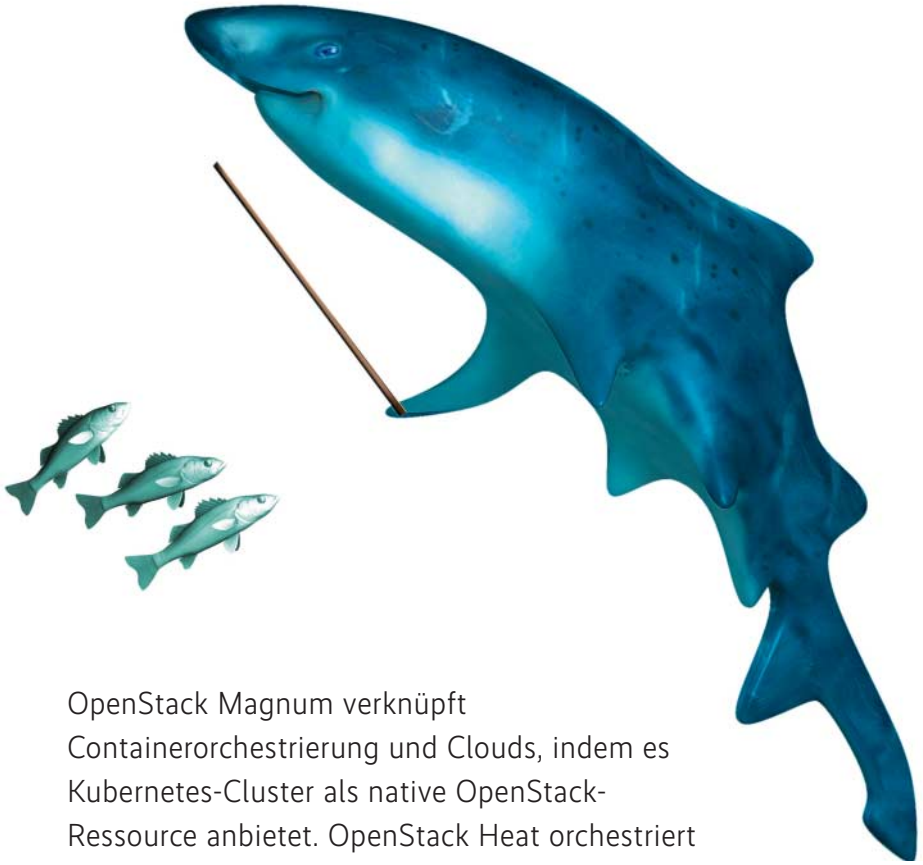
Containerorchestrierung in der Cloud mit OpenStack Magnum

Konzertmeister

David Rabel, Christian Berendt



OpenStack Magnum verknüpft Containerorchestrierung und Clouds, indem es Kubernetes-Cluster als native OpenStack-Ressource anbietet. OpenStack Heat orchestriert dabei virtuelle Maschinen und Netzwerkressourcen, um einen vollwertigen Kubernetes-Cluster zur Verfügung zu stellen.

iX-TRACT

- Magnum ist ein vom OpenStack-Containers-Team entwickelter OpenStack-API-Service. Er soll Container Orchestration Engines wie Docker Swarm, Apache Mesos und Kubernetes als First-Class-Ressourcen in OpenStack verfügbar machen.
- Zusammen mit der Komponente Heat orchestriert Magnum ein Betriebssystem-Image, das Docker und Container Orchestration Engines enthält.
- Mit Heat Orchestration Templates lassen sich Vorlagen definieren. Magnum verwendet sie, um Cluster aus virtuellen Maschinen aufzubauen.

