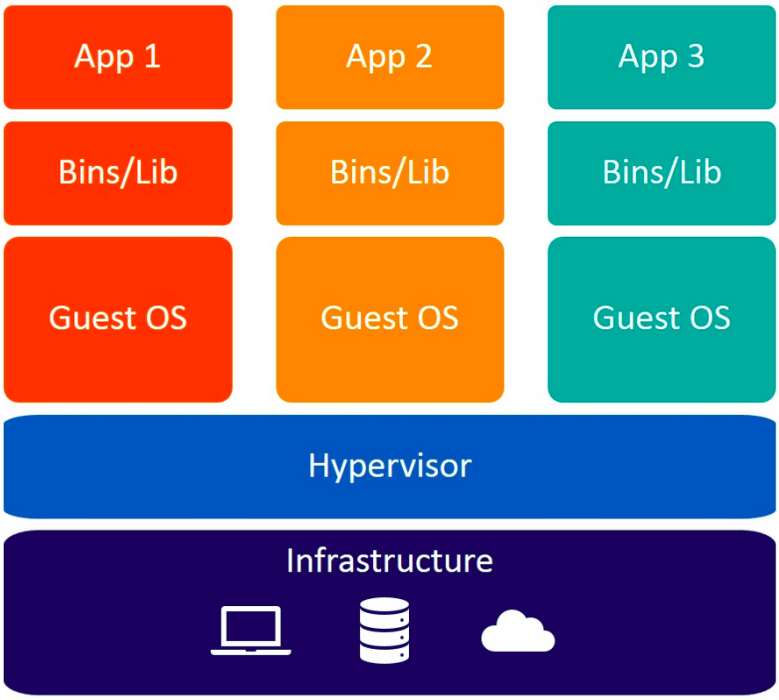
Working Mechanism of Virtual Machine

A **Virtual Machine (VM)** is a virtual environment in which we run a system above another system. It has its own separate storage, memory allocation, network interfaces and operating system (OS). Virtual Machine thus allows the user to run multiple separate systems within a single system.



Virtual Machine works through Virtualization technology. **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.

Virtualization uses software to simulate virtual hardware that allows multiple VMs to run on a single machine. The physical machine is known as the host while the VMs running on it are called guests.

This process is managed by software known as a **Hypervisor**. The hypervisor is responsible for managing and provisioning resources, like memory and storage; from the host to guests. It also schedules operations in VMs so they don't overrun each other when using resources. VMs only work if there is a hypervisor to virtualize and distribute host resources.

Hypervisors are of two types:

a) *Type 1 hypervisors:*

   They are bare metal hypervisors. They are installed natively on the underlying physical hardware.

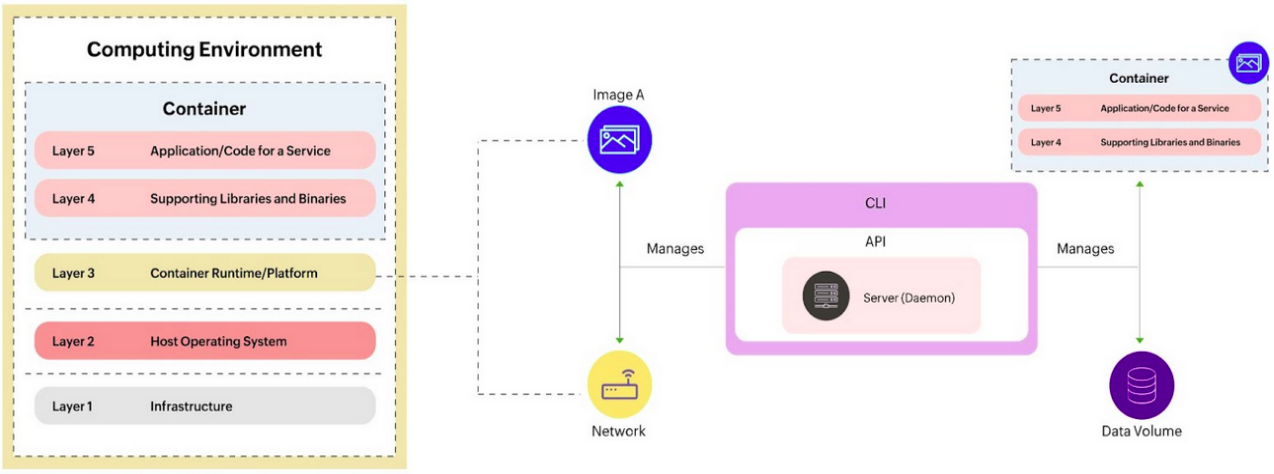
b) *Type 2 hypervisors:*

   They are hosted hypervisors.They run on the host computer's operating system.

