Implementações com containers

Luís Felipe Borsoi
Luiz Felipe Flores

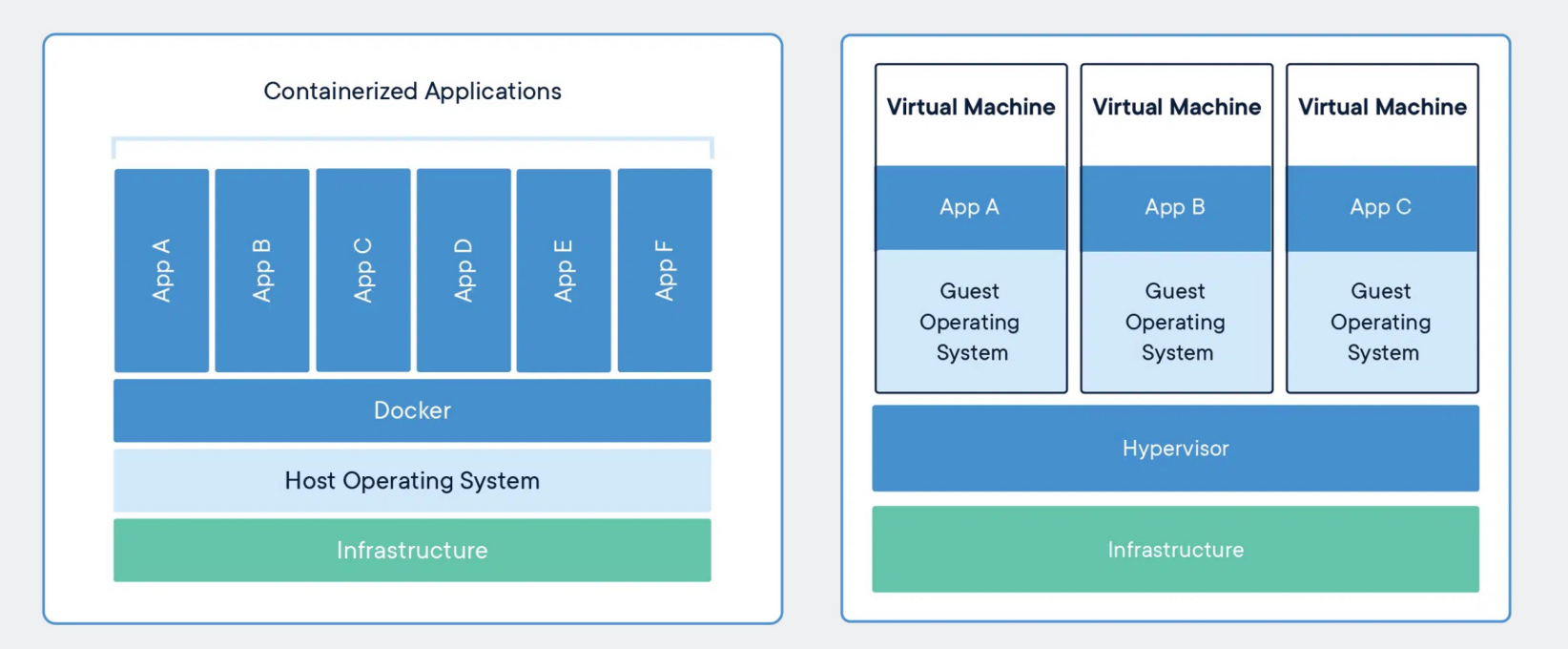
O que é um container?