Eine wachsende Zahl von Unternehmen setzt für ihre IT-Infrastruktur auf die Cloud. Die Konsolidierung von Hardware und damit verbundene Kostenersparnisse sowie die einfache und praktisch unbegrenzte Verfügbarkeit virtueller Ressourcen machen verschiedene Angebote von Public-Cloud-Anbietern oder selbst betriebene Private-Cloud-Umgebungen sehr attraktiv. In der Open-Source-Welt ist OpenStack das größte, aber keineswegs das einzige Softwareprojekt, mit dem man eine solche Cloud betreiben kann.

Auf der anderen Seite feiern Cluster Orchestration Engines (COEs) wie das ursprünglich von Google entwickelte Kubernetes (K8s) in den letzten Jahren ihren Siegeszug. Nach der alten Unix-Philosophie, dass ein Programm nur eine Aufgabe übernehmen soll, setzt sich mehr und mehr der Architekturentwurf der Microservices durch. Anstelle großer monolithischer Software werden Dienste in möglichst kleine abgeschlossene Einheiten aufgespalten, die über fest definierte Schnittstellen miteinander kommunizieren. Dieses Konzept passt genau zu Cluster Orchestration Engines, die die einzelnen Dienste in Containern beliebig oft repliziert auf einen Cluster aus Servern verteilen und die Kommunikation unter den Containern und nach außen sicherstellen.

Die Cloud wird angesichts der raschen Entwicklung von Cluster Orchestration Engines bereits von manchen als abgelöst angesehen. Wie man sieht, sind beide Konzepte verwandt, haben aber unterschiedliche Intentionen und können sich hervorragend ergänzen. Die Möglichkeit, einen Kubernetes-Cluster aus virtuellen Maschinen in einer Public Cloud aufzubauen, findet sich auch bei OpenStack.

Master und Minions

Kubernetes ist eine Container Orchestration Engine aus dem Hause Google, die mittlerweile unter der Schirmherrschaft der Cloud Native Computing Foundation (CNCF) weiterentwickelt wird. Mit ihrer Hilfe lassen sich Container nach vorher festgelegter Definition des gewünschten Zustands auf einen Cluster von Nodes verteilen. Dabei sorgt Kubernetes dafür, dass die Container in ausreichender Zahl vorhanden sind, miteinander kommunizieren können und von außen erreichbar sind. Sollten einzelne Container oder ein ganzer Node abstürzen, stellt Kubernetes automatisch den gewünschten Zustand wieder her. Die kleinste verwaltete Einheit sind dabei nicht einzelne Container,

Listing 1: Mit einem einfachen Template lässt sich eine virtuelle Maschine erstellen

```
heat_template_version: 2017-09-01

description: Hello World

resources:
  server:
    type: OS::Nova::Server
    properties:
      image: "Cirros 0.4.0"
      flavor: 1C-1GB-10GB
      networks:
        - network: net-to-public-sandbox
```

sondern sogenannte Pods (englisch Schote, Gehäuse, Herde), die aus mehreren zusammengehörenden Containern bestehen. Sie teilen sich einen gemeinsamen Namensraum und Ressourcen.

Ein Kubernetes-Cluster besteht aus mindestens einem Master und vielen Nodes, auch als Minions bekannt. Der Master ist häufig redundant und auf Hochverfügbarkeit ausgelegt. Er stellt das Gehirn des Clusters dar. Hier laufen der API-Server als zentrale Schnittstelle und die Managerdienste, die permanent den Zustand des Clusters überwachen und gegebenenfalls anpassen. Zusätzlich beherbergt der Master weitere Dienste wie einen Scheduler oder den verteilten Key Value Store etcd, falls dieser nicht separat läuft.

Die verwalteten Container platziert man auf den Nodes, den Arbeitstieren des Clusters. Hier läuft jeweils ein kubelet-Dienst, der Befehle vom Master entgegennimmt und umsetzt, weiterhin ein kube-Proxy, der den äußeren Zugriff auf die lokalen Pods eines Nodes erlaubt.

