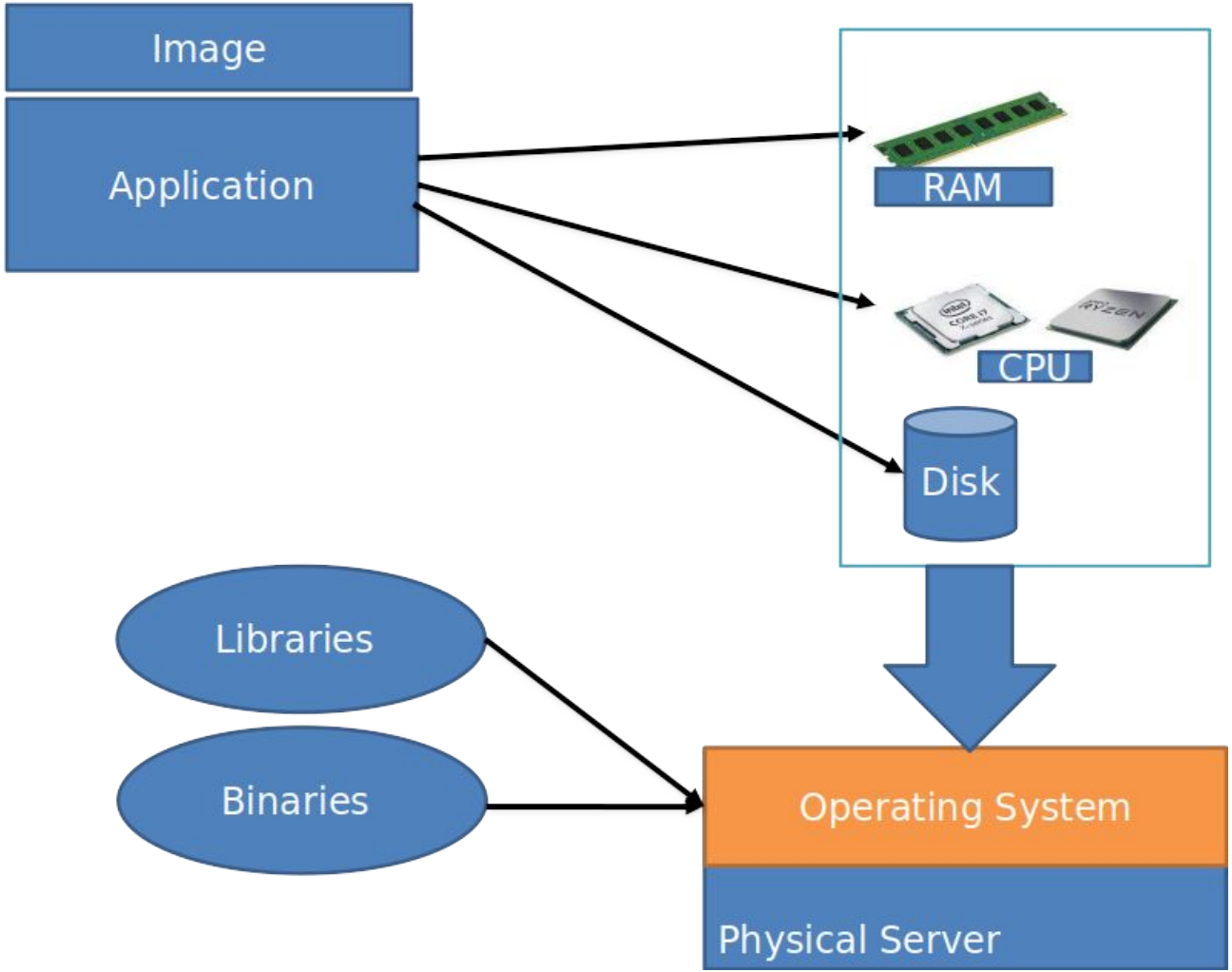
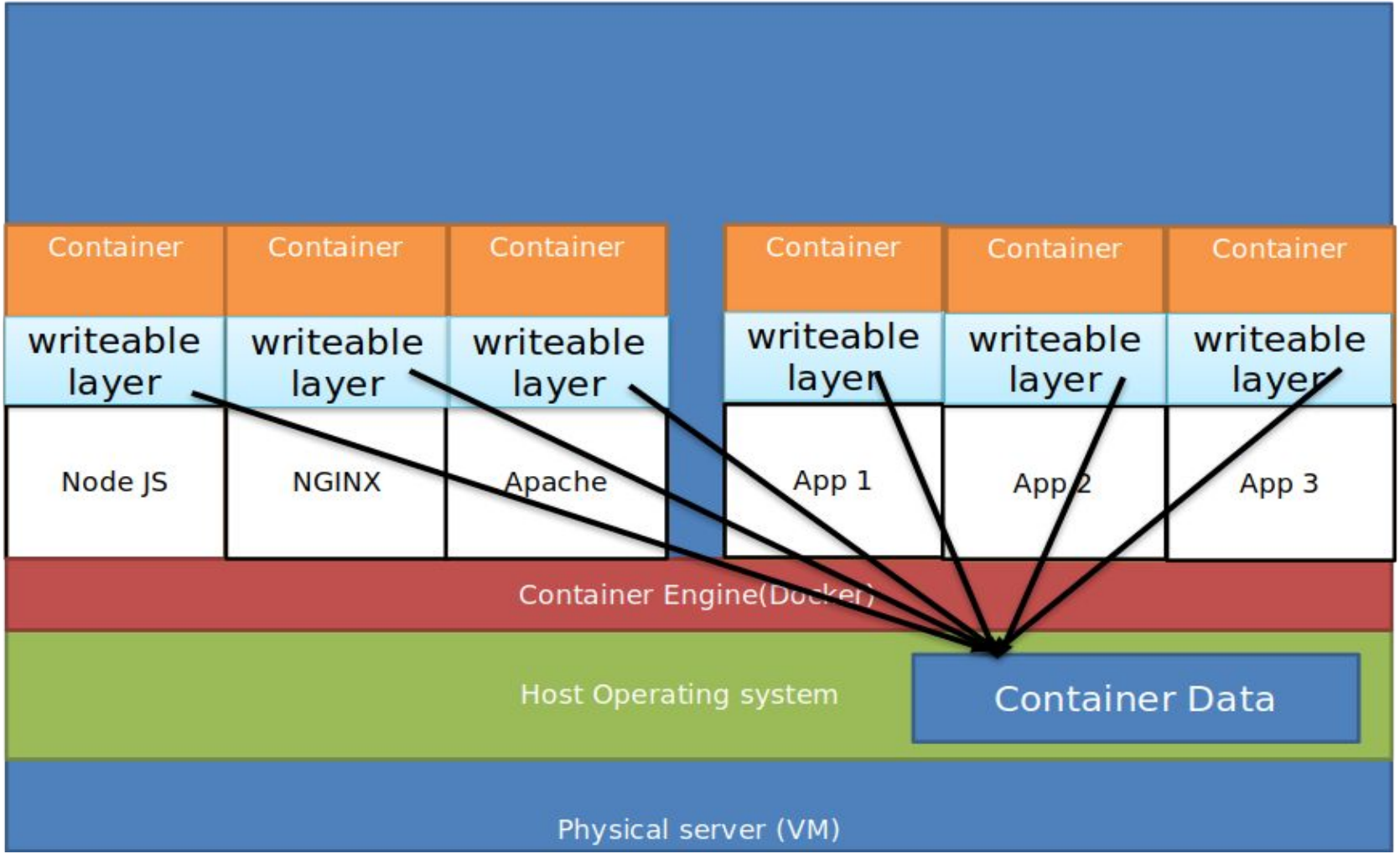