- Um processo linux isolado;
- Unidade de software que empacota um código, todas as dependências e configurações, a fim de facilitar a operação/execução em diferentes ambientes;
- Instância de uma container image;
- Facilitam o desenvolvimento de aplicações na nuvem (microserviços);

Container != Máquina Virtual

- Containers virtualizam o Sistema operacional ao invés de hardware;
- Todos os containers compartilham recursos do mesmo sistema operacional da máquina, rodando como processos isolados no espaço do usuário;
- Imagens de containers são muito menores que imagens de máquinas virtuais;
- Containers são mais performáticos;
- Pode rodar vários containers no mesmo ambiente;
- Containers foram criados pensando na portabilidade;
- Containers fazem boot muito mais rápido;

Container != Máquina Virtual



Conceitos importantes

- Containers;
- Container images;
- Container engines;
- Container host;
- Registry;
- Image Layers;

Container images

- É um arquivo “executável” baixado de um registry que contém tudo que é necessário para rodar a aplicação;
- Elemento estático que origina containers quando instanciado;
- Pode possuir uma ou mais layers, assim como metadata;
- Possui um formato específico: hoje o padrão é o OCI (Open Container Initiative);
- Criado a partir de camadas em cima de um imagem base;

Container Engines

- Componente que executa comandos do usuário e roda as container images;

Container Engines

- Na prática, um container engine não roda os containers diretamente, mas utiliza um container runtime (ex.: runC etc);
- Responsabilidades principais:

    - Lidar com inputs do usuário;
    - Lidar com chamadas de API de um orquestrador de containers (e.g. Kubernetes);
    - Baixar imagens do registry;
    - Preparar o container para a execução;

Container Host

- Sistema operacional que os containers estão rodando;
- No Linux, é possível rodar os containers diretamente no sistema operacional da máquina;
- No Windows, normalmente utiliza-se o WSL2;
- No MacOS, normalmente é rodado uma máquina virtual Linux por debaixo dos panos;
- Em aplicações rodando na nuvem, a máquina do provedor cloud (AWS, GCP, Azure) está rodando um Linux;

Registry

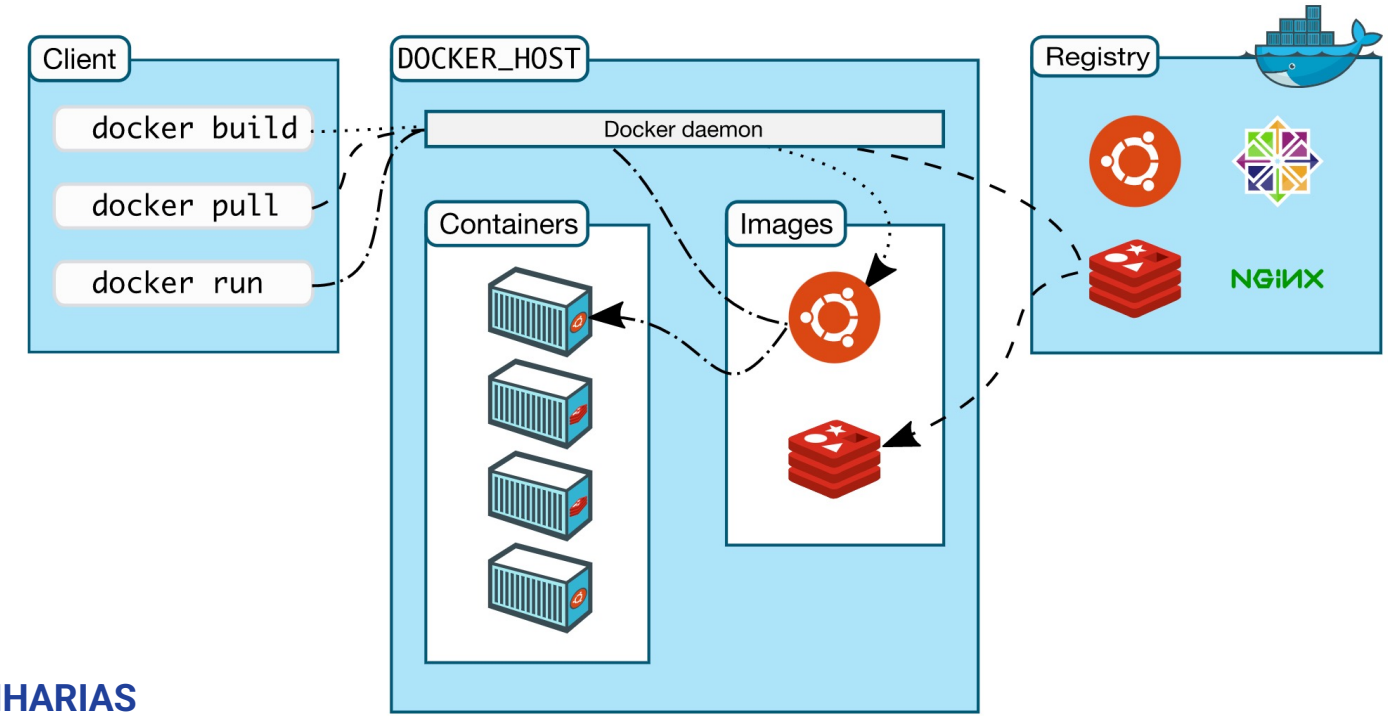
- Basicamente é um sistema de arquivos que armazena e distribui container images;
- Pode ser público ou privado;

