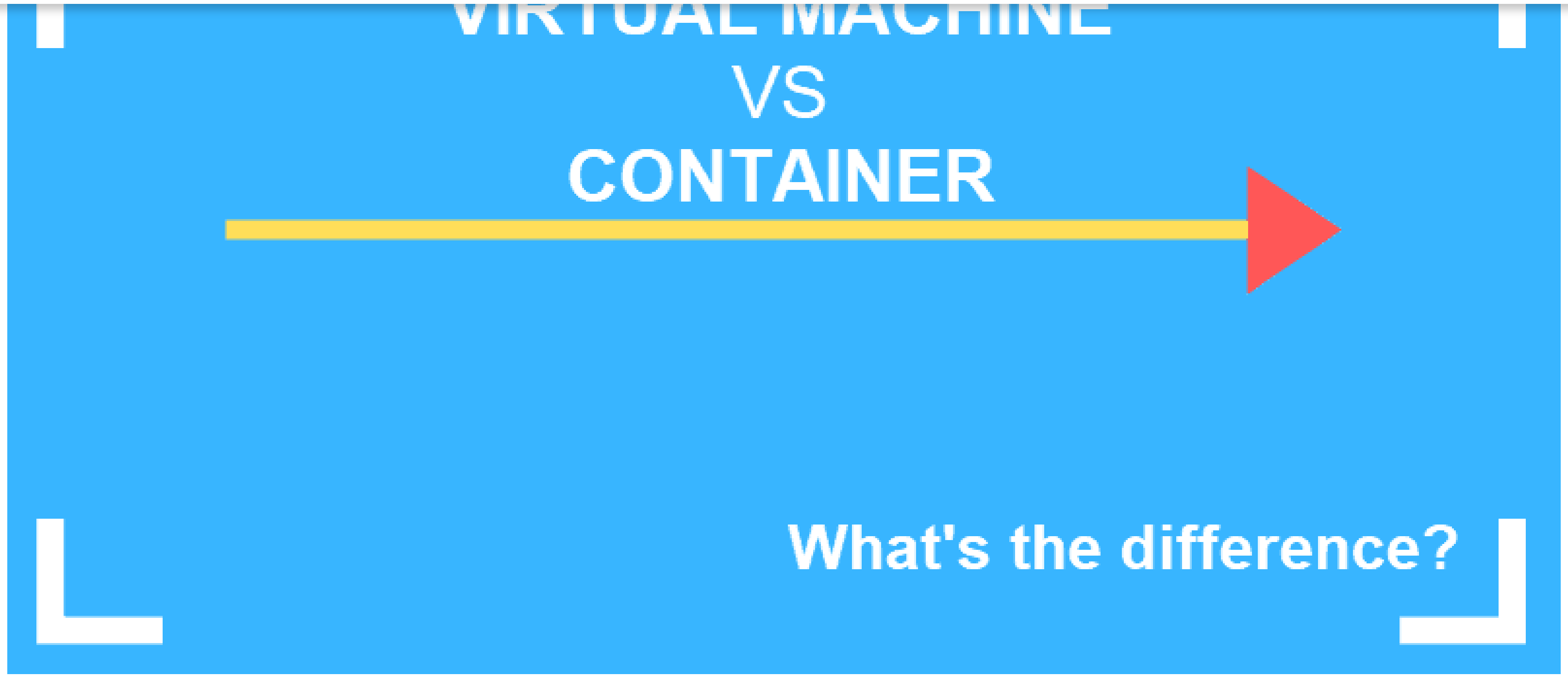
# Containers Vs Virtual Machines (VMs): What's The Difference?

Posted April 15, 2019 CONTAINERS VM

Both **virtual machines** and **containers** are used to created isolated virtual environments for developing and testing applications or software. The question is how they differ.

This article examines the two concepts to help understand the difference between a container and a VM. It defines containers and virtual machines separately, as well as side-by-side, to see what makes one distinct from the other.



