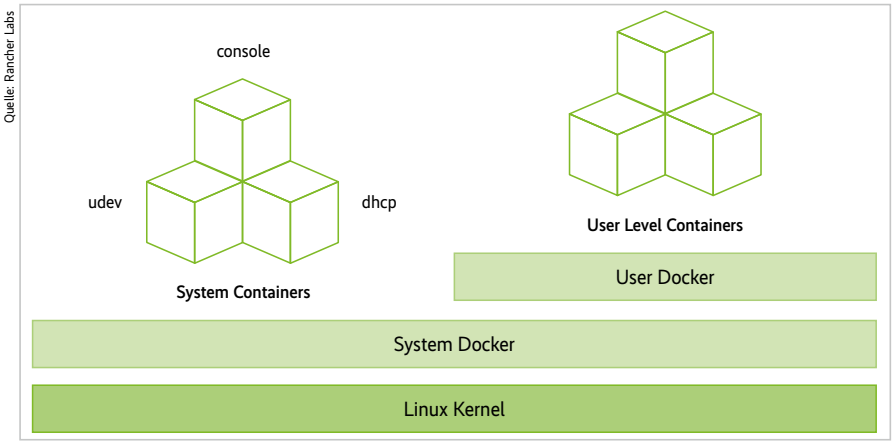
Das Tool skuba arbeitet dateisystembasiert und erstellt für jeden zu administrierenden Cluster ein Verzeichnis mit Konfigurationsdateien. Nach der Konfiguration des ersten Masters lassen sich dem Cluster unter Angabe der jeweiligen Rolle, Master oder Worker, weitere Knoten hinzufügen. Auch die Softwareupdates der Nodes steuert skuba, und zwar mit dem Werkzeug skuba-update, das den Paketmanager zypper und den Kubernetes Reboot Daemon kured nutzt. Der Daemon hilft, Clusterknoten automatisch neu zu starten, wenn das Paketmanagement des Betriebssystems das veranlasst. kured stellt

iX-TRACT

- Kubernetes ist nicht leicht bereitzustellen. Installation und Wartung sind komplex und es gilt, die Clusterkomponenten stets aktuell zu halten.
- Die drei großen Kubernetes-Distributionen SUSE Caas Platform, Rancher und Red Hat OpenShift bieten kommerziellen Support.
- Verschiedene Tools helfen beim Cluster-Deployment oder der Workload-Verwaltung.



Die SUSE Caas Platform automatisiert das Management von Containern. Sie vereinfacht den Aufbau von Kubernetes-Clustern (Abb. 1).



Die Container User Docker und System Docker sind getrennt. Das verhindert unbeabsichtigtes Löschen von Systemcontainern (Abb. 2).

dann sicher, dass immer nur ein Knoten neu startet und der Cluster normal weiterarbeiten kann. Am Ende steht ein funktionsfähiger Kubernetes-Cluster mit Cilium, CNI-Add-on, dex für die rollenbasierte Authentifizierung und – auf Wunsch – Helm zur Verfügung.

Will man eine Weboberfläche zum Management der Workload nutzen, lässt sich das Kubernetes-Dashboard per Helm-Chart installieren. Die Art und Weise der Nutzung der Kubernetes-Installation ist nicht von SUSE vorgegeben. Man hat freie Hand, welche Add-ons den Cluster erweitern sollen und wie dieser im Unternehmensnetzwerk einzubinden ist.

Rancher

Das Managementwerkzeug verwaltet Kubernetes-Cluster und Container-Workloads mithilfe verschiedener, unter der Apache License 2.0 veröffentlichter Tools. Basis ist das von Rancher herausgegebene RancherOS, eine leichtgewichtige Linux-Distribution, in der jeder Prozess in einem Container läuft. Statt eines init-Systems startet RancherOS als ersten Prozess eine Docker-Instanz, System Docker genannt, und diese anschließend sogenannte System-Docker-Container wie ntpd, rsyslog, acpid oder cron. Außerdem laufen Dienste wie udev oder die für die eigentliche Workload verwendete Docker-Instanz als Container in System Docker.