Image Layers

- Um container image pode ser composto por várias camadas;
- Cada camada é uma imagem por si só;
- Cada camada armazena a diff entre si própria e a camada anterior;
- Utilidade: evitar transferir pacotes redundantes em um build/pull de uma imagem;
- É possível converter múltiplas camadas em uma só (squashing);
- No caso do Docker: cada comando em um Dockerfile gera uma camada diferente para a imagem;
- A camada final terá uma tag para que possa ser distribuída pelo registry.

Docker

- Plataforma open-source para desenvolver, distribuir e rodar aplicações em containers;



Docker

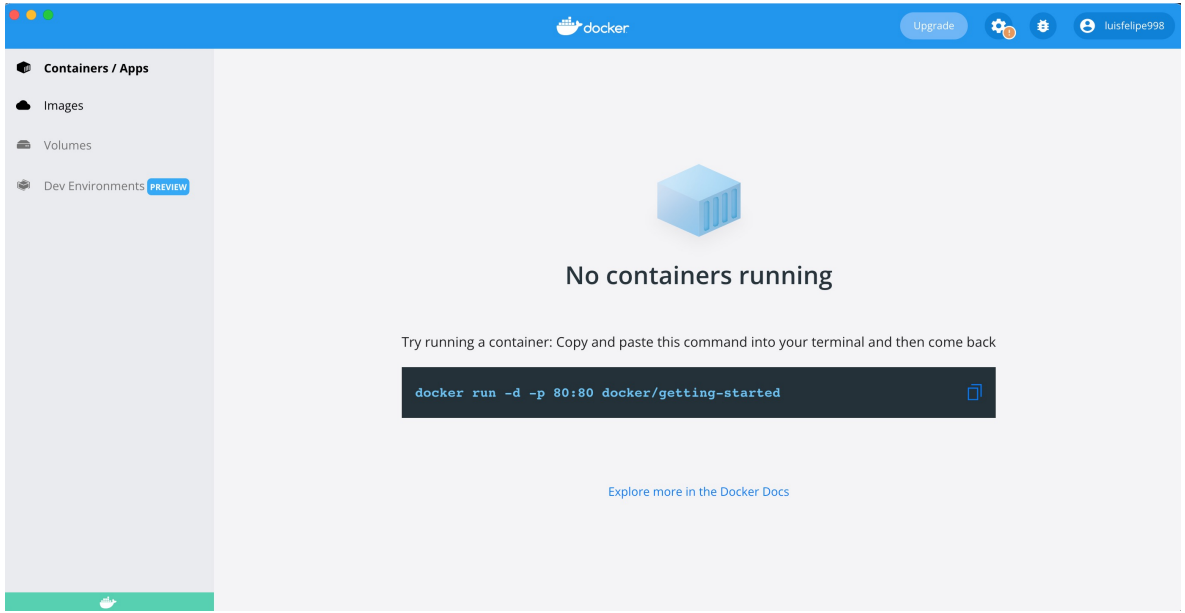

IT WORKS ON MY MACHINE
THEN WE'LL SHIP YOUR MACHINE
AND THAT IS HOW DOCKER WAS BORN

Elementos do Docker

- Docker Desktop;
- CLI;
- Docker Daemon;
- Container, Container image, Container host e registry;
- Dockerfile
- Docker-compose
- Network e Data Volumes

