Docker vs. Virtual Machines: Differences You Should Know

What's the difference between Docker and Virtual Machine? In this article, we'll compare the differences and provide our insights to help you decide between the two. Before we get started discussing Docker vs VM differences, let's first explain the basics.

What is Docker?

Organizations in today's world look forward to transforming their business digitally but are constrained by the diverse portfolio of applications, cloud, and on-premises-based infrastructure. Docker solves this obstacle of every organization with a container platform that brings traditional applications and microservices built on Windows, Linux, and mainframe into an automated and secure supply chain.

Docker is a software development tool and a virtualization technology that makes it easy to develop, deploy, and manage applications by using containers. A container refers to a lightweight, stand-alone, executable package of a piece of software that contains all the libraries, configuration files, dependencies, and other necessary parts to operate the application.

in other words, applications run the same irrespective of where they are and what machine they are running on because the container provides the environment throughout the software development life cycle of the application. Since containers are isolated, they provide security, thus allowing multiple containers to run simultaneously on the given host. Also, containers are lightweight because they do not require an extra load of a hypervisor. A hypervisor is a guest operating system like VMWare or VirtualBox, but instead, containers run directly within the host's machine kernel.

Containers provide the following benefits:

- Reduced IT management resources
- Reduced size of snapshots
- Quicker spinning up apps
- Reduced and simplified security updates
- Less code to transfer, migrate and upload workloads

What is a Virtual Machine?

A Virtual Machine (VM), on the other hand, is created to perform tasks that if otherwise performed directly on the host environment, may prove to be risky. VMs are isolated from the rest of the system; the software inside the virtual machine cannot tamper with the host computer. Therefore, implementing tasks such as accessing virus-infected data and testing of operating systems are done using virtual machines. We can define a virtual machine as:

A virtual machine is a computer file or software usually termed as a guest, or an image that is created within a computing environment called the host.

A virtual machine is capable of performing tasks such as running applications and programs like a separate computer making them ideal for testing other operating systems like beta releases, creating operating system backups, and running software and applications. A host can have several virtual machines running at a specific time. Logfile, NVRAM setting file, virtual disk file, and configuration file are some of the key files that make up a virtual machine. Another sector where VMs are of great use is server virtualization. In server virtualization, a physical server is divided into multiple isolated and unique servers, thereby allowing each server to run its operating system independently. Each virtual machine provides its virtual hardware, such as CPUs, memory, network interfaces, hard drives, and other devices.

VMs are broadly divided into two categories depending upon their use:

1. System Virtual Machines: A platform that allows multiple VMs, each running with its copy of the operating system to share the physical resources of the host system. Hypervisor, which is also a software layer, provides the virtualization technique. The hypervisor executes at the top of the operating system or the hardware alone.
2. Process Virtual Machine: Provides a platform-independent programming environment. The process virtual machine is designed to hide the information of the underlying

hardware and operating system and allows the program to execute in the same manner on every given platform.

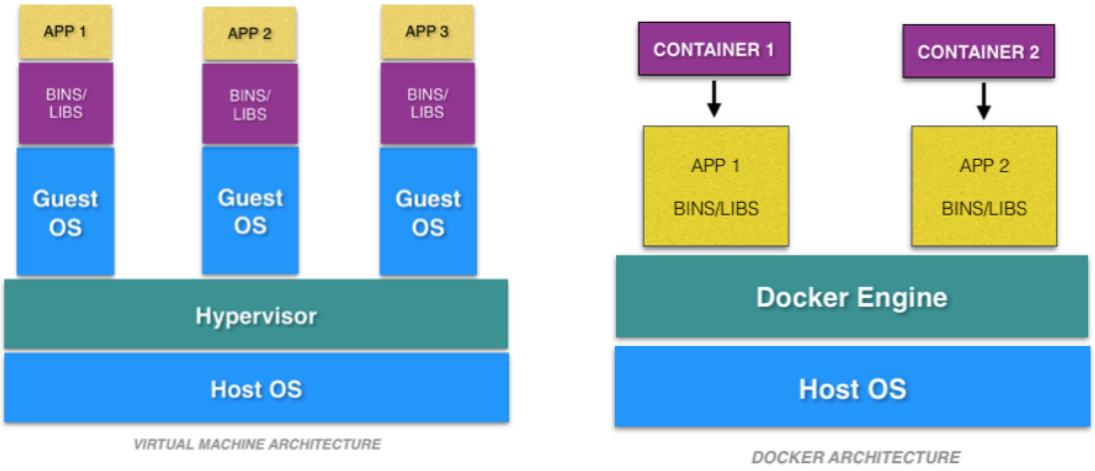
Although several VMs running at a time may sound efficient, it leads to unstable performance. As the guest OS would have its kernel, set of libraries and dependencies, this would take up a large chunk of system resources.

