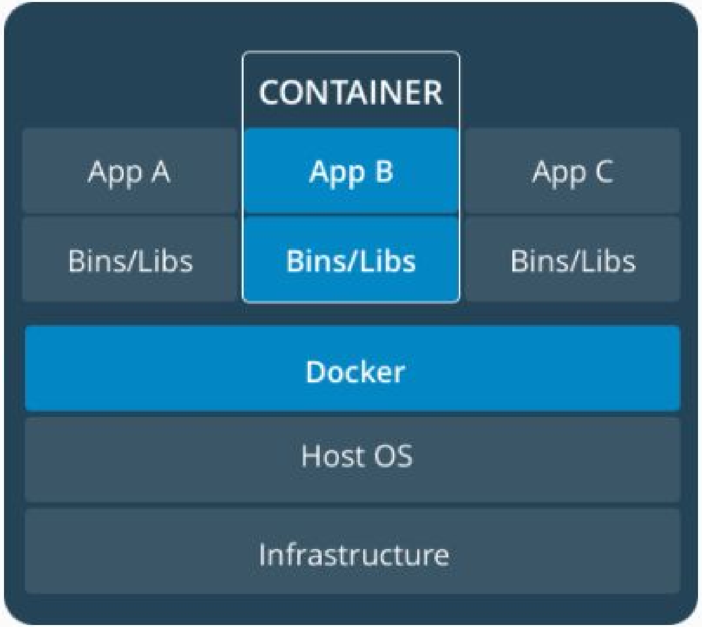
Containers and Docker

Sinziana Gafitanu

What are containers?

- A lightweight alternative to VMs
- Stand-alone, executable package of a piece of software that includes everything needed to run it:
  - Code
  - Runtime
  - System tools
  - System libraries
  - Settings
- Regardless if it’s Linux, Windows or Mac it will always rung the same
- Containers create isolation that can help deploy production and development easily
- The isolation helps manage dependencies and conflicts
- Standardization effort: OCI - https://www.opencontainers.org/

Container vs VMs



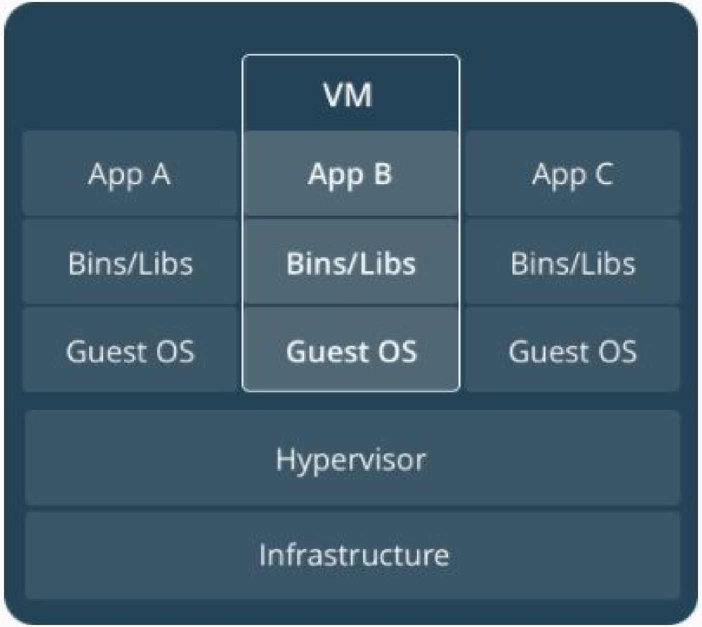
CONTAINER

App A	App B	App C
Bins/Libs	Bins/Libs	Bins/Libs

Docker

Host OS

Infrastructure



VM

App A	App B	App C
Bins/Libs	Bins/Libs	Bins/Libs
Guest OS	Guest OS	Guest OS

Hypervisor

Infrastructure

Credit: https://www.docker.com/what-container

Why use containers?

Pros:

- Almost all operations on docker containers happen in a matter of a few seconds
- They scale really well (1000 1GB VMs will need 1000 GB of space to run, 1000 1GB containers will need 1 GB + variable sized storage that the apps use to store each containers isolated data).
- Deploys are easier and faster
- Running tests (even with large databases) can be parallelized.

Cons:

- Full isolation is not guaranteed (but there are best practices to increase security)
- Resources are not guaranteed (that can be easily mitigated with auto-scaling cloud solutions)

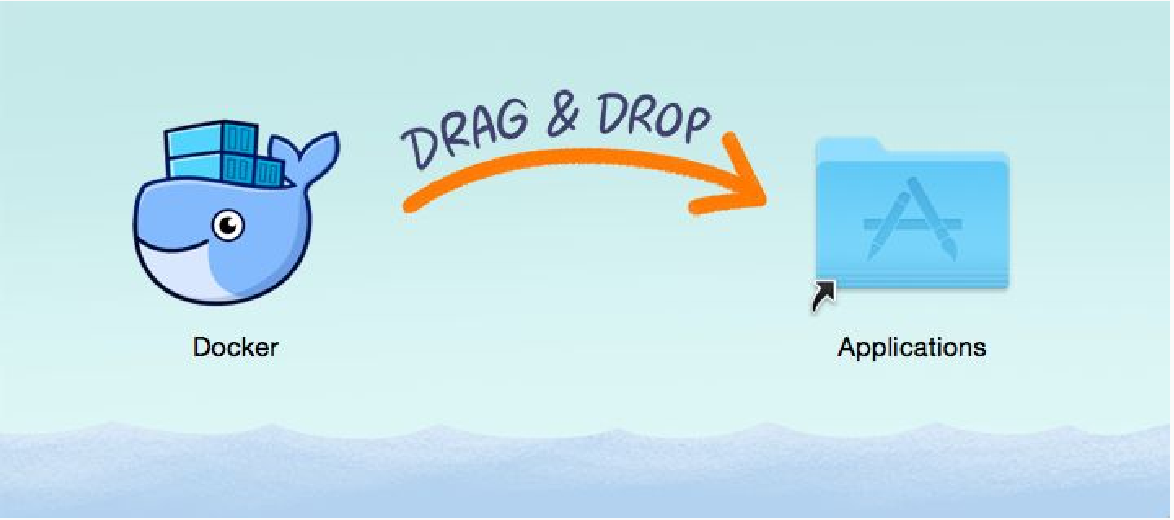
Docker

- The most popular
- Based on libcontainer/runC
- Uses a layered filesystem (AuFS)

- It accesses five Linux namespaces:
  - Process
  - Network
  - Mount
  - Hostname
  - Shared Memory

Install Docker

https://docs.docker.com/engine/getstarted/step_one/



Demo