✓ 100 XP

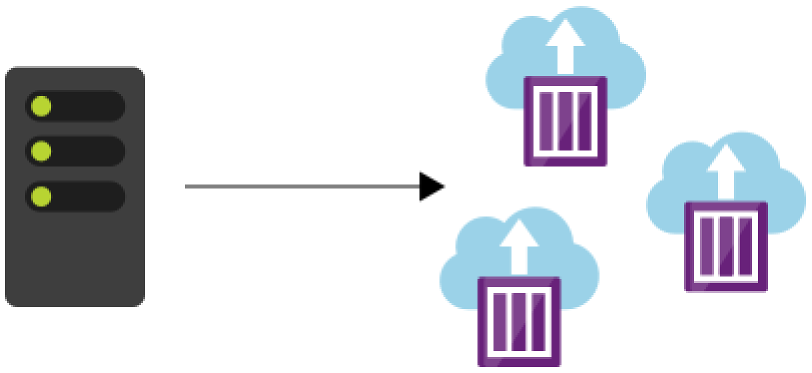
What is Azure Kubernetes Service?

7 minutes

Let's start with a few definitions and a quick tour through Azure Kubernetes Service (AKS). This overview should help you decide whether AKS might be a good fit for your containerization management strategy.

What is a container?

A *container* is an atomic unit of software that packages up code, dependencies, and configuration for a specific application. Containers allow us to split up monolithic applications into individual services that make up the solution. This rearchitecting of our application will enable us to deploy these separate services via containers.

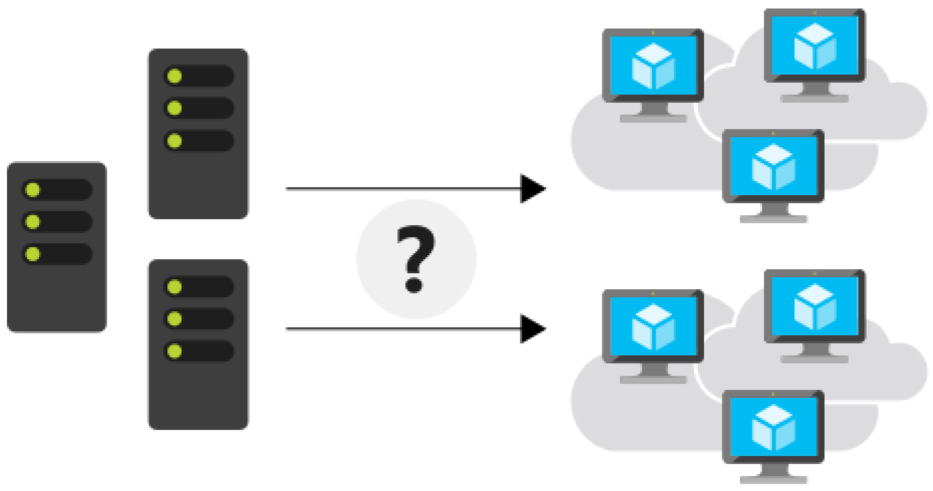


Why use a container?

Suppose your fleet management solution included three major applications:

- A fleet tracking website that includes maps and information about the assets being tracked.
- A data processing service that collects and processes information sent from tracked vehicles.
- An MSSQL database for storing tracking and user information captured from the website.

You realize that you have to scale out your solution to meet customer demand. One option is to deploy a new virtual machine (VM) for each application and then deploy the applications to the VMs. However, doing so will make you responsible for the management of each additional VM. For example, you'll have to make sure the correct operating system (OS) versions and dependencies for each application is installed and configured. You also must make sure you're installing and upgrading the correct versions of the applications. If there are errors, you have to make sure you can roll back the installation with the least amount of disruption to your solution.



The above deployment is cumbersome, sometimes error-prone and doesn't easily scale single services. For example, you might find you can't easily scale only the caching service used in the web application. Containers help solve these types of problems.

