DOCKER

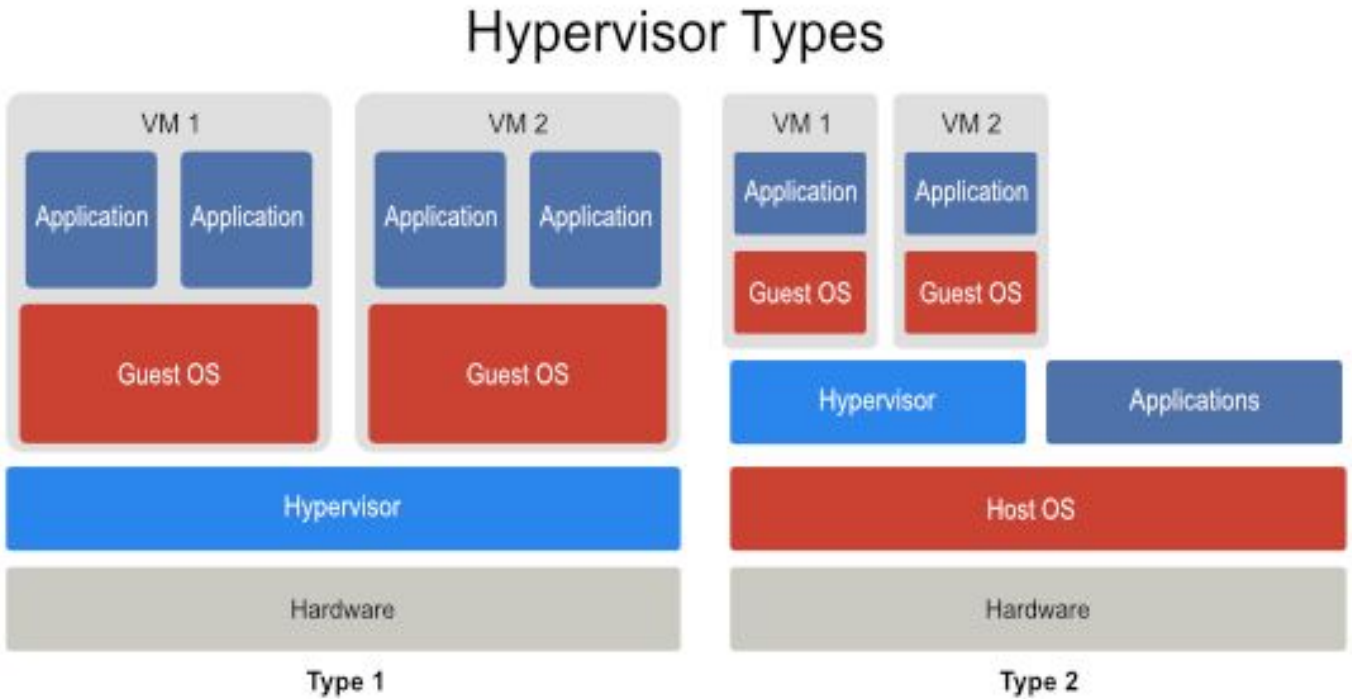
VIRTUALIZATION

Virtualization is the process of running a virtual instance of a computer system in a layer abstracted from the actual hardware. Most commonly, it refers to running multiple operating systems on a computer system simultaneously. To the applications running on top of the virtualized machine, it can appear as if they are on their own dedicated machine, where the operating system, libraries, and other programs are unique to the guest virtualized system and unconnected to the host operating system which sits below it.

What is a hypervisor?

A hypervisor is a program for creating and running virtual machines. Hypervisors have traditionally been split into two classes: type one, or "bare metal" hypervisors that run guest virtual machines directly on a system's hardware, essentially behaving as an operating system. Type two, or "hosted" hypervisors behave more like traditional applications that can be started and stopped like a normal program. In modern systems, this split is less prevalent, particularly with systems like KVM. KVM, short for kernel-based virtual machine, is a part of the Linux kernel that can run virtual machines directly, although you can still use a system running KVM virtual machines as a normal computer itself.

TYPES OF HYPERVISOR



What is a virtual machine?

A virtual machine is the emulated equivalent of a computer system that runs on top of another system. Virtual machines may have access to any number of resources: computing power, through hardware-assisted but limited access to the host machine's CPU and memory; one or more physical or virtual disk devices for storage; a virtual or real network interface; as well as any devices such as video cards, USB devices, or other hardware that are shared with the virtual machine. If the virtual machine is stored on a virtual disk, this is often referred to as a disk image. A disk image may contain the files for a virtual machine to boot, or, it can contain any other specific storage needs.

WHAT IS DOCKER ?

Docker is an open-source software designed to facilitate and simplify application development. It is a set of platform-as-a-service products that create isolated virtualized environments for building, deploying, and testing applications.