Die Trennung der Docker-Instanzen User Docker und System Docker isoliert die Container des Anwenders (hier Kubernetes inklusive Workload) vom Betriebssystem. Dies bringt nicht nur Sicherheitsvorteile, es verhindert auch das ungewollte Löschen von Systemcontainern und damit, dass RancherOS unbrauchbar wird (Abbildung 2).

RancherOS kann man Bare Metal oder als VM installieren. Für OpenStack, VMware, Amazon EC2, Google Compute Engine, DigitalOcean und Microsoft Azure gibt es vorgefertigte Images. RancherOS kann man auf zwei Arten konfigurieren: über ein Cloud-config-File, mit dem sich Rancher automatisiert installieren und konfigurieren lässt, und mit dem Kommando `ros config`, das man jedoch erst nach erfolgreicher Installation verwenden kann.

Rancher Kubernetes Engine, kurz RKE, ist ein Tool zum Erstellen, Verwalten und Aktualisieren von Kubernetes-Clustern. Es vereinfacht die Installation der Kubernetes-relevanten Komponenten. RKE benötigt ein Linux mit installiertem Docker-Daemon, die Möglichkeit, alle in den Cluster aufzunehmenden Hosts per SSH zu erreichen, und einen unprivilegierten SSH-Benutzer, der jedoch die Berechtigung hat, Docker-Container zu starten. Außerdem müssen noch diverse Kernel-Module geladen und ein `sysctl`-Wert gesetzt sein. Die Installation von RKE ist einfach: Man lädt das aktuelle Binary von github.com/rancher herunter, macht es ausführbar und kopiert es an eine Stelle, die sich in der `$PATH`-Variablen wiederfindet.

Installieren und konfigurieren

Anschließend erzeugt man mit `rke config` eine Clusterkonfiguration cluster.yml. Dieser Wizard ermöglicht es, auf einfache Weise einen Cluster zu beschreiben, den RKE dann automatisch installiert. Die Clusterkonfigurationsdatei lässt sich auch von Hand anlegen und editieren, da nicht alle Optionen über den Wizard konfigurierbar sind. Der Befehl `rke up` installiert

Listing 1: Bootstrapping eines CaaS-Platform-4-Clusters mit skuba

```
skuba cluster init caasp4 \
    --control-plane caasp4.example.org
cd caasp4
skuba node bootstrap --user sles --sudo \
    --target master1.caasp4.example.org master1
skuba node join --role worker --user sles --sudo \
    --target worker1.caasp4.example.org worker1
```

Listing 2: Cluster mit rke up installieren

```
[john@doe]$ rke config
[+] Cluster Level SSH Private Key Path [~.ssh/id_rsa]:
[+] Number of Hosts [1]:
[+] SSH Address of host (1) [none]: 10.0.0.10
...

[+] Cluster Network CIDR [10.42.0.0/16]:
[+] Cluster DNS Service IP [10.43.0.10]:
[+] Add addon manifest URLs or YAML files [no]:

[john@doe]$ rke up
INFO[0000] Building Kubernetes cluster
INFO[0000] [dialer] Setup tunnel for host [johndoe.example.com]
INFO[0000] [state] Found local kube config file, trying to get state from cluster
...
INFO[0027] [ingress] ingress controller nginx is successfully deployed
INFO[0027] [addons] Setting up user addons
INFO[0027] [addons] no user addons defined
INFO[0027] Finished building Kubernetes cluster successfully
```

den Cluster (Listing 2). Nach der Installation liegt im gleichen Verzeichnis eine Datei kube_config_cluster.yml, die Konfigurationsdatei für das Kommando `kubectl`.

RKE kann nicht nur Kubernetes-Cluster installieren, sondern bietet noch weitere Funktionen an, um ein etcd-Back-up anzulegen und im Fehlerfall wiederherzustellen sowie die Zertifikate für einen Kubernetes-Cluster zu verwalten und, bei selbst signierten, auszutauschen.