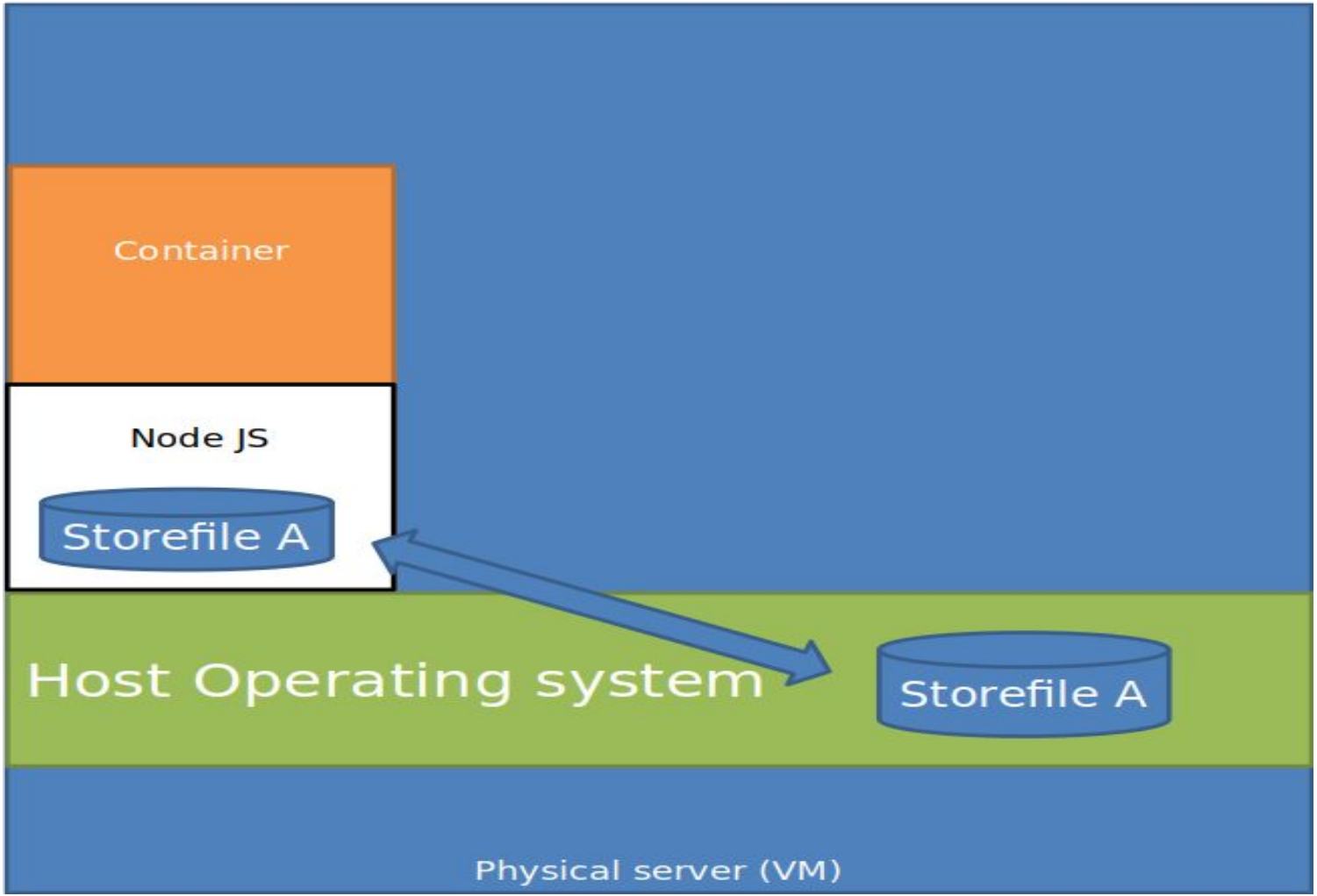