OpenStack stellt einen Softwarestack zum Betrieb einer Infrastructure-as-a-Service-Cloud (IaaS) dar. Damit lassen sich primär typische Infrastrukturressourcen wie Server, Netzwerke oder Storage virtualisieren, welche die Nutzer über eine API oder ein Webinterface provisionieren können. Mit seiner Vielzahl an Diensten ist OpenStack erweiterbar und bietet auch Ressourcen an, die eher in den Bereich Platform as a Service (PaaS) fallen, also über das Bereitstellen virtueller Infrastruktur hinausgehen.

Effizient arbeiten

Eine dieser Komponenten ist Heat, die sich zur Orchestrierung von OpenStack-Ressourcen nutzen lässt – nicht zu verwechseln mit Containerorchestrierung. Mit parametrisierbaren Template-Dateien, sogenannten Heat Orchestration Templates (HOTs), lassen sich Vorlagen definieren, mit denen man sogenannte Heat Stacks erstellen kann.

Ein Beispiel: Für eine Testumgebung sind regelmäßig mehrere virtuelle Maschinen zu erstellen und zu löschen, die über ein privates virtuelles Netzwerk miteinander kommunizieren. Eine dieser virtuellen Maschinen soll über eine externe IP-Adresse erreichbar sein. Die Speicherung anfallender Daten erfolgt persistent auf virtuellen Block Devices. Anstatt jedes Mal virtuelle Maschinen, Netzwerk, Floating IP, Security Groups und Volumes manuell anzulegen, kann man sie einma-

Selbst ein einfacher Magnum-Cluster besteht aus vielen einzelnen Ressourcen.

Listing 2: Der Treiber für Kubernetes auf Fedora Atomic und virtuellen Maschinen *(template_def.py)*

```
class AtomicK8sTemplateDefinition(kftd.K8sFedoraTemplateDefinition):
  """Kubernetes template for a Fedora Atomic VM."""

  provides = [
    {'server_type': 'vm',
     'os': 'fedora-atomic',
     'coe': 'kubernetes'},
  ]
```

Listing 3: Kern des Clustertreibers ist ein Heat Orchestration Template *kubecluster.yaml (template_def.py)*

```
  @property
  def template_path(self):
    return os.path.join(os.path.dirname(os.path.realpath(__file__)),
      'templates/kubecluster.yaml')
```

lig in einem HOT definieren und dieses mehrmals verwenden. So lassen sich alle benötigten Ressourcen auf einmal erstellen oder löschen (Listing 1).

Eine weitere Komponente von OpenStack ist Magnum, das Heat Orchestration Templates verwendet, um Cluster aus virtuellen Maschinen aufzubauen, auf denen dann wiederum eine Cluster Orchestration Engine läuft. Auch von der OpenStack-Komponente Ironic verwaltete Bare-Metal-Server lassen sich statt virtueller Maschinen verwenden. Außer Kubernetes können auch Docker Swarm oder Apache Mesos zur Containerorchestrierung zum Einsatz kommen. Die Anzahl der zu erstellenden virtuellen Maschinen für Nodes und Master ist dabei über Parameter einstellbar. Die Abbildung zeigt einen simplen Magnum-Cluster im OpenStack-Dashboard, der bereits eine hohe Komplexität aufweist.

Bevor man einen Magnum-Cluster erstellt, empfiehlt es sich, zunächst ein Cluster-Template zu definieren. Cluster-Templates sind nicht mit Heat Orchestration Templates zu verwechseln. Sie fungieren lediglich als Schablone für einen Magnum-Cluster, in dem gewisse Parameter zu definieren sind. So wird im Cluster-Template unter anderem festgelegt, welche Cluster Orchestration Engine zum Einsatz kommt und welches Basis-Image für die virtuellen Maschinen verwendet werden soll.