Um einen Cluster zu aktualisieren, muss man in der zur Installation verwendeten Datei cluster.yml den Eintrag `system_images` auf die gewünschte Version ändern. Ein erneuter Aufruf von `rke up` aktualisiert den Cluster auf die entsprechende Version.

Um für den Disaster-Recovery-Fall vorbereitet zu sein, lassen sich von etcd Snapshots anlegen und in andere, mit RKE installierte Cluster einspielen. Somit müssen Anwendungen nicht in einen leeren Cluster deployt werden und die angebotenen Dienste sind schnell wieder verfügbar.

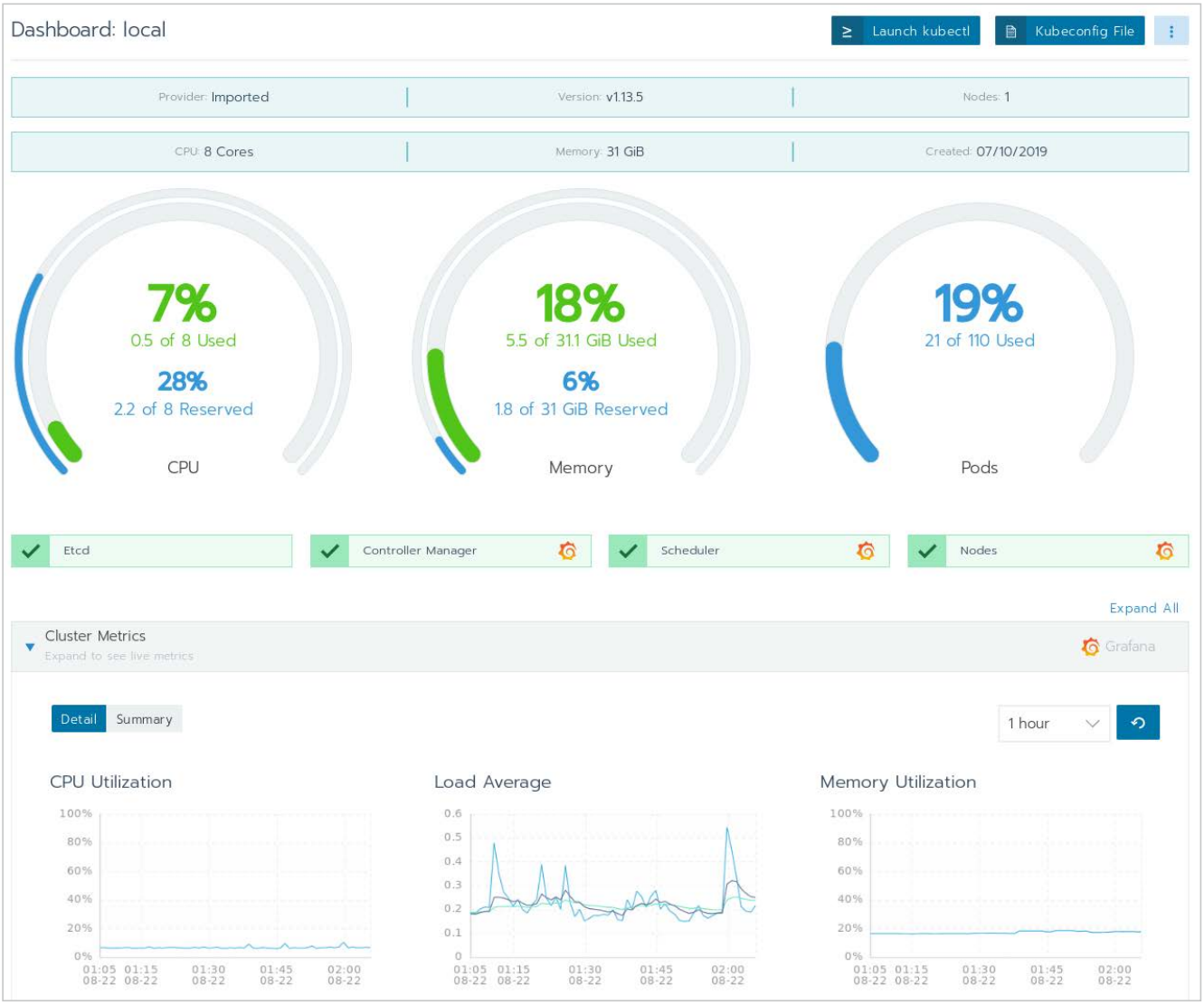
In der Regel verwendet RKE selbst signierte, zehn Jahre gültige Zertifikate. Es lassen sich aber auch von einer eigenen oder offiziellen Zertifizierungsstelle signierte Zertifikate nutzen. Dabei ist jedoch auf deren Gültigkeit zu achten. RKE bietet zu diesem Zweck das Subkommando `cert-rotate` an. Es tauscht die Zertifikate für die einzelnen Komponenten aus und startet alle betroffenen Dienste neu.