In this way, the **Virtual Machine** does work.

Working Mechanism of Containers

Containers are a consistent, efficient, portable, and secure virtualization solution that focuses on virtualizing the operating system level rather than the hardware level. A container is actually just an isolated process that shares the same Linux kernel as the host operating system .



Above figure shows the working of a container at a glance. A container is compiled from a base image, and the sample application or microservice is packaged into a container image and deployed for use through the container platform. The container platform is a client-server software facilitating the execution of the container by providing three key operational components: a daemon service, an API, and a CLI interface. Once deployed, the container remains active as long as the application or microservice needs to perform its role in the overall application, and it shuts down once the delivery is complete. The container can be further activated as needed.

In this way, **Containers** do work.

Problems that Virtualization solves and its drawbacks in context to modern application Deployments

The problems that the Virtualization solves are :

a) Enhances IT processes across several fronts

Virtualization improves several aspects of IT and data management. This type of environment does wonders to recovering data, involving automatic snapshots of the virtual machines to ensure data is updated and easily recovered. Said machines can also be transferred quickly should the disaster happen to the data center itself. Thus, nearly everything is quicker and more convenient, leading to less issues upon execution.

b) Provides an environment for better testing

This set-up creates separate user environments for the purpose of effective testing. With independent user environments in place, you can do testing without the repercussions of deleting the important data of the whole enterprise.

c) Diminishes hardware and costs

Hardware can be quite costly per piece, and this is especially apparent when you have so many assets in your network. By employing virtualization, you segment virtual machines from one server – allowing you to multitask in managing several systems.

d) Direct Communication with Physical Infrastructure

With Virtualization, the VM's can directly communicate with the Physical Infrastructure with/without the need of Host Os which is not in the case of Containers where the Containers first must communicate with the Host Os to interact with the Physical Infrastructure

The drawbacks of Virtualization in context to modern application deployments are :

a) It can have a high cost of implementation

The cost for the average individual or business when virtualization is being considered will be quite low. For the providers of a virtualization environment, however, the implementation costs can be quite high.

b) It creates an availability issue

The primary concern that many have with virtualization is what will happen to their work should their assets not be available. If an organization cannot connect to their data for an extended period of time, they will struggle to compete in their industry. And, since availability is controlled by third-party providers, the ability to stay connected is not in one's control with virtualization.

c) Portability Issues

Many more portability issues occur because of Virtualization as virtual machines are gigabytes-sized software which is very large compared to certain megabytes-sized containers.

d) It takes time

Although you save time during the implementation phases of virtualization, it costs users time over the long-run when compared to local systems. That is because there are extra steps that must be followed to generate the desired result.

Problems Containers solves in regard to app deployment

a) Traditionally, software applications were deployed directly onto the host machine. This had the disadvantage of entangling the application's executables, configuration, libraries, and life cycles with each other and with the host OS. But this new approach to package each application and its dependencies into a self-sufficient isolated container, makes the application completely decoupled from the host OS and infrastructure. It allows for the deployment of an application on any host, without first worrying if the host has all the dependencies installed.

b) Aside from decoupling an application from the host, an even more important advantage of using containers is the isolation of resources. If multiple containers are running on a host, a fatal error inside one container can't affect other containers, or the host.

c) Compared to this, when all the applications are directly installed on the host, a fatal error can bring down, or corrupt, other applications, and in some cases could crash the entire host. But there is no scope for such a scenario if the applications are deployed in containers. Containerization of applications adds great value, and will simplify a great deal of complexity involved in developing, deploying, and maintaining distributed applications.

d) It wasn't long before various engineering teams across the industry realized the advantages with containerization of applications, and decided that every application that gets deployed within their organization must be containerized.