**Assignment-2**

By- Utkarsh Mishra (4ni19is110 “B-section”)

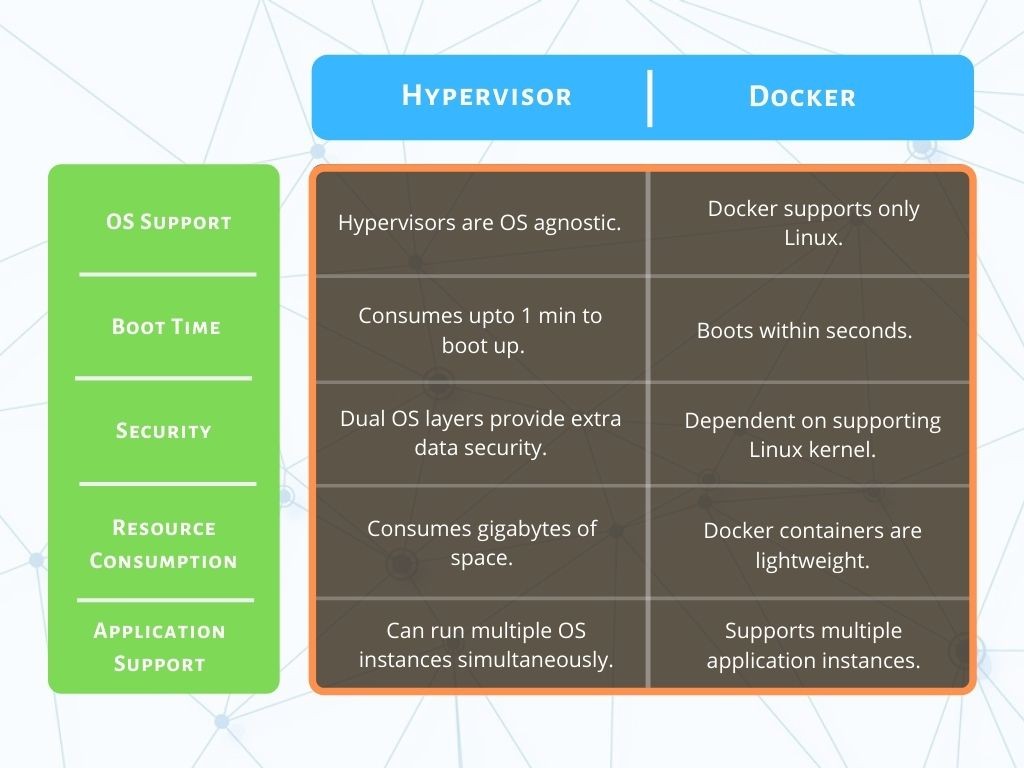
#### Compare Hypervision and Docker.

A hypervisor is more like the operating system itself. It has top level privileges and runs in Ring 0 of protected memory. It's aware of the host OS, and all the hardware the host is controling. It's able to take the hardware and virtualize it into separate spaces for a guest OS including CPU cores, memory, storage, human interface devices, and even GPU and sound devices.

The host shares all or this with the hypervisor and in turn the guest OS'es.

Docker operates within Linux with elevated permissions and makes use of the Linux kernel's CGroups, and Namespace features to share resources with containers that are like guest operating systems, but are running in isolated sections of the kernel that makes hardware requests through Linux, but under it's supervision. The kernel is handling all of the hardware for Docker. The Linux guests under Docker run from a read only image with a union ﬁlesystem where all the changes and work are stored.

Containers are allocated resources in much smaller amounts via software virtualization requests whereas hypervisors have access to hardware virtualization.



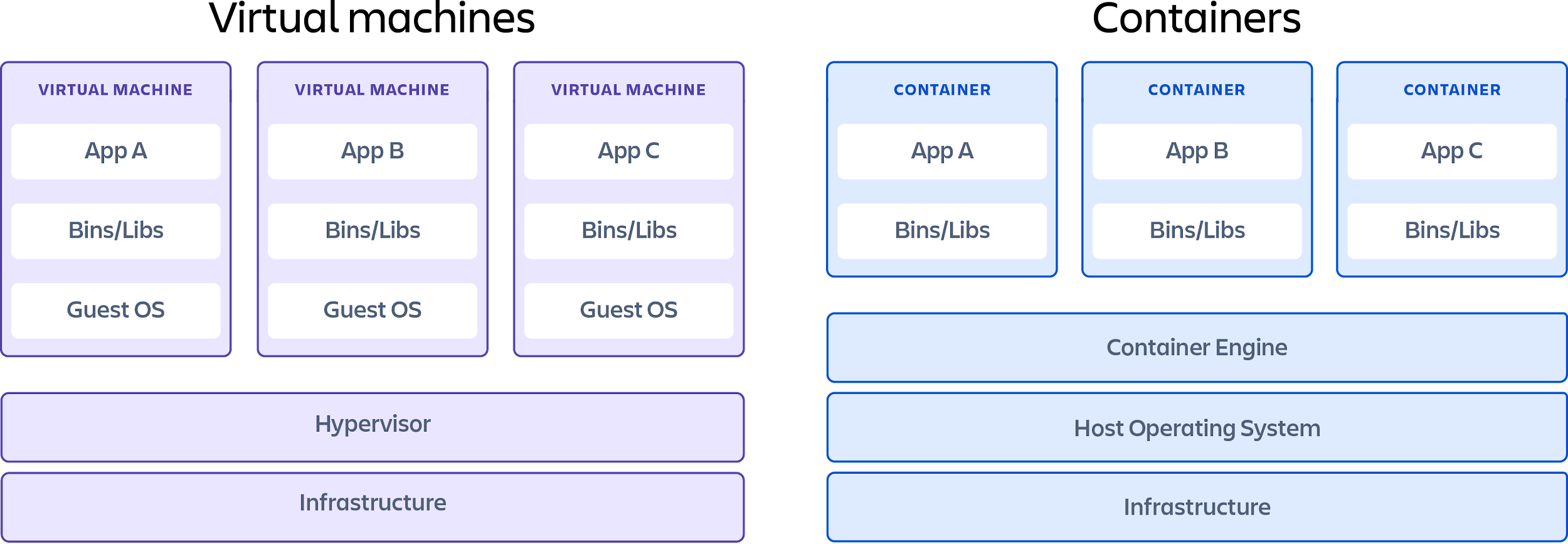
#### Comparison between Containers and Virtual Machines.