Docker Desktop

- Aplicação com UI para criar, gerenciar e distribuir imagens docker;
- Inclui ferramentas de desenvolvimento, Kubernetes e sincronização com Docker engines produtivas.



Docker CLI

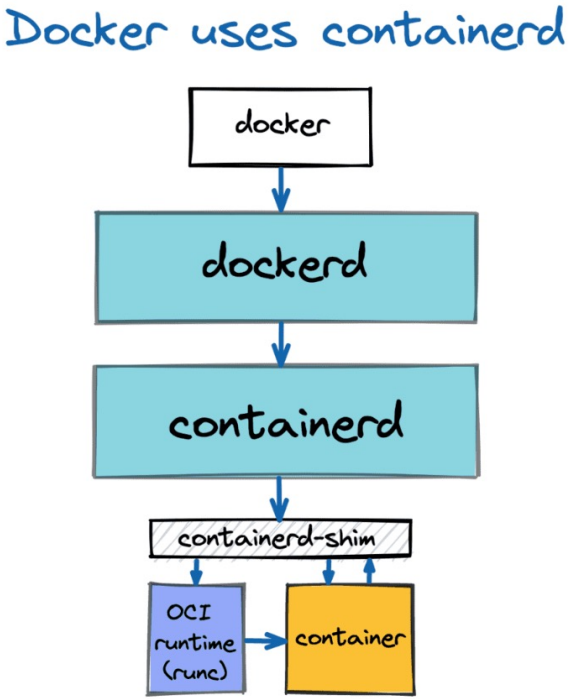
- Ferramenta para rodar comandos docker pelo terminal (linha de comando);
- Envia os comandos via chamadas REST (protocolo Unix) para o servidor (docker daemon);

```
Last login: Mon May 30 19:19:43 on ttys002
➜  ~ docker --help

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default
                           "/Users/I521862/.docker")
  -c, --context string     Name of the context to use to connect to the
                           daemon (overrides DOCKER_HOST env var and
                           default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level
                           ("debug"|"info"|"warn"|"error"|"fatal")
                           (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default
                           "/Users/I521862/.docker/ca.pem")
      --tlscert string     Path to TLS certificate file (default
                           "/Users/I521862/.docker/cert.pem")
      --tlskey string      Path to TLS key file (default
                           "/Users/I521862/.docker/key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx (Docker Inc., v0.8.1)
  compose*    Docker Compose (Docker Inc., v2.3.3)
  config      Manage Docker configs
  container   Manage containers
  context     Manage contexts
  image       Manage images
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
  node        Manage Swarm nodes
  plugin      Manage plugins
  scan*       Docker Scan (Docker Inc., v0.17.0)
  secret      Manage Docker secrets
  service     Manage services
  stack       Manage Docker stacks
  swarm       Manage Swarm
  system      Manage Docker
  trust       Manage trust on Docker images
  volume      Manage volumes
```