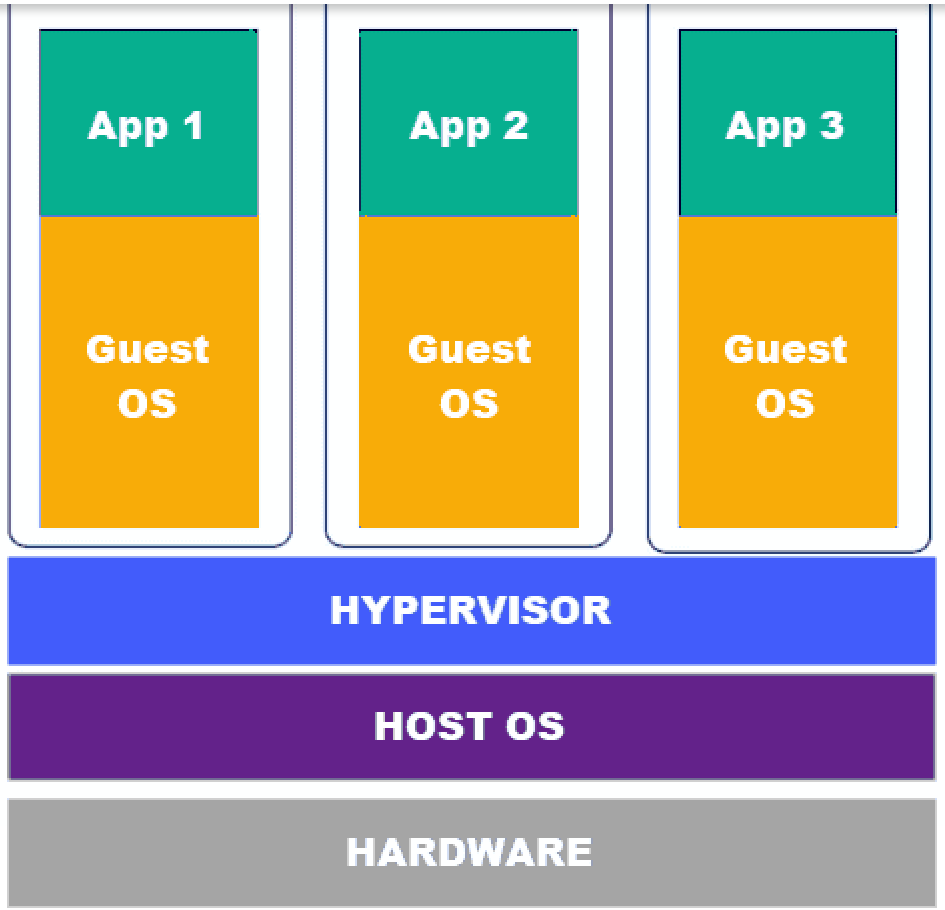
## What are Virtual Machines (VMs)?

A **virtual machine (VM)** is an operating system that shares the physical resources of one server. It is a guest on the host's hardware, which is why it is also called a **guest machine**.

There are several layers that make up a virtual machine. The layer that enables virtualization is the **hypervisor**. A hypervisor is a software that virtualizes the server.

## How a Virtual Machine Works

Everything necessary to run an app is contained within the virtual machine – the virtualized hardware, an OS, and any required binaries and libraries. Therefore, virtual machines have their own infrastructure and are self-contained.



Each VM is completely isolated from the host operating system. Also, it requires its own OS, which can be different from the host's OS. Each has its own binaries, libraries, and applications.

**Virtual machine monitor (VMM)**: another name for the hypervisor
**Host machine**: the hardware on which the VM is installed
**Guest machine**: another name for the VM

### Virtual Machine: PROS

servers.

Because there is one host machine, it allows you to **efficiently manage** all the virtual environments with the centralized power of the hypervisor. These systems are entirely separate from each other meaning you can install **multiple system environments**.

Most importantly, a virtual machine is isolated from the host OS and is a **safe** place for experimenting and developing applications.

## Virtual Machine: CONS

**Virtual machines may take up a lot of system resources** of the host machine, being many GBs in size. Running a single app on a virtual server means running a copy of an operating system as well as a virtual copy of all the hardware required for the system to run. This quickly adds up to a lot of RAM and CPU cycles.

The process of **relocating an app running on a virtual machine can also be complicated** as it is always attached to the operating system. Hence, you have to migrate the app as well as the OS with it. Also, when creating a virtual machine, the hypervisor allocates hardware resources dedicated to the VM.

A virtual machine rarely uses all the resources available which can **make the planning and distribution difficult.** That’s still economical compared to running separate actual computers.

## Popular VM providers:

- VMware vSphere
- VirtualBox
- Zen
- Hyper-V
- KVM

**Note:** Learn more about the beneficial role of virtualization in Devops.

## What is a Container?

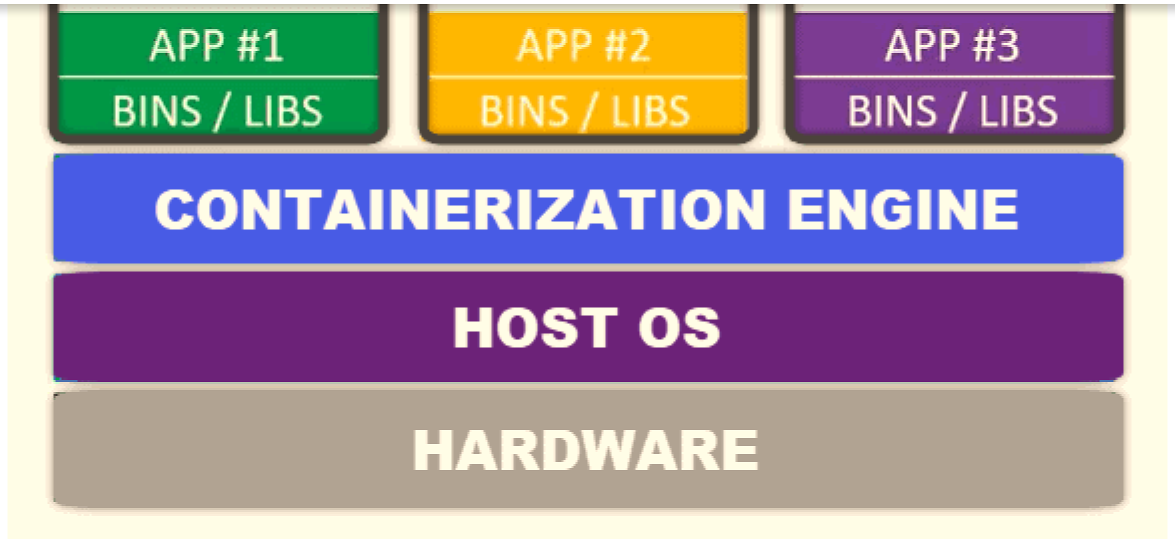
A container is an environment that runs an application that is not dependent on the operating system. It isolates the app from the host by virtualizing it. This allows users to created multiple workloads on a single OS instance.