Ist der Cluster installiert, kommt das letzte Tool im Rancher-Softwarestack zum Einsatz: eine Webmanagementoberfläche zum Verwalten von Clustern und Workloads (Abbildung 3).

Rancher kann man als einzelnen Docker-Container oder mit Helm als Hochverfügbarkeitsapplikation installieren. Dafür muss Helm eingerichtet und funktionsbereit sein. In diesem Beispiel bekommt Rancher ein Zertifikat von Let's Encrypt ausgestellt. Die für den von RKE installierten Ingress Controller genutzte Domain sollte über das Internet erreichbar sein. Alternativ lässt sich auch ein selbst signiertes oder von einer externen Zertifizierungsstelle ausgestelltes Zertifikat verwenden (Listing 3).

Beim ersten Aufruf der URL muss ein Passwort für den Benutzer `admin` vergeben werden. Anschließend ist die Oberfläche betriebsbereit. Der angezeigte Cluster ist der sogenannte local-Cluster, auf dem Rancher läuft (Abbildung 4).

Zur Verwaltung mehrerer Cluster empfiehlt es sich, Rancher in einem dedizier-



Das Dashboard administriert und verwaltet nicht nur Cluster und Workloads. Es ist auch möglich, Benutzer anzulegen, Berechtigungen auf Cluster, Projekte und andere Kubernetes-Objekte zu verteilen und neue Cluster on Premises oder bei Cloud-Providern zu provisionieren (Abb. 3).

ten Cluster zu installieren, da es Zugriff auf alle Cluster und somit auf alle Container/Workloads hat. Rancher kann sowohl Kubernetes-Cluster in diversen Cloud-Umgebungen als auch VMs auf verschiedenen Infrastrukturprovidern starten und Kubernetes dort installieren. Außerdem ist es möglich, selbst erstellte Kubernetes-Cluster in Rancher zu importieren und zu verwalten.

Kommunikation mit den Clustern

Rancher bietet mehrere Benutzerberechtigungen. Auch können Benutzer auf verschiedenen Clustern berechtigt werden, um dort Workloads zu starten, Volumes zu erstellen oder um Ingress-Ressourcen anzulegen. Zu diesem Zweck werden lokale Benutzer mit Authentifizierungsprovidern wie Active Directory, OpenLDAP oder Okta verbunden. Die Benutzer erhalten je nach Berechtigung Zugriff auf die in Rancher verwalteten Kubernetes-Cluster. Eine zentrale Rancher-API ermöglicht es, mit allen verwalteten Kubernetes-Clustern zu kommunizieren.

Im Catalog, einer Art Helm-App-Store, kann man abhängig von eingebundenen Helm-Repos die Applikationen über die Weboberfläche installieren (Abbildung 5). Auch das Zusammenbauen einer Datei values.yml entfällt, da Rancher die Möglichkeit bietet, die Variablen für das Helm-Chart über die Weboberfläche zu konfigurieren. Eigene Applikationen lassen sich auch deployen, und zwar über den Reiter „Workloads“. Ein Klick auf den Button Deploy ruft einen Wizard auf, der sich ausfüllen lässt (Abbildung 6). Das Erstellen von YAML-Dateien zum Deployment von Applikationen entfällt somit.