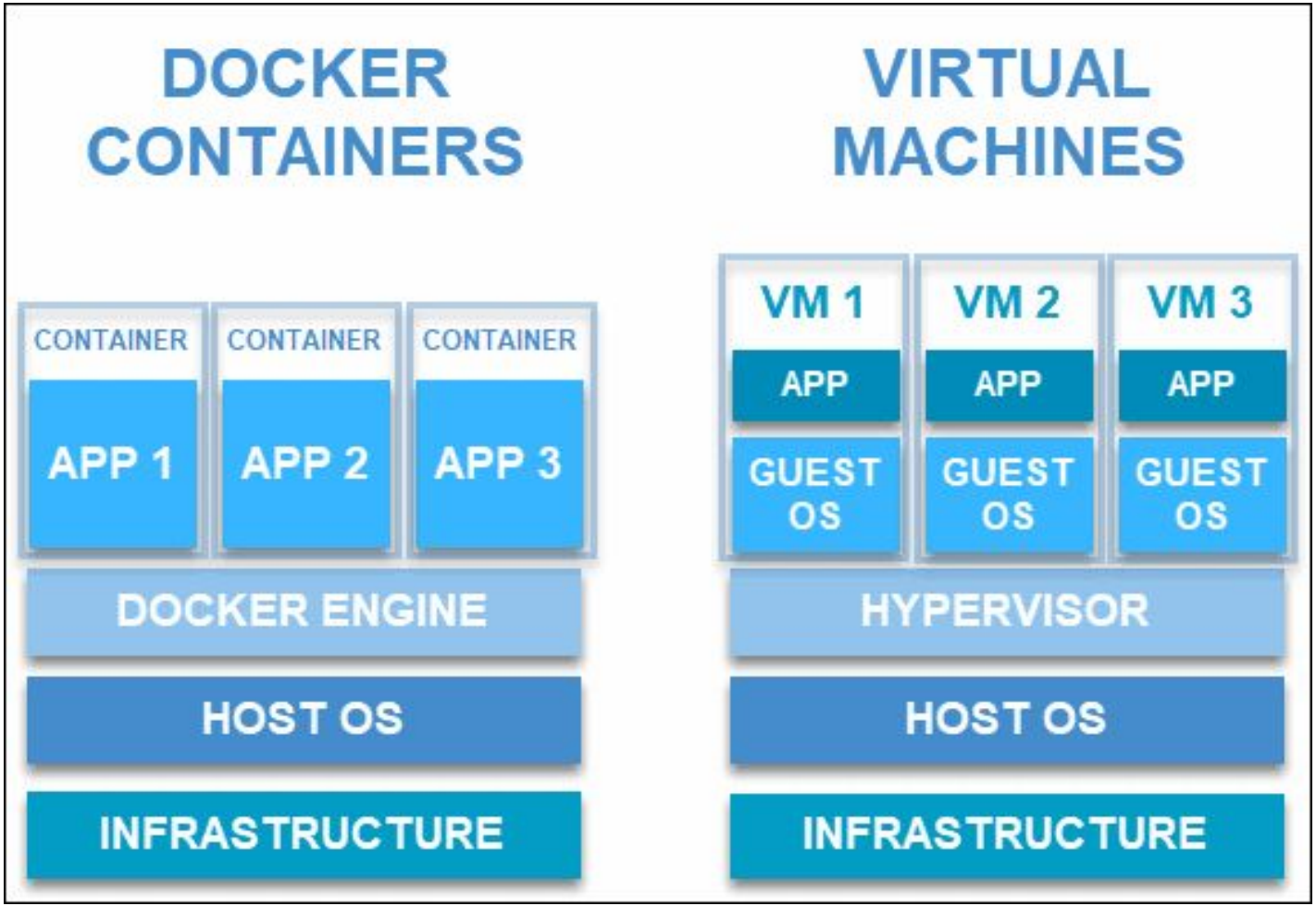
Although the software is relatively simple to master, there are some Docker-specific terms that new users may find confusing. Dockerfiles, images, containers, volumes, and other terminology will need to be mastered and should become second nature over time.

What are containers?

For a long time, companies have been using container technologies to address the weak points of virtual machines. We can think of containers as **more lightweight versions of VMs**. The important difference between containers and VMs is that containers don’t need their own operating system. All containers on a host share that host’s operating system, which frees up a lot of system resources.

IN OTHER WORDS

A Docker container is a virtualized run-time environment where users can isolate applications from the underlying system. These containers are compact, portable units in which you can start up an application quickly and easily.


DOCKER CONTAINERS
CONTAINER
CONTAINER
CONTAINER
APP 1
APP 2
APP 3
DOCKER ENGINE
HOST OS
INFRASTRUCTURE
VIRTUAL MACHINES
VM 1
VM 2
VM 3
APP
APP
APP
GUEST OS
GUEST OS
GUEST OS
HYPERVISOR
HOST OS
INFRASTRUCTURE

What is Dockerfile ?

Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

Dockerfile keywords

- **ADD** copies the files from a source on the host into the container's own filesystem at the set destination.
- **CMD** can be used for executing a specific command within the container.
- **ENTRYPOINT** sets a default application to be used every time a container is created with the image.
- **ENV** sets environment variables.
- **EXPOSE** associates a specific port to enable networking between the container and the outside world.
- **FROM** defines the base image used to start the build process.
- **MAINTAINER** defines a full name and email address of the image creator.
- **RUN** is the central executing directive for Dockerfiles.
- **USER** sets the UID (or username) which is to run the container.
- **VOLUME** is used to enable access from the container to a directory on the host machine.
- **WORKDIR** sets the path where the command, defined with CMD, is to be executed.
- **LABEL** allows you to add a label to your docker image.

What is a Docker Swarm?

A Docker Swarm is a group of either physical or virtual machines that are running the Docker application and that have been configured to join together in a cluster. Once a group of machines have been clustered together, you can still run the Docker commands that you're used to, but they will now be carried out by the machines in your cluster. The activities of the cluster are controlled by a swarm manager, and machines that have joined the cluster are referred to as nodes.

What is Docker Compose?

To understand Docker Compose, let’s look at Myntra as an example. Myntra is a fashion e-commerce website similar to Amazon. You visit the Myntra website through your web browser and go through several activities, like logging in to your account, browsing a catalog, checking out, and so on. Behind each of these activities or services are different products, such as an account database, product database, cart database, and others that run behind the scenes.

THE END