Kubernetes basics for sysadmins

Learn when Kubernetes can be effectively used and how the containers it manages might be better than virtual machines.

Posted: October 21, 2020 | Shashank Nandishwar Hegde (Red Hat, Sudoer)

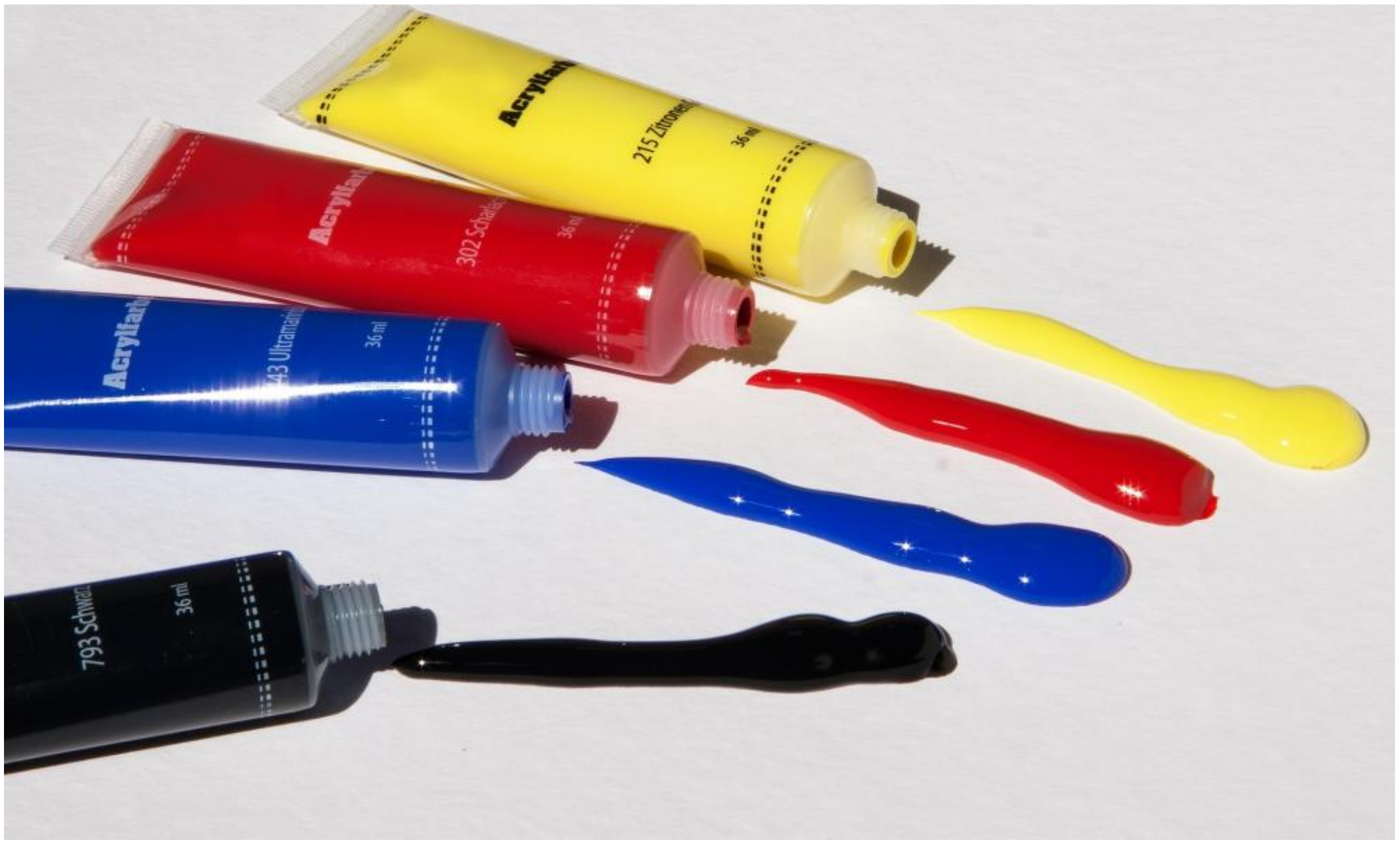
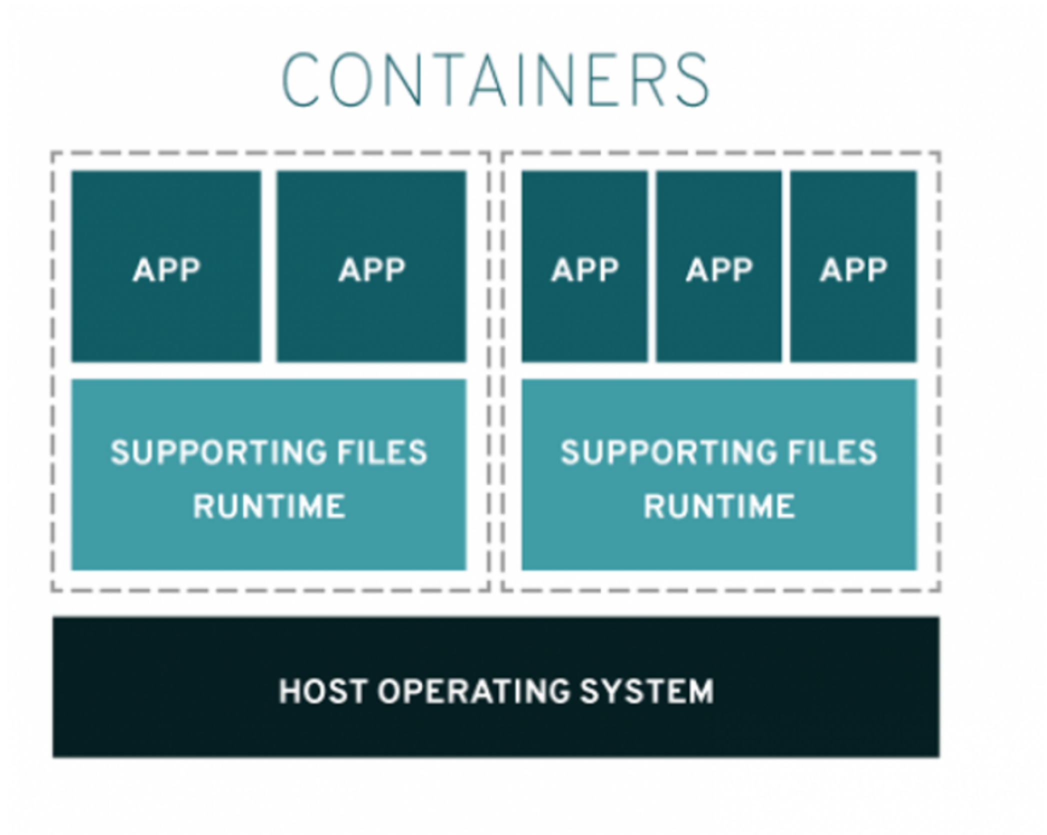


