nine
cloud navigators

Einstieg in Kubernetes

Eine 10-Punkte-Checkliste

Einstieg in Kubernetes
Eine 10-Punkte-Checkliste

1. Sind Ihre Container-Image-Erstellung und die Veröffentlichung automatisiert?

Bei Kubernetes wird Ihr Container-Image entweder aus einem privaten oder öffentlichen Container-Registry gezogen. Wenn Sie Ihren Test- und Deployment-Prozess also noch nicht automatisiert haben, dann ist es ein guter erster Schritt, Ihre Images nach dem Commit im Git Repository automatisch per Push zu verschieben. Sicherlich haben Sie bereits ein Image entwickelt und es in einem Registry veröffentlicht. Die nötigen Commands kennen Sie daher vermutlich bereits, sodass Sie sich darauf konzentrieren können, die richtigen Tools für die Automatisierung Ihres Deployment-Workflows zu finden. Dabei können Sie mit dem Image-Push beginnen und rückwärts die Entwicklung, das Testing, den Code Review und weitere Schritte in Ihrem Workflow betrachten, die möglicherweise noch anzugehen sind.

2. Wie kompakt ist Ihr Container? Haben Sie alles Unnötige entfernt?

Das Container-Image Ihrer Anwendung wird im Rahmen Ihrer Pipeline von verschiedenen Systemen heruntergeladen. Es kann beispielsweise für das Testing oder für Integration Deployments sowie von CI/CD-Systemen oder weiteren Systemen benötigt werden. Um sowohl die Zeit zu verringern, die Sie für die Entwicklung einer neuen Version Ihrer Anwendung brauchen, als auch die Kosten der Datenübertragung zu optimieren, sollten Sie alles Unnötige aus Ihrem Container entfernen. Dazu gehört möglicherweise auch die Verwendung eines Container-optimierten Betriebssystems im Container selbst, um seine Grösse weiter zu minimieren. Ausserdem kann die Reduzierung der Anzahl an Image-Schichten die Angriffsfläche reduzieren und Ihre Images somit sicherer machen. Wir empfehlen daher, Ihre Container so kompakt wie möglich zu gestalten, indem Sie zum Beispiel Multi-Stage Builds verwenden.

3. Haben Sie Ihre Kubernetes-Konfiguration festgelegt?

Wenn Sie noch keine eigenen Kubernetes-Manifeste geschrieben haben, sollten Sie die verschiedenen möglichen Vorlage-Sprachen betrachten. Wenn Sie sich bereits mit Docker-Compose auskennen, können Sie Kompose verwenden, um die erstellte Datei in die von Ihnen benötigte Kubernetes-Konfiguration umzuwandeln. Wenn Sie gerade erst beginnen, mit Kubernetes zu arbeiten, kann dies ein sinnvoller erster Schritt sein. Allerdings ist dabei wichtig zu verstehen, dass Ihre Kubernetes-Konfiguration ein wesentlicher Bestandteil Ihrer Anwendung ist und entsprechend behandelt werden sollte. Sie sollten also sichergehen, dass Sie jede Zeile verstehen. Kubernetes verwendet eine deklarative Konfiguration und es besteht eine ganze Reihe an Vorlage-Sprachen, wie z. B. Jsonnet, die Ihnen dabei helfen können, die von Ihnen benötigten Dateien zu erstellen. Auch komplexere Sprachen wie Cue, Pulumi oder cdk8s sind möglicherweise einen Blick wert. Sie ermöglichen die Definition von Konfigurationen im Code und produzieren ein Format, das für die Arbeit über verschiedene Clouds, Sprachen und Tools hinweg geeignet ist.

4. Haben Sie Ihre Konfigurationen in einem Helm Chart verpackt?

Sobald Sie das Deployment weiterer Services angehen, die von Ihrer Anwendung benötigt werden — selbst wenn es nur um das Testing geht — werden Ihnen fast zwangsläufig Helm Charts begegnen. Ein Helm Chart stellt eine Sammlung an Dateien dar, die ein Set an miteinander verbundenen Kubernetes-Ressourcen definieren. Dabei kann es sich um verschiedene Dinge handeln — vom einfachen Service bis zum vollumfänglichen Web-Anwendungs-Stack. Dementsprechend sollten Sie sich auf jeden Fall mit den dahinter stehenden Konzepten vertraut machen. Die Erstellung eines Helm Charts ist jedoch sicherlich eine gute Idee, da es das Deployment komplexer Anwendungen vereinfacht und Helm ausserdem aus dem Container-Bereich nicht mehr wegzudenken ist.

5. Kennen Sie das Performance-Profil Ihrer Anwendung?

Um Kubernetes optimal zu nutzen, sollten Sie verstehen, welche Leistung Ihre Anwendung bei voller Auslastung erbringt und was die Indikatoren für die gewöhnliche und für eine ungewöhnliche Performance sind. Wenn Sie zumindest das CPU- und RAM-Profil Ihrer Anwendung kennen, können Sie Ihre Ressourcen-Anfragen und -Höchstgrenzen richtig konfigurieren. Dadurch können Sie Kosten optimieren, aber auch die Selbstreparatur Ihrer Anwendung voranbringen. Denn ohne die richtige

Konfiguration findet bei einem Problem möglicherweise kein Neustart statt, oder Kubernetes erkennt das normale Anwendungsverhalten als Verletzung der Ressourcen-Anforderungen, was zum CrashLoopBackOff führt. Da es allerdings eher unwahrscheinlich ist, dass Sie das Performance-Profil sofort korrekt konfigurieren, ist es zu empfehlen, die tatsächliche Nutzung Ihrer Anwendung zu beobachten und die benötigten Ressourcen ständig anzupassen.