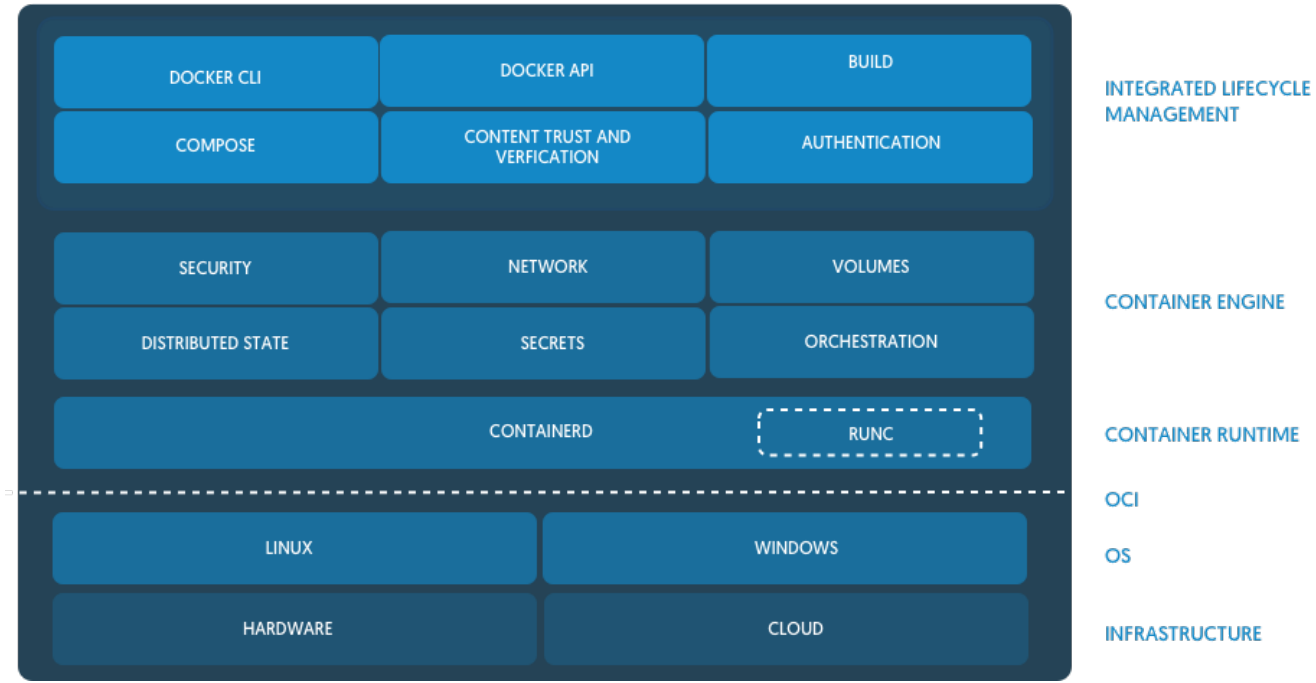
```
Commands:
  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  images      List images
  import      Import the contents from a tarball to create a filesystem image
  info        Display system-wide information
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  login       Log in to a Docker registry
  logout      Log out from a Docker registry
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  run         Run a command in a new container
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  search      Search the Docker Hub for images
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  version     Show the Docker version information
  wait        Block until one or more containers stop, then print their exit codes

Run 'docker COMMAND --help' for more information on a command.
```

Docker Daemon

- Processo rodando em background;
- Recebe comandos via chamadas REST e abstrai chamadas de mais baixo nível;
- Se comunica com o containerd (container engine): outro daemon que gerencia as imagens, containers, rede, volumes e tudo que é necessário no lifecycle do container;
- Containerd abstrai chamadas de sistema (syscalls) que variam em cada sistema operacional;