Image by Thanks for your Like • donations welcome from Pixabay

In this first of a two-part series, this article looks at similarities and differences in containers, virtual machines, and the pros and cons of each technology. I also look at Kubernetes (often written as K8s) and why it came into existence.

Containers

In the simplest terms, think of containers as standard boxes of software that can be used as a standalone deployment unit in any infrastructure. Containers come bundled with code and all dependencies inside. They are lightweight, standalone, and contain all runtimes, settings, and system tools needed to run applications.

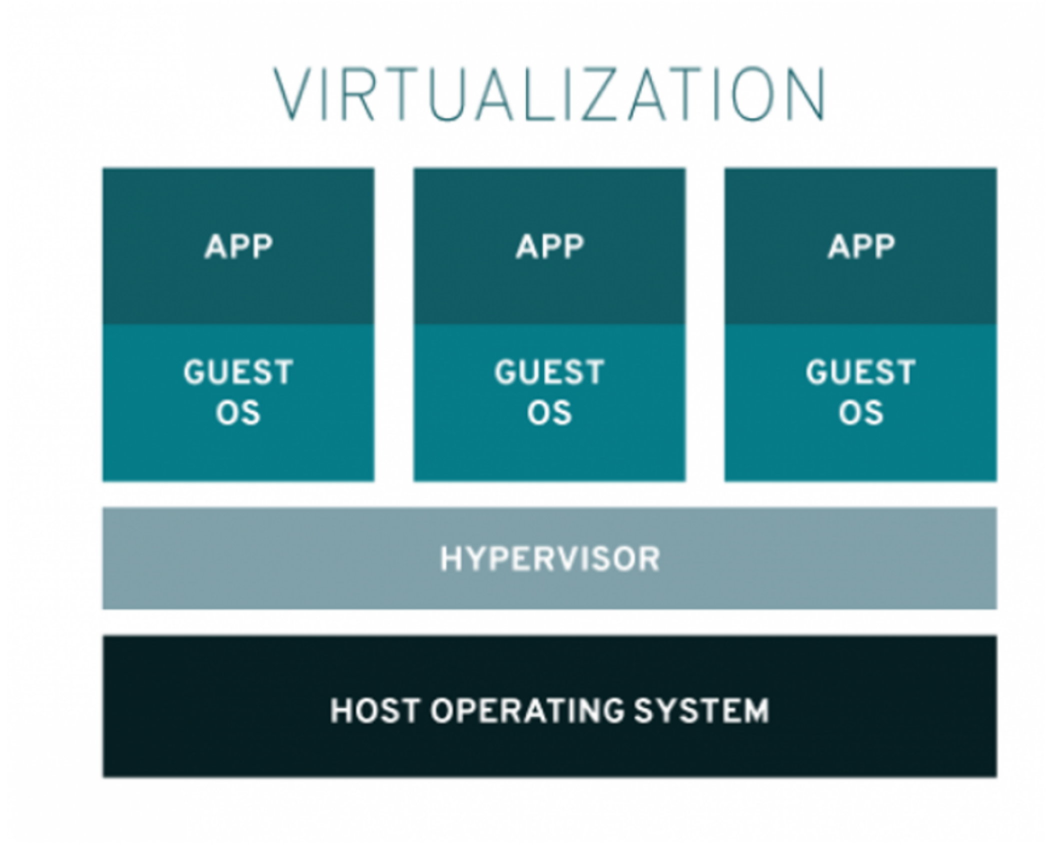


Virtual machines

Virtual machines (VMs) are sandboxed programs (guest machines) that behave just like physical machines, and they run on physical hardware and operating systems (host machines). Virtualization is like creating multiple machines inside a machine. The software inside the virtual machine doesn't interfere with the host operating system. This makes VMs ideal for testing beta applications, dissecting and studying virus-infected files, and exploring any other software you don't want to place on a physical system. Virtual machine host systems are designed and equipped to accommodate multiple virtual machines—from a few to several dozen depending on the technologies and hardware involved.

[Readers also liked: [Automate virtual machine deployment with Ansible: Design]()]

Each VM uses its own virtual CPU, memory, hard disk, NICs (network interface cards), which map to physical machine resources. Basically, VMs are test environments which can save you money, time, and other resources because of their disposability, rapid build times, and portability between hosts. VMs provide safe environments to test the latest software and applications without the risk of harming your physical host system.



Note: While VMs and hosts are logically separated from one another, VMs themselves are no more or less secure than a standalone operating system on a physical machine. And networked systems, physical or virtual, are equally vulnerable to network-based attacks and challenges.

There are key differences between VMs and containers

Now that you know the definition and some use cases for VMs and containers, you can compare the main differences between VMs and containers in the table below:

Virtual Machine	Container
Hypervisor virtualizes the host operating system, which results in virtual machines containing a complete guest OS.	Containers virtualize the operating systems, which then only needs applications and their dependencies and libraries.
VMs provide isolation from the host operating system, giving a favorable isolated testing environment.	Containers are isolated from the host and other containers on the system.
Hypervisor separates the resources of the physical machine from the virtual machine.	Everything on a container is preserved in a code-based file called an image.
VMs require more system resources because they use a complete OS with a kernel, CPU, hard drives, and NICs.	Containers can be better tailored to your own application needs, thus consuming fewer system resources.