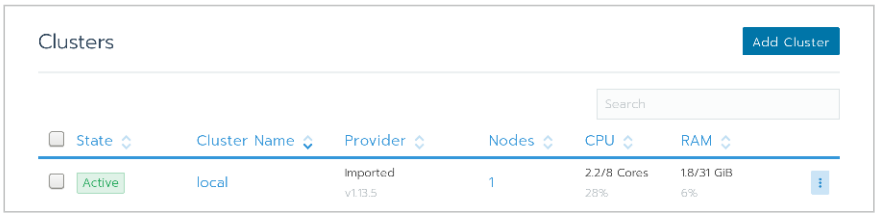
Konsolenliebhaber können mit kubectl gegen die Rancher-API arbeiten. Eine passende kubeconfig-Datei stellt Rancher über die Weboberfläche bereit.

Red Hat OpenShift Container Platform

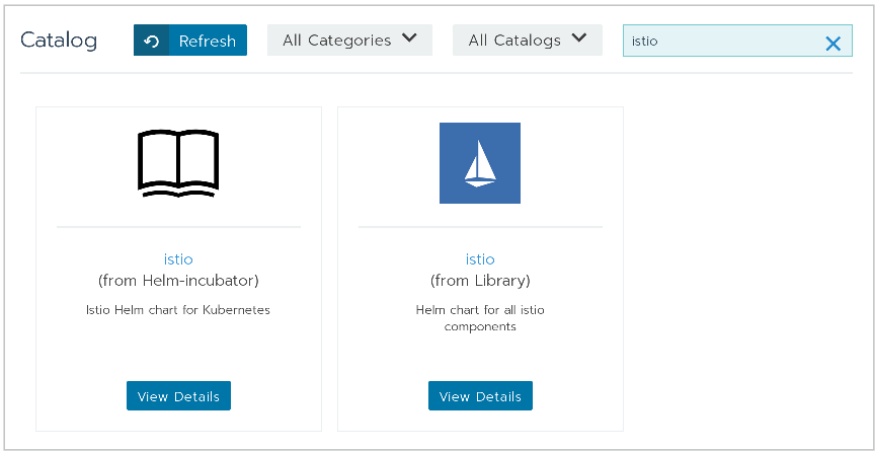
Red Hats Kubernetes-Plattform nutzt statt der Docker Engine die aus dem Kubernetes-Projekt heraus entwickelte Standardimplementierung des Container-Runtime-Interface CRI-O (Abbildung 7). Red Hat hat CRI-O speziell auf und für Kubernetes geschaffen. Es enthält nur die für Kubernetes benötigten Funktionen und Schnittstellen. Analog zu Docker gibt es bei CRI-O das Kommandozeilentool Podman, mit dem sich CRI-O auch ohne