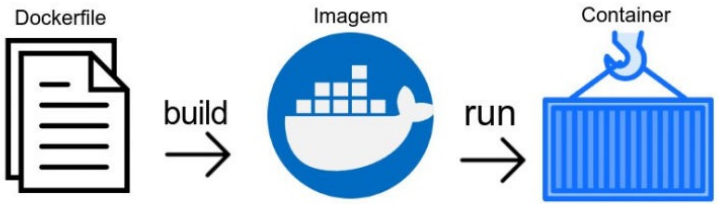


Docker Daemon



Dockerfile

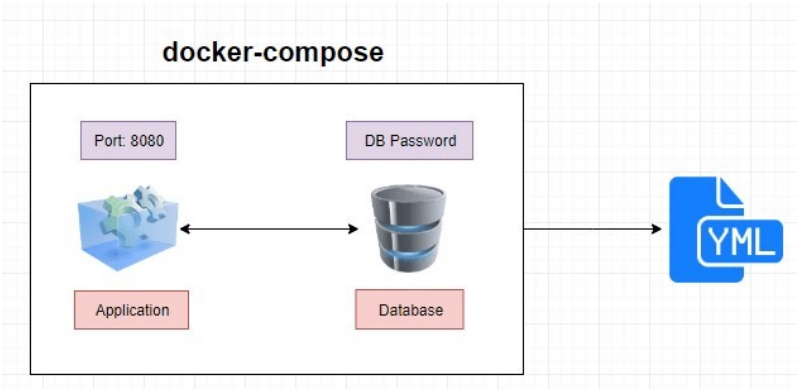
- Arquivo de texto com instruções de como fazer o build de uma docker image;

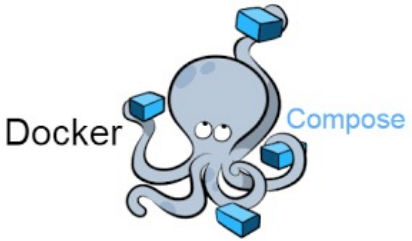



```
FROM ubuntu

MAINTAINER ARSTECH arstech@e-mail

RUN apt-get update && apt-get upgrade -y
RUN apt-get install -y apt-utils htop

CMD ["echo","It's my Docker Image "]
```

Docker-compose

- Utilitário que facilita a criação e uso de múltiplos containers como um serviço único (e.g. um container para a aplicação e outro container para o banco de dados);
- Roda os containers de forma isolada, mas permite a comunicação entre eles;
- Descrito por um arquivo docker-compose.yml

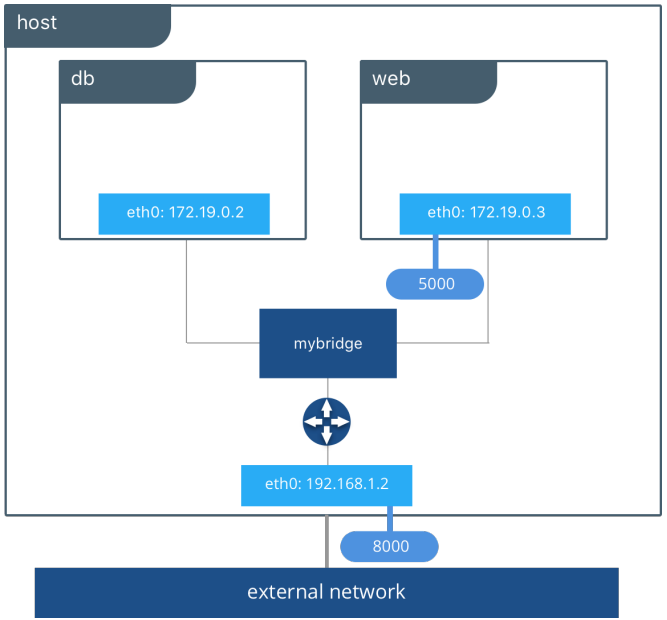




Networks

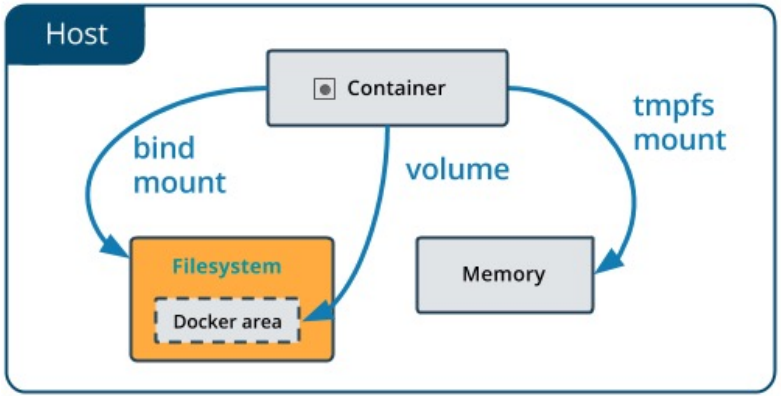
- Containers são isolados por padrão;
- É necessário expor as portas que se quer comunicar;
- Tipos de rede: bridge e overlay;
- Docker daemon gerencia as redes criadas e a distribuição dos IPs para cada container (servidor DHCP);

```
→  ~ docker network ls
NETWORK ID     NAME                                DRIVER    SCOPE
a920f576d955   bridge                              bridge    local
e0a0fd8a2db8   host                                host      local
```