The container concept gives us three major benefits:

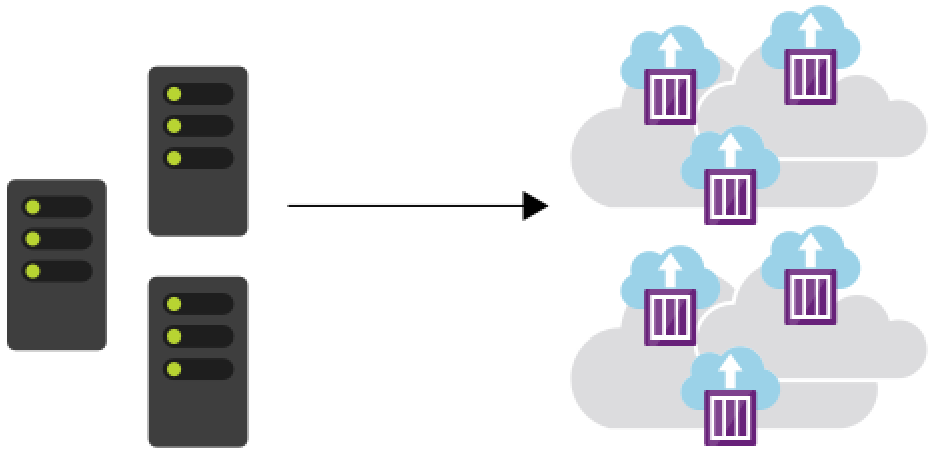
1. **A container is immutable** - the unchanging nature of a container allows it to be deployed and run reliably with the same behavior from one compute environment to another. A container image tested in a QA environment is the same container image deployed to production.

2. **A container is lightweight** - you can think of a container as a VM image, but smaller. A VM image is normally installed on a physical host. The image contains both the OS and the application you want to run. In contrast, a container doesn't need an OS, only the application. The container always relies on the host installed OS for Kernel-specific services. Containers are less resource-intensive, and multiple containers can be installed on the same compute environment.

3. **Container startup is fast** - containers can start up in few seconds instead of minutes, like a VM.

The above benefits make containers a popular choice for developers and IT operations alike.

What is container management?



Even though you can think of containers as VMs, you have to keep in mind that they aren't. A container has a distinct life cycle. It's deployed, started, stopped, and destroyed as requested. This life cycle makes containers disposable and impacts how developers and IT operations should think about the management of large container deployments.

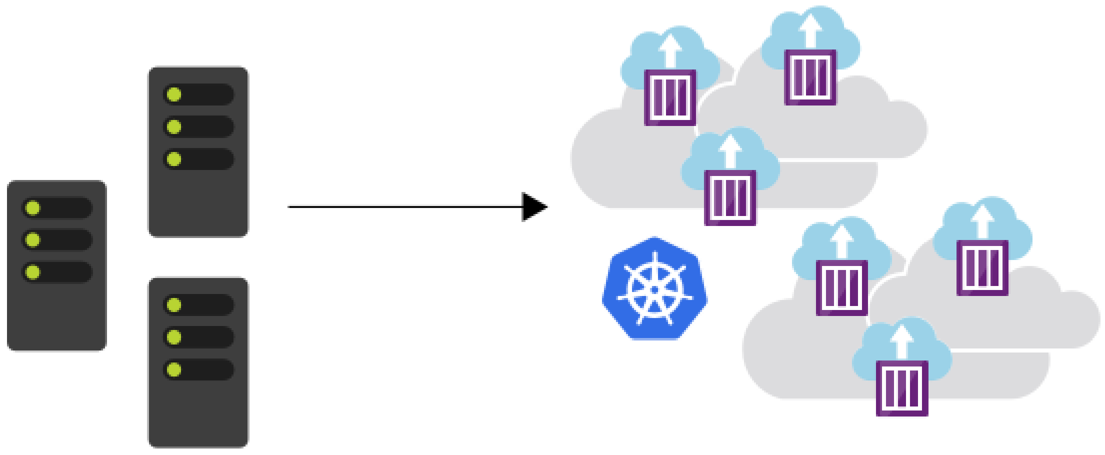
The process of deploying, updating, monitoring, and removing containers introduces many challenges.

Suppose you want to scale your fleet tracking website. You find that at specific times during the day, you need more instances of the site's caching service to manage performance. You can solve this problem by adding additional caching service containers.

Now let's suppose, you need to roll out a new version of your caching service. How do you make sure you update all the containers? How do you remove all the older versioned containers?