6. *Haben Sie die Skalierung Ihrer Anwendung in Betracht gezogen?*

Einer der wichtigsten und stärksten Aspekte von Kubernetes ist die einfache horizontale Skalierung Ihrer Container. Sobald Sie das Performance-Profil Ihrer Container kennen, sollten Sie die Verwendung eines HorizontalPodAutoscalers in Betracht ziehen, um Ihre Anwendung basierend auf dem aktuellen Traffic zu skalieren. Statt eine Traffic-Überlastung zu fürchten, wie es früher vielleicht einmal der Fall war, können Sie Ihren Kubernetes-Cluster für sich arbeiten lassen.

7. *Wissen Sie, wie Ihre Anwendung auf Neustarts reagiert?*

Kubernetes wird Ihre Container ständig starten, anhalten und neu starten – und dabei handelt es sich um ein Feature, nicht um einen Bug! Ein Neustart erfolgt beispielsweise, wenn Kubernetes den Eindruck hat, dass Ihre Anwendung abstürzt. Oder Ihre Applikation wird auf einen neuen Node verschoben, wenn zum Beispiel. Instandhaltungsarbeiten ausgeführt werden, Sie selbst Ihre Node Pools verändern oder auch hoch- oder herunterskalieren wollen. Sie sollten also darüber nachdenken, was passiert, wenn Ihre Anwendung neu startet. Dabei sollten Sie sicherstellen, dass beispielsweise nicht jedes Mal Database Schema Upgrades vorgenommen werden oder andere Vorgänge laufen, die den Neustart verzögern und Ihre Kunden davon abhalten, die Applikation zu erreichen. Auch für die Skalierung sollten Sie sich über dieses Thema Gedanken machen, da die Instanzen so schnell wie möglich angehalten und wieder neu gestartet werden sollten, um die dynamischen Funktionalitäten des Kubernetes-Schedulers optimal zu nutzen.

8. *Haben Sie sich Gedanken um die Verfügbarkeit Ihrer Anwendung gemacht?*

Sowohl physische als auch virtuelle Maschinen stürzen ab – und sie werden dies auch in Zukunft weiter tun. Es führt kein Weg daran vorbei. Allerdings können Sie Ihre Container so konfigurieren, dass eine hohe Verfügbarkeit besteht, d. h. kein Single Point of Failure existiert. So ist es wenig nützlich, 10 Kopien Ihrer Container auf einem einzigen Node zu betreiben, wenn Sie einen Cluster mit mehreren Nodes haben. Stattdessen sollten Sie Ihre Duplikate auf so viele Nodes wie möglich verteilen und so die Robustheit und Verfügbarkeit Ihrer Anwendung erhöhen.

9. *Haben Sie erwägt, wie Ihre Anwendung auf die Nichtverfügbarkeit anderer Services reagiert?*

Sie sollten möglicherweise auch darüber nachdenken, wie Ihr Service Stack generell auf die Nichtverfügbarkeit anderer Services reagiert. Denn ein Ausfall ist immer möglich – unabhängig von der gewählten Infrastruktur, Architektur oder dem Anbieter. Allerdings ist dieses Thema besonders in Cloud-Umgebungen wichtig. Es sollte daher in Ihrer Entwicklungsphase vorrangig sichergestellt werden, dass Ihre Anwendung in solchen Szenarien resilient ist, für Kunden erreichbar bleibt und keine Daten verloren gehen.

10. *Wissen Sie, wie Sie die Services betreiben, von denen Ihre Anwendung abhängig ist?*

Beim Betrieb von Anwendungen in Kubernetes gibt es viel zu beachten. Auch wenn es bei richtiger Verwendung massive Vorteile mit sich bringt, kann es etwas überwältigend sein, sich um alles selbst zu kümmern. Egal ob Sie Kubernetes gerade erst entdecken oder es schon kennen: Es kann sich lohnen, sich mit einem erfahrenen Partner zusammenzutun, der die für Ihre Anwendung nötigen zusätzlichen Services anbietet und dabei eine gewisse Qualität garantiert. Wenn Sie sich bereits gut auskennen, ist es ein Option, direkt einen Cloud Provider anzufragen und einfach dessen Dienstleistungen nutzen. Wenn Sie allerdings etwas mehr benötigen – z. B. direkten Kontakt mit Entwicklern, zusätzliche Services, Solution Architecture Support, etc. – dann kann es von wesentlichem Vorteil sein, sich an einen Service Provider wie Nine zu wenden, der jahrelange Erfahrung sowohl mit klassischen als auch mit Container-Infrastrukturen bietet.