Data Volumes

- A memória utilizada na execução de um container é transiente;
- Para persistir dados permanentemente, é necessário criar um volume;
- Docker daemon gerencia os volumes criados;
- Containers diferentes podem acessar um mesmo volume;

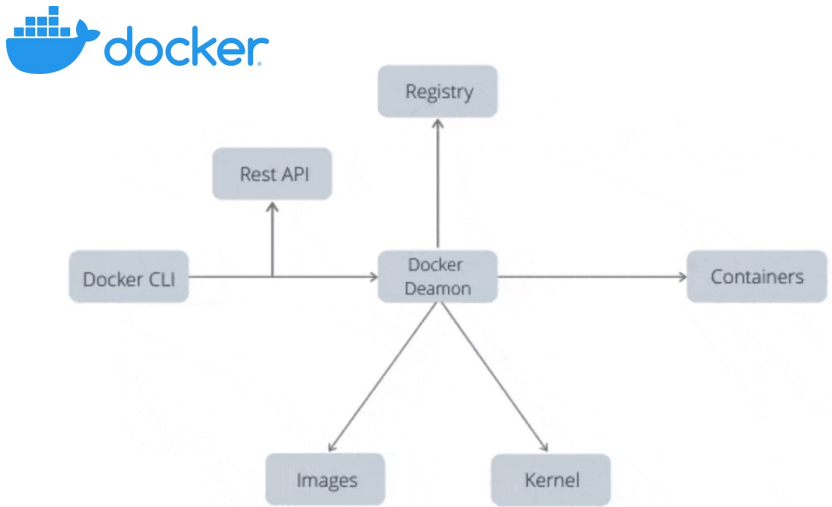


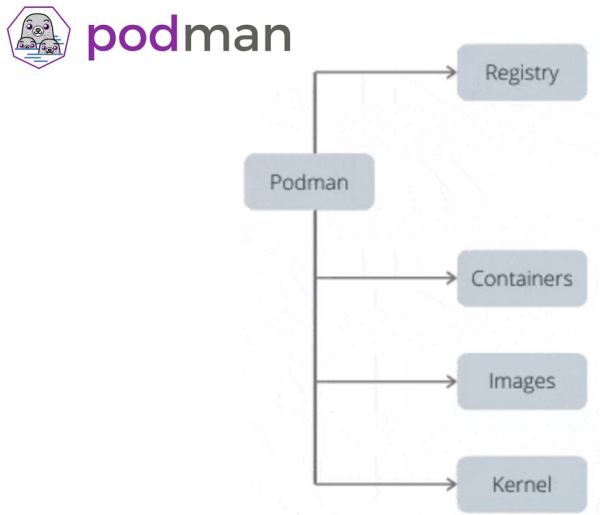
Docker vs Podman

- Daemon é um ponto de falha: os containers podem ficar em processos orfãos;
- Por padrão, o docker daemon precisa de acesso root pra executar;
- Suporte pra Linux e Windows (WSL2);

- Não utiliza um daemon: o processo do podman interage diretamente com a criação de containers;
- Rootless (mais seguro);
- Compatível com imagens criadas pelo docker (OCI);
- Suporte só pra Linux;