Image
Application
RAM
CPU
Disk
Libraries
Binaries
Operating System
Physical Server


Container
Container
Container
Container
Container
Container
writeable layer
writeable layer
writeable layer
writeable layer
writeable layer
writeable layer
Node JS
NGINX
Apache
App 1
App 2
App 3
Container Engine(Docker)
Host Operating system
Container Data
Physical server (VM)


Container
Node JS
Storefile A
Host Operating system
Storefile A
Physical server (VM)

Type of store Files

- Volumes
- Bind Mounts
- tmpfs

Volumes

- Data is **stored in the Docker Host filesystem**, which is **managed by Docker**. It means, Docker isolate the core functionality of the Host machine.
- A new directory will be created within Docker storage directory on the host machine.
- Path in Linux (**/var/lib/docker/volumes/**).
- Non-Docker processes should not modify this part of the filesystem.
- Docker engine create a Volume at the time of **Container creation / Service Creation**.
- Volume can be **shared between multiple containers** simultaneously.
- Volumes also support the use of *volume drivers*, which allow us to store the data on cloud or remote host.

Type of Volumes

Named : At the time of container creation, we specify the name of Volume.

Anonymous: Volume name is not mentioned at the time of creation of container.

Docker gives a random unique name.

Both behave in the same ways.

- How to create ?