Listing 3: Installation von Rancher

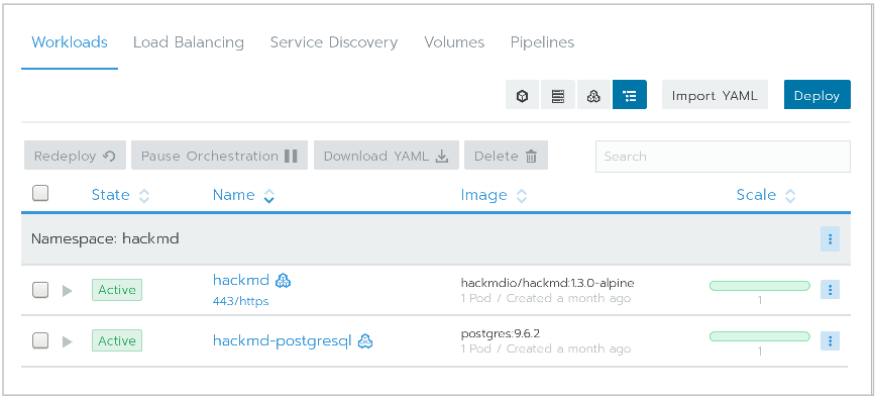
```
helm repo add rancher-stable \
  https://releases.rancher.com/server-charts/stable
kubectl create namespace cattle-system
kubectl apply -f
https://raw.githubusercontent.com/jetstack/cert-manager/release-0.12/deploy/manifests/00- ↵
                                                                                    crds.yaml
kubectl create namespace cert-manager
helm repo add jetstack https://charts.jetstack.io
helm repo update
helm install \
  cert-manager jetstack/cert-manager \
  --namespace cert-manager \
  --version v0.12.0
helm install rancher rancher-stable/rancher \
  --namespace cattle-system \
  --set hostname=rancher.johndoe.example.com \
  --set ingress.tls.source=letsEncrypt \
  --set letsEncrypt.email=johndoe@example.com
```



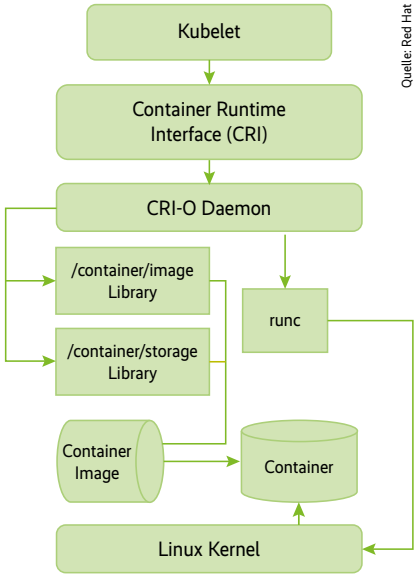
Der erste Cluster „local“ zeigt den Cluster, in dem Rancher installiert wurde. Weitere Cluster sind erst nach einem Import oder erfolgreicher Provisionierung sichtbar (Abb. 4).



Über den Helm-App-Store lassen sich Applikationen relativ einfach installieren. Eigene Helm-Repositories sind zusätzlich einbindbar (Abb. 5).



Unter „Workloads“ sind die bereits ausgerollten Applikationen sichtbar, die im Kubernetes-Cluster laufen. Hierüber lassen sich Logs einsehen, Pods skalieren und Ingress-Ressourcen konfigurieren (Abb. 6).



Quelle: Red Hat

CRI-O verwendet die container/image- und container/storage-Bibliotheken, um Container-Images abzurufen und zu verwalten (Abb. 7).

Kubernetes verwenden lässt. Da die Images für Docker OCI-konform sind (Open Container Initiative), funktionieren sie auch mit CRI-O.

Die OCP-Konsole, die mitgelieferte Weboberfläche, ermöglicht die Steuerung und Überwachung des Clusters. Hierüber hat man volle Kontrolle über die Workload des Clusters bis hin zu Updates der Clustersoftware. Für die Installation liefert Red Hat einen eigenen Installer mit. Er lässt sich für Clusterinstallation und Dateierstellung von einer beliebigen Workstation aus nutzen:

```
openshift-install --dir=/home/tux/ ↵
                ix-cluster create install-config
openshift-install --dir=/home/tux/↵
                        ix-cluster create cluster
```

Beim Deployment in AWS werden benötigte Infrastrukturressourcen wie Netzwerke, Load Balancer und DNS mit erstellt. Der Installer hilft beim Anlegen der erforderlichen Konfigurationsdateien und beim Ausrollen (Listing 4). Die Installation lässt sich derzeit nur in AWS direkt starten, weitere Provider sollen aber mit den nächsten Versionen folgen.