The kernel of the host operating system serves the needs of running different functions of an app, separated into containers. Each container runs isolated tasks. It cannot harm the host machine nor come in conflict with other apps running in separate containers.

## How do Containers Work?

When working inside a container, you can create a template of an environment you need. The container essentially runs a snapshot of the system at a particular time, providing consistency in the behavior of an app.

The container shares the host’s kernel to run all the individual apps within the container. The only elements that each container requires are bins, libraries and other runtime components.



### Container: PROS

Containers can be as small as 10MB and you can easily limit their memory and CPU usage. This makes containers remarkably **lightweight** and **fast to launch** as opposed to deploying virtual machines, where the entire operating system needs to be deployed.
Because of their size, you can quickly **scale** in and out of containers and add identical containers.

Also, containers are excellent for **Continuous Integration and Continuous Deployment** (CI/CD) implementation. They foster collaborative development by distributing and merging images among developers.

### Container: CONS

A container uses the kernel of the host OS and has operating system dependencies. Therefore, containers can differ from the underlying OS by dependency, but not by type. The host's kernel **limits the use of other operating systems**.

Containers still do not offer the same **security and stability** that VMs can. Since they share the host's kernel, they cannot be as isolated as a virtual machine. Consequently, containers are process-level isolated, and one container can affect others by compromising the stability of the kernel.

**provisioning** on the host.

### Popular Container Providers:

- Docker
- AWS
- LXD
- Java Containers
- Hyper-V Containers
- Windows Server Containers

## How to Choose VMs vs Containers

Deciding whether to go for virtual machines or containers depends on the work you want your virtual environment to carry out.

### Virtual machines are a better solution if you need to:

1. Manage a variety of operating systems
2. Manage multiple apps on a single server
3. Run an app that requires all the resources and functionalities of an OS
4. Ensure full isolation and security

### Containers are suitable if you need to:

1. Maximize the number of apps running on a server
2. Deploy multiple instances of a single application

**Note:** VMs and containers should not necessarily be seen as rivals. Rather, you can use both to balance the workload between the two.

**Virtual machines are commonly used for** demanding applications, network infrastructure, and apps that will consume most of the resources of the VM.

**Containers are commonly used for** web, applications and caching services, network daemons, and small databases.

## Conclusion

Before committing to VMs or containers, make sure to consider all the factors that will influence efficiency. Take into account the kind of tasks they will have to perform as well as the resources you have available.

facebook twitter linkedin email

### Next you should also read

DevOps And Development, Virtualization

Virtualization, Bare Metal Servers

**What Is VDI and How Does It Work?**

Web Servers, Virtualization

**How to Set Up Apache Virtual Hosts on Ubuntu**

DevOps And Development

**How To Remove Docker Images, Containers,**

### Differences

October 31, 2019

Docker images and containers are elements in Docker's platform-as-a-service software. They are both essential...

READ MORE

This article explains what a virtual desktop environment is and how it can be implemented. It also details...

READ MORE

Name-based virtual hosts allow you to have a number of domains with the same IP address. This article...

READ MORE

Docker allows users to create a container in which an application or process can run. In this guide, you will...

READ MORE

## Author

### Sofija Simic

Sofija Simic is an aspiring Technical Writer at phoenixNAP. Alongside her educational background in teaching and writing, she has had a lifelong passion for information technology. She is committed to unscrambling confusing IT concepts and streamlining intricate software installations.

RECENT POSTS

Python Data Types {Comprehensive Overview}

How to Install Puppet on macOS

SYSADMIN
VIRTUALIZATION
DEVOPS AND DEVELOPMENT
SECURITY

BACKUP AND RECOVERY
BARE METAL SERVERS
WEB SERVERS
NETWORKING

ABOUT US
GITHUB
BLOG
RFP TEMPLATE
CAREERS

COLOCATION
SERVERS
CLOUD SERVICES
SOLUTIONS

EVENTS
PRESS
CONTACT US