Herausforderung Clustertreiber

Ein Clustertreiber enthält die benötigten Heat Orchestration Templates zum Aufbau eines Clusters. Er stellt die Integration einer Kombination aus Cluster Orchestration Engine, Basis-Image und Deployment-Methode bereit. Beispielsweise gibt es einen Clustertreiber für die Kombination von Kubernetes auf einem Fedora Atomic Image und virtuellen Maschinen (Listing 2). Ein anderer Clustertreiber kommt für Kubernetes auf CoreOS und virtuellen Maschinen zum Einsatz.

Neben Python-Code besteht ein Clustertreiber aus einem Heat Orchestration Template, das rekursiv weitere Heat Orchestration Templates einbindet (Listing 3 und 4). Mit Heat Orchestration Templates lassen sich Umgebungsvariablen und Shell-Skripte mit Ressourcen vom Typ *SoftwareConfig* übergeben (Listing 5). Diese Möglichkeit nutzt Magnum, um die virtuellen Maschinen im Cluster zu konfigurieren.

Die Implementierung von Clustertreibern ist relativ statisch. So ist es nicht vorgesehen, unterschiedliche Clustertreiber für das gleiche Basis-Image parallel zu betreiben. Das kann aber notwendig werden, wenn man unterschiedliche Versionen des gleichen Basis-Images hat. Auf den ersten Blick scheint das unproblematisch, weil man immer die neueste stabile Version der zugrunde liegenden Linux-Distribution verwenden möchte. Probleme gibt es, wenn der von Magnum mitgelieferte Clustertreiber nicht mehr mit aktuellen Basis-Images funktioniert oder wenn bestimmte Features noch nicht Teil des Clustertreibers sind.

Da die Clustertreiber ein fester Teil von Magnum sind, ist ein Ersetzen oder Verändern zwar möglich, aber umständlich. Die Änderungen sind manuell auf allen OpenStack-Controller-Nodes auszurollen, alte und neue Version können nicht nebeneinander existieren und gegebenenfalls funkt das für das OpenStack-Deployment verwendete Framework dazwischen.

Listing 4: Weitere Heat Orchestration Templates bindet man rekursiv ein *(kubecluster.yaml)*

```
  kube_masters:
    type: OS::Heat::ResourceGroup
    depends_on:
      - extrouter_inside
    properties:
      count: {get_param: number_of_masters}
      resource_def:
        type: kubemaster.yaml
...
  kube_minions:
    type: OS::Heat::ResourceGroup
    depends_on:
      - extrouter_inside
    properties:
      count: {get_param: number_of_minions}
      removal_policies: [{resource_list: {get_param: minions_to_remove}}]
      resource_def:
        type: kubeminion.yaml
```

Listing 5: Einbinden eines Shell-Skripts als *SoftwareConfig* *(kubemaster.yaml)*

```
configure_etcd:
  type: OS::Heat::SoftwareConfig
  properties:
    group: ungrouped
    config: {get_file: ../../common/templates/kubernetes/fragments/configure-etcd.sh}
```

Fazit

Trotz einiger Schwächen ist OpenStack Magnum ein mächtiges Werkzeug, um schnell und bequem Kubernetes-Cluster auf OpenStack zu deployen. Einmal richtig eingerichtet, bietet Magnum außerdem den Komfort, mit wenigen Klicks im OpenStack-Dashboard Horizon einen fertigen Cluster zu provisionieren. Es soll aber nicht unerwähnt bleiben, dass es spezialisierte und deutlich ausgereiftere Deployment-Frameworks für Kubernetes gibt. Dazu zählt Kubespray, das die Nutzung von Kubernetes auf vielen Distributionen und Cloud-Lösungen – darunter auch OpenStack – ermöglicht und das Thema eines Artikels in einer der nächsten *iX*-Ausgaben sein wird.

(nb@ix.de)

David Rabel

ist bei B1 Systems als Consultant und Trainer im Linux- und OpenStack-Umfeld tätig.

Christian Berendt

betreut als Cloud Solution Architect bei B1 Systems das Thema OpenStack.