Ec2 Instance

An Amazon EC2 instance is **a virtual server in Amazon's Elastic Compute Cloud** (EC2) for running applications on the Amazon Web Services (AWS) infrastructure. AWS is a comprehensive, evolving cloud computing platform; EC2 is a service that enables business subscribers to run application programs in the computing environment. It can serve as a practically unlimited set of virtual machines (VMs).

Amazon provides various types of instances with different configurations of CPU, memory, storage and networking resources to suit user needs. Each type is available in various sizes to address specific workload requirements.

Features of Amazon EC2

Amazon EC2 provides the following features:

- Virtual computing environments, known as *instances*
- Preconfigured templates for your instances, known as *Amazon Machine Images (AMIs)*, that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as *instance types*
- Secure login information for your instances using *key pairs* (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop, hibernate, or terminate your instance, known as *instance store volumes*
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as *Amazon EBS volumes*
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as *Regions* and *Availability Zones*
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using *security groups*
- Static IPv4 addresses for dynamic cloud computing, known as *Elastic IP addresses*
- Metadata, known as *tags*, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS Cloud, and that you can optionally connect to your own network, known as *virtual private clouds* (VPCs)

Docker v/s Virtual Machine

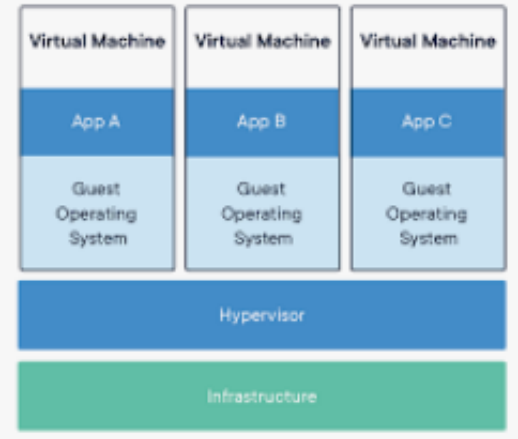
DOCKER: Docker is a software development tool and a virtualization technology that makes it easy to develop, deploy, and manage applications by using containers. A container refers to a lightweight, stand-alone, executable package of a piece of software that contains all the libraries, configuration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

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 can perform 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.

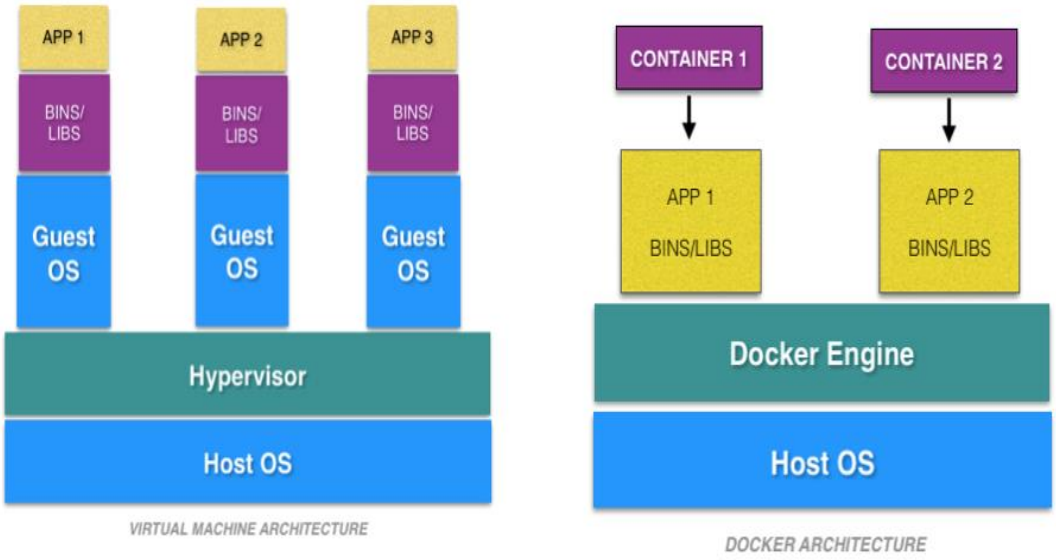


Docker vs Virtual Machine: main differences

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

OS Support and Architecture

The main difference between Docker and VMs lies in their architecture, demonstrated below.



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 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.