Other drawbacks include inefficient hypervisor and long boot uptime. The concept of Containerization overcomes these flaws. Docker is one such containerization platform.

Docker vs Virtual Machine: main differences

The following are the significant differences between Docker and virtual machines.

OS Support and Architecture



VMs have the host OS and guest OS inside each VM. A guest OS can be any OS, like Linux or Windows, irrespective of the host OS. In contrast, Docker containers host on a single physical server with a host OS, which shares among them. Sharing the host OS between containers makes them light and increases the boot time. Docker containers are considered suitable to run multiple applications over a single OS kernel; whereas, virtual machines are needed if the applications or services required to run on different OS.

Security

The second difference between VMs and Docker is that Virtual Machines are stand-alone with their kernel and security features. Therefore, applications needing more privileges and security run on virtual machines.

On the flip side, providing root access to applications and running them with administrative premises is not recommended in the case of Docker containers because containers share the host kernel. The container technology has access to the kernel subsystems; as a result, a single infected application is capable of hacking the entire host system.

Portability

Another relevant Docker vs Virtual Machine difference is about portability: VMs are isolated from their OS, and so they are not ported across multiple platforms without incurring compatibility issues. At the development level, if an application is to be tested on different platforms, then Docker containers must be considered.

Docker container packages are self-contained and can run applications in any environment, and since they don't need a guest OS, they can be easily ported across different platforms. Docker containers can be easily deployed in servers since containers being lightweight can be started and stopped in very less time compared to virtual machines.

Performance

The last main Docker vs VM difference refers to performance: Virtual Machines are more resource-intensive than Docker containers as the virtual machines need to load the entire OS to start. The lightweight architecture of Docker containers is less resource-intensive than virtual machines.

In the case of a virtual machine, resources like CPU, memory, and I/O may not be allocated permanently to containers — unlike in the case of a Docker container, where the resource usage works with the load or traffic.

Scaling up and duplicating a Docker container is simple and easy as compared to a virtual machine because there is no need to install an operating system in them.

Apart from the major differences between Docker and VMs, some other ones are summarized below:

	Docker	Virtual Machines (VMs)
Boot-Time	Boots in a few seconds.	It takes a few minutes for VMs to boot.
Runs on	Dockers make use of the execution engine.	VMs make use of the hypervisor.

Memory Efficiency	No space is needed to virtualize, hence less memory.	Requires the entire OS to be loaded before starting the surface, so less efficient.
Isolation	Prone to adversities as no provisions for isolation systems.	Interference possibility is minimum because of the efficient isolation mechanism.
Deployment	Deploying is easy as only a single image, containerized, can be used across all platforms.	Deployment is comparatively lengthy as separate instances are responsible for execution.
Usage	Docker has a complex usage mechanism consisting of both third party and docker managed tools.	Tools are easy to use and simpler to work with.

Should I choose Docker or Virtual Machine (VM)?

It won't be fair to compare Docker and virtual machines since they are intended for different use. Docker, no doubt, is gaining momentum these days, but they cannot be said to replace virtual machines. In spite of Docker gaining popularity, a virtual machine is a better choice in certain cases. Virtual machines are considered a suitable choice in a production environment, rather than Docker containers since they run on their own OS without being a threat to the host computer. But if the applications are to be tested then Docker is the choice to go for, as Docker provides different OS platforms for the thorough testing of the software or an application.

Furthermore, a Docker container uses docker-engine instead of a hypervisor, like in a virtual machine. As the host kernel is not shared, using docker-engine makes containers small, isolated, compatible, high performance-intensive, and quickly responsive. Docker containers have comparatively low overhead as they have compatibility to share single kernel and application libraries. Organizations are making use of the hybrid approach mostly as the choice between virtual machines and Docker containers depend upon the kind of workload offered.

Also, not many digital operational companies rely on virtual machines as their primary choice and prefer migrating towards using containers as the deployment is comparatively lengthy and running microservices is also one of the major challenges it possesses. However, they are still some firms that prefer virtual machines over Dockers whereas companies who are interested in enterprise-grade security for their infrastructure prefer to make use of Dockers.

Finally, containers and Docker are not in conflict with virtual machines, they are both complementary tools for different workloads and usage. Virtual machines are built for

applications that are usually static and don’t change very often. Whereas, the Docker platform is built with a mindset to be more flexible so that containers can be updated easily and frequently.

Will containers ever replace virtual machines? Comment your thoughts below or give further suggestions.