  Command : **“docker volume create”**

- docker volume create --mount or -v

docker run -d --name Vol-mount-container1 **--mount** source=my-voloume1,target=/app nginx:latest

- For named Volume => Vol-mount-container1 is the volume name.
  For anonymous volume -> this field is blank.
  Command Option (--mount)
- It consist of multiple key value belgium, <key>=<value>; **type=volume**
- It includes all options in one field, separated by commas (,). source= my-voloume1 or src= my-volume1
- destination (dst, or target)= This key has the value of path where the file or directory is mounted in the container. target=/app Or dst=/app
- volume-opt, this key can be specified more than once.
  volume-opt=**type=nfs**,
  volume-opt=device=**<nfs-server>:<nfs-path>**

Command option (-v or –volume)

docker run -d --name Vol-v-container1 -v my-volume1:/app nginx:latest

- It is more explicit and verbose mode.
- It has 3 options, separated by colon (:) .
- All these 3 fields should be in order.

name:path:permission

1. field is name, in named volume, name of volume should be unique in the host machine. In anonymous this field is blank.
2. field is path, where the file or directory will be mounted in the container.
3. field is permission like, ro => Read Only.

command Options

--mount

It includes all options in one field.

It consists of multiple key-value pairs, separated by commas (,).

Order of keys isn’t mandatory. Keys values are understandable.

-v or –volume

It is more explicit and verbose mode.

It has 3 options, separated by colon (:)

All these 3 fields should be in order.

name:path:permission.

LAb for volume

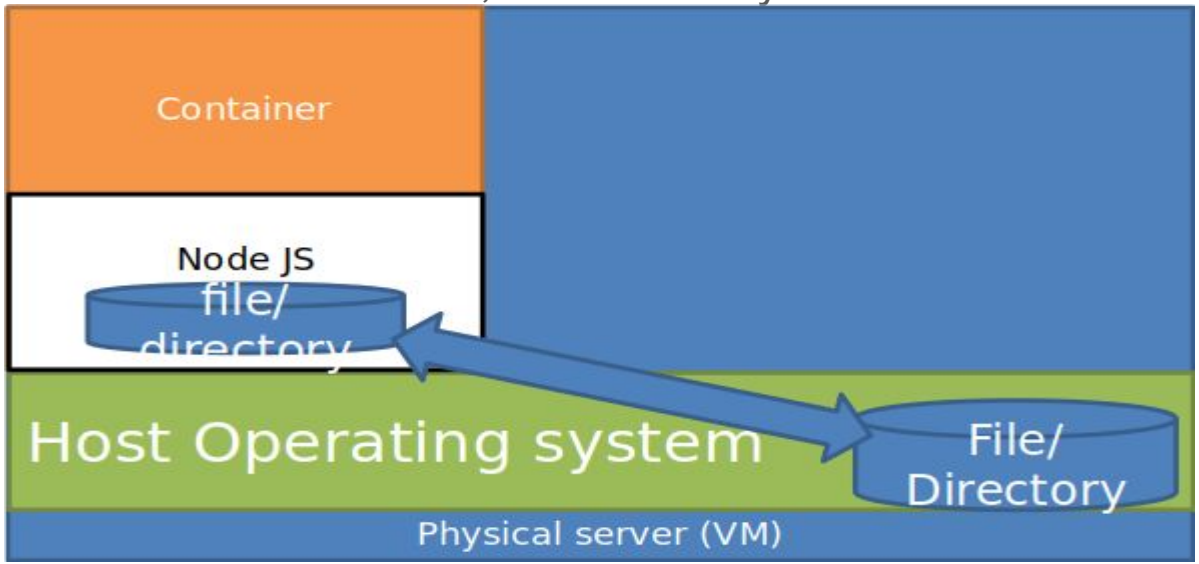
https://github.com/cfitechops/Docker/blob/main/07-docker.md

2. Bind Mounts

- It is stored in anywhere on Docker Host filesystem.
- Path : ***Anywhere***.
- Non-Docker processes can modify this part of the filesystem.
- It has limited functionality as compared to Volume

Bind Mounts… Continue

- In Bind Mounts, a file or directory on the Docker host machine is mounted in the container.
- This file or directory is referenced by its full path on the host machine.
- It will create file on demand, if not already created.



- Performance is really imaging, but it depends on host machine file system. Because it isn’t managed by Docker.

- Thus, Docker CLI commands to directly manage bind mounts is not possible

LAb for Bind Mounts

https://github.com/cfitechops/Docker/blob/main/08-docker.md

3. tmpfs

- This option is only available, if you are using Docker on Linux OS.
- tmpfs mounts can not be shared between multiple containers, like volume and bind mount.
- In tmpfs, we can create files outside the container's writeable layers.
- In this, we are saving the data into Host memory not in docker host machine disk, like Volume and Bind mount.
- It means that when container stop, tmpfs mount removed and data erase from the memory.

Use case

- ***Sharing configuration files from the host machine to containers***.
- Like dns config through /etc/resolv.conf file?

LAb for tmpfs

https://github.com/cfitechops/Docker/blob/main/09-docker.md