These types of questions justify a system to help you manage your container deployment.

What is Kubernetes?



Kubernetes is a portable, extensible open-source platform for automating deployment, scaling, and the management of containerized workloads. Kubernetes abstracts away complex container management and provides us with declarative configuration to orchestrate containers in different compute environments. This orchestration platform gives us the same ease of use and flexibility as with Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) offerings.

Kubernetes allows you to view your data center as one large computer. We don't worry about how and where we deploy our containers, only about deploying and scaling our applications as needed.

However, this view might be slightly misleading as there are a few aspects to keep in mind:

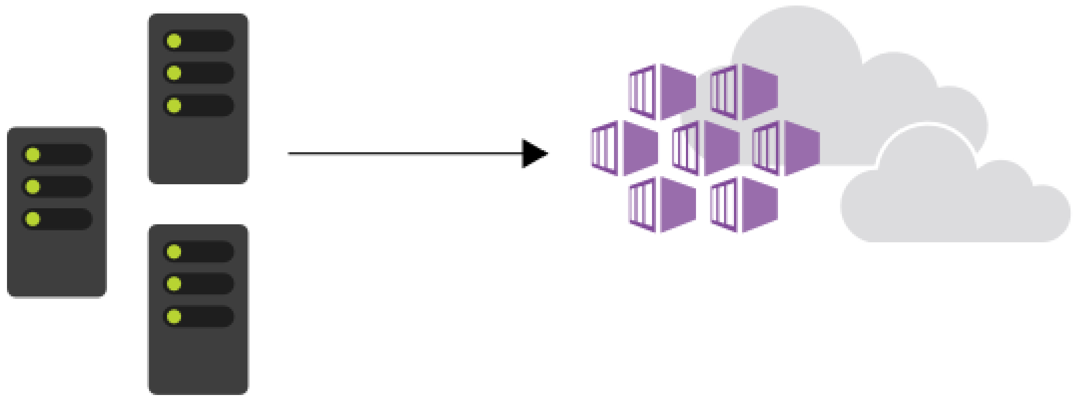
- Kubernetes isn't a full PaaS offering. It operates at the container level and offers only a common set of PaaS features.
- Kubernetes isn't monolithic. It's not a single application that is installed. Aspects such as deployment, scaling, load balancing, logging, and monitoring are all optional. You're responsible for finding the best solution that fits your needs to address these aspects.
- Kubernetes doesn't limit the types of applications that can run. If your application can run in a container, it can run on Kubernetes. Your developers need to understand concepts such as microservices architecture, to make optimal use of container solutions.
- Kubernetes doesn't provide middleware, data-processing frameworks, databases, caches, nor cluster storage systems. All these items are run as containers or as part of another service offering.
- A Kubernetes deployment is configured as a cluster. A cluster consists of at least one master machine and one or more workers machines. For production deployments, the preferred master configuration is a multi-master high availability deployment with three to five replicated masters. These machines can be physical hardware or VMs. Theses worker machines are called nodes or agent nodes.

With all the benefits you receive with Kubernetes, keep in mind that you're responsible for maintaining your Kubernetes cluster. For example, you need to manage OS upgrades and the Kubernetes installation and upgrades. You also manage the hardware configuration of the host machines, such as networking, memory, and storage.

Note

Kubernetes is sometimes abbreviated to **K8s**. The 8 represents the eight characters between the K and the s of the word K[*ubernete*]s.

What is Azure Kubernetes Service?



Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment and makes it simple to deploy and manage containerized applications in Azure. Your AKS environment is enabled with features such as automated updates, self-healing, and easy scaling. The Kubernetes cluster master is managed by Azure and is free. You manage the agent nodes in the cluster and only pay for the VMs on which your nodes run.

You can either create your cluster in the Azure portal or use the Azure CLI. When you create the cluster, you can use Resource Manager templates to automate cluster creation. With these templates, you specify features such as advanced networking, Azure Active Directory (AD) integration, and monitoring. This information is then used to automate the cluster deployment on your behalf.

With AKS, we get the benefits of open-source Kubernetes without the complexity or operational overhead compared to running our own custom Kubernetes cluster.

Next unit: How Azure Kubernetes Service works

Continue >