Containers and virtual machines are very similar resource virtualization technologies.

Virtualization is the process in which a system singular resource like RAM, CPU, Disk, or Networking can be ‘virtualized’ and represented as multiple resources. The key

differentiator between containers and virtual machines is that virtual machines

virtualize an entire machine down to the hardware layers and containers only virtualize software layers above the operating system level.



## What is a container?

Containers are lightweight software packages that contain all the dependencies required to execute the contained software application. These dependencies include things like system libraries, external third-party code packages, and other operating system level applications. The dependencies included in a container exist in stack levels that are higher than the operating system.

# Pros

* Iteration speed

Because containers are lightweight and only include high level software, they are very fast to modify and iterate on.

* Robust ecosystem

Most container runtime systems offer a hosted public repository of pre-made containers. These container repositories contain many popular software applications like databases or messaging systems and can be instantly downloaded and executed, saving time for development teams

# Cons

* Shared host exploits

Containers all share the same underlying hardware system below the operating system layer, it is possible that an exploit in one container could break out of the container and affect the shared hardware. Most popular container runtimes have public repositories of pre-built containers. There is a security risk in using one of these public images as they may contain exploits or may be vulnerable to being hijacked by nefarious actors.

## Popular container providers

* Docker

Docker is the most popular and widely used container runtime. Docker Hub is a giant public repository of popular containerized software applications. Containers on

Docker Hub can instantly downloaded and deployed to a local Docker runtime.

* RKT

Pronounced "Rocket", RKT is a security-ﬁrst focused container system. RKT containers do not allow insecure container functionality unless the user explicitly enables insecure features. RKT containers aim to address the underlying cross

contamination exploitive security issues that other container runtime systems suffer from.

* Linux Containers (LXC)

The Linux Containers project is an open-source Linux container runtime system. LXC is used to isolate operating, system-level processes from each other. Docker actually uses LXC behind the scenes. Linux Containers aim to offer a vender neutral

open-source container runtime.

* CRI-O

CRI-O is an implementation of the Kubernetes Container Runtime Interface (CRI) that allows the use of Open Container Initiative (OCI) compatible runtimes. It is a lightweight alternative to using Docker as the runtime for Kubernetes.

## What is a virtual machine?

Virtual machines are heavy software packages that provide complete emulation of low level hardware devices like CPU, Disk and Networking devices. Virtual machines may also include a complementary software stack to run on the emulated hardware. These hardware and software packages combined produce a fully functional snapshot of a computational system.

### Pros

* Full isolation security

Virtual machines run in isolation as a fully standalone system. This means that virtual machines are immune to any exploits or interference from other virtual

machines on a shared host. An individual virtual machine can still be hijacked by an exploit but the exploited virtual machine will be isolated and unable to contaminate any other neighboring virtual machines.

* Interactive development

Containers are usually static deﬁnitions of the expected dependencies and

conﬁguration needed to run the container. Virtual machines are more dynamic and can be interactively developed. Once the basic hardware deﬁnition is speciﬁed for a virtual machine the virtual machine can then be treated as a bare bones computer.

Software can manually be installed to the virtual machine and the virtual machine can be snapshotted to capture the current conﬁguration state. The virtual machine snapshots can be used to restore the virtual machine to that point in time or spin up additional virtual machines with that conﬁguration.

### Cons

* Iteration speed

Virtual machines are time consuming to build and regenerate because they

encompass a full stack system. Any modiﬁcations to a virtual machine snapshot can take signiﬁcant time to regenerate and validate they behave as expected.

* Storage size cost

Virtual machines can take up a lot of storage space. They can quickly grow to several gigabytes in size. This can lead to disk space shortage issues on the virtual machines host machine.

### Popular virtual machine providers

* Virtualbox

Virtualbox is a free and open source x86 architecture emulation system owned by Oracle. Virtualbox is one of the most popular and established virtual machine

platforms with an ecosystem of supplementary tools to help develop and distribute virtual machine images.

* VMware

VMware is a publicly traded company that has built its business on one of the ﬁrst

x86 hardware virtualization technologies. VMware comes included with a hypervisor which is a utility that will deploy and manage multiple virtual machines. VMware has robust UI for managing virtual machines. VMware is a great enterprise virtual machine option offering support.

* QEMU

QEUM is the most robust hardware emulation virtual machine option. It has support for any generic hardware architecture. QEMU is a command line only utility and does not offer a graphical user interface for conﬁguration or execution. This trade-off makes QEMU one of the fastest virtual machine options.