Used in traditional application testing environments.	Containers are used in microservices and Kubernetes environments.

Kubernetes

Kubernetes is an open source system for automating deployment, scaling, and managing containerized applications. K8s, formerly known as Borg, was used by Google before it became an open source project. It is now maintained and managed by the CNCF (Cloud Native Computing Foundation). K8s group containers together into a logical unit called a pod. Pods form applications. Pods also provide easy management and discovery.

Why should we use Kubernetes?

As microservices and cloud began to rise, there was a need to manage thousands of containers running on a system. This was due in part to high availability requirements and other necessities.

The main advantages of Kubernetes include:

- Automated rollouts and rollbacks: Helps with automatic rollouts of changes while making sure all the pods don't go down at the same time. It monitors health and can be used to roll back changes.
- Service discovery and load balancing: Helps give the pods (which are wrappers for containers) their own IP addresses and a single DNS name. K8s also helps balance the load.
- Service topology: Helps route the service traffic based on cluster topology.
- Storage orchestration: Mounts the storage system of your choice, such as local storage or cloud provider storage services.
- Scaling: Scales your application up or down based on CPU usage via CLI or a GUI.
- Self-healing: Restarts failed containers, kills containers that don't respond to user requests, and helps bring up containers that are not responding to your health check.

Use cases, efficiency, and cost reduction

You've seen the advantages of K8s but now I'll show you why any company can benefit from Kubernetes. Companies need their products to be available to their customers quickly and reliably. Kubernetes helps accomplish exactly that. It breaks your components into microservices, which can be the focus of smaller teams. The parts are later integrated together via APIs. In this way, each team has a smaller focus and makes the overall release process faster.

Improved scalability

In modern times, where the user load changes dynamically, you never know when downtime will occur. Let's say you have a flash sale scheduled on a particular day. On that day, the availability of your application to the customers is a must. Kubernetes helps in scaling the pods, which maintain the functionality. Also, when the sale is over, it scales back the pods, which provides automatic cost reduction.

Available for multi-cloud environments

Kubernetes' biggest advantage is that it helps enterprises deploy their apps on various public and private cloud environments. It allows you to use hybrid clouds, which avoids vendor lock-in. It also allows enterprises to migrate their workloads, giving them the maximum return on investment (ROI).

[Learn the basics of using Kubernetes in this free cheat sheet.]

Wrap up

This single article can't cover all the use cases for Kubernetes but now you have some idea of how K8s came to be, what it is, and what it can do for you.