Beim Deployment in AWS greift das Tool auf Terraform zurück, um die komplette Clusterinfrastruktur mit zu erstellen. Im Gegensatz zu CaaS Platform und Rancher ist der Installationsaufwand größer. Lässt sich der Load Balancer noch leicht einrichten, hilft ein „gefakter" DNS-Name nicht weiter. Es müssen vollständige DNS- mit SRV-Einträgen für spezielle Dienste vorliegen.

Mit Red Hat Enterprise Linux CoreOS, kurz RHCOS, hat Red Hat ein spezielles Betriebssystem für die Nutzung in der OpenShift Container Platform 4, das auf den Betrieb von Containern zugeschnitten ist. Konfiguration und Administration erfolgen von einer zentralen Stelle aus und werden über die Clustersoftware auf alle Nodes verteilt. Updates werden über Container-Images ausgeliefert und ermöglichen Transactional Updates und Rollbacks. Statt cloud-init kommt das Werkzeug Ignition zum Einsatz, dessen Funktionsumfang mit cloud-init vergleichbar ist. Anhand einer Konfigurationsdatei lässt sich das System seiner Rolle entsprechend – Master, Worker/Compute – konfigurieren (Listing 5). Diese Datei enthält Informationen über die API und die im Cluster genutzten Zertifikate. Darüber hinaus lassen sich darin auch weitere Systemparameter wie das Netzwerk konfigurieren. Der OpenShift Installer erzeugt diese Konfigurationen.

Alle Bestandteile von OCP sind als Operator realisiert und damit Bestandteil des Kubernetes-Systems. Sie laufen in separaten Namespaces, um eine saubere Trennung zu gewährleisten. Alle innerhalb des Clusters zum Betrieb benötigten Services managt und überwacht der Cluster selbst.

Listing 4: Beispiel einer YAML-Datei für das Cluster-Deployment

```
apiVersion: v1
baseDomain: openshift.test
compute:
- hyperthreading: Enabled
    name: worker
    platform: {}
    replicas: 3
controlPlane:
    hyperthreading: Enabled
    name: master
    platform: {}
    replicas: 3
metadata:
    creationTimestamp: null
    name: ix-cluster
networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
        hostPrefix: 23
    machineCIDR: 10.0.0.0/16
    networkType: OpenShiftSDN
    serviceNetwork:
    - 172.30.0.0/16
platform:
    aws:
        region: eu-central-1
pullSecret: ''
sshKey: |
```

Listing 5: Ignition-Konfiguration für ein Master-System

```
{
    "ignition": {
      "config": {
        "append": [
          {
            "source": "https://api-int.  ↵
              ix-cluster.example.org:22623/ ↵
                                config/master",
            "verification": {}
          }
        ]
      },
      "security": {
        "tls": {
          "certificateAuthorities": [
            {
              "source": "...",
              "verification": {}
            }
          ]
        }
      },
      "timeouts": {},
      "version": "2.2.0"
    },
    "networkd": {},
    "passwd": {},
    "storage": {},
    "systemd": {}
}
```

Fazit

SUSE CaaS Platform 4, Rancher und Red Hat OpenShift Container Platform 4 bieten gute Möglichkeiten zum Aufbau und Verwalten von Kubernetes-Clustern. Welches Produkt das richtige ist, hängt von Situation, Anforderungen und weiteren, geschäftsspezifischen Merkmalen ab.

(avr@ix.de)

Quellen

Alle Listings zum Download: ix.de/z1cj

Christian Rost

ist Linux-Consultant und -Trainer bei B1 Systems und beschäftigt sich mit hochverfügbaren Containerumgebungen, Elastic Stack, ownCloud sowie Konfigurationsmangement mit Puppet und Chef.

Thorsten Schifferdecker

arbeitet als Linux-Consultant und -Trainer für B1 Systems. Schwerpunkt seiner Arbeit sind Administration und Konfiguration Open-Source-basierter, hochverfügbarer Virtualisierungs- und Containerumgebungen.

Christian Schneemann

ist Linux-Consultant und -Trainer bei B1 Systems und beschäftigt sich mit Monitoring, Systemmanagement und Open Build Service.