Docker vs Podman





Orquestradores de Containers

- Automatizam muito do esforço operacional requerido para rodar containers e serviços:
    - Provisionamento
    - Implantação
    - Escalabilidade
    - Monitoramento
    - Rede
    - Armazenamento
    - Balanceamento de carga
    - Gerenciamento de configurações e secrets
- Simplifica a operação
- Aumenta a resiliência
- Aumenta a segurança

Orquestradores de Containers

kubernetes

Orquestradores de Containers

- Kubernetes é open source;
- Declarativo;
- Extensivo;

ENGENHARIAS
Teoria de Redes

Conclusão

- Repositório da demo: https://github.com/luisfelipe998/docker-demo
- Containers cumprem um papel fundamental na computação distribuída e na nuvem atualmente;
- Containers facilitam o desenvolvimento, sendo possível portar a aplicação pra diferentes máquinas garantindo o mesmo ambiente;
- Docker e Kubernetes são as ferramentas mais famosas para trabalhar com containers, mas existem muitas outras;

Referências

- ADEKANYE, F. **Understanding Docker concepts.** Disponível em https://www.section.io/engineering-education/docker-concepts/ Acesso em 02 Jun 2022
- AGARWAL, N. **Docker Container's Filesystem Desmystified.** Disponível em https://medium.com/@BeNitinAgarwal/docker-containers-filesystem-demystified-b6ed8112a04a Acesso em 02 Jun 2022
- CALIZO, M. **6 container concepts you need to understand.** Disponível em https://opensource.com/article/20/12/containers-101. Acesso em 02 Jun 2022
- CONTAINERD. **Containerd.** Disponível em containerd.io. Acesso em 02 Jun 2022
- CROSBY, M. **What is containerd?** Disponível em https://www.docker.com/blog/what-is-containerd-runtime/ Acesso em 02 Jun 2022
- DOCKER. **Container Network**. Disponível em: https://docs.docker.com/config/containers/container-networking/ Acesso em 02 Jun 2022
- DOCKER. **Docker Desktop**. Disponível em: https://www.docker.com/products/docker-desktop/ Acesso em 02 Jun 2022
- DOCKER. **Docker Overview**. Disponível em: https://docs.docker.com/get-started/overview/ Acesso em 02 Jun 2022
- DOCKER. **Manage Data in Docker**. Disponível em: https://docs.docker.com/storage/ Acesso em 02 Jun 2022
- DOCKER. **Persist the DB**. Disponível em: https://docs.docker.com/get-started/05_persisting_data/ Acesso em 02 Jun 2022
- DOCKER. **Use Bind Mounts**. Disponível em: https://docs.docker.com/storage/bind-mounts/ Acesso em 02 Jun 2022
- DOCKER. **Use Bridges**. Disponível em: https://docs.docker.com/network/bridge/ Acesso em 02 Jun 2022
- DOCKER. **Use Volumes**. Disponível em: https://docs.docker.com/storage/volumes/ Acesso em 02 Jun 2022
- DOCKER. **Use containers to Build, Share and Run your applications.** Disponível em https://www.docker.com/resources/what
- -container/#:~:text=A%20Docker%20container%20image%20is,tools%2C%20system%20libraries%20and%20settings. Acesso em 02 Jun 2022
- KUBERNETES. **Kubernetes.** Disponível em kubernetes.io Acesso em 02 Jun 2022
- MCCARTY, S. **A Practical Introduction to Container Terminology.** Disponível em https://developers.redhat.com/blog/2018/02/22/container-terminology-practical-introduction# Acesso em 02 Jun 2022
- NETAPP. **What are containers.** Disponível em https://www.netapp.com/devops-solutions/what-are-containers/#:~:text=Benefits%20of%20containers,-Containers%20are%20a&text=Containers%20require%20less%20system%20resources,t%20include%20operating%20system%20images.&text=Applications%20running%20in%20containers%20can,operating%20systems%20and%20hardware%20platforms. Acesso em 02 Jun 2022
- SANTOS, L. **Entendendo Runtimes de Containers.** Disponível em: https://blog.lsantos.dev/entendendo-runtimes-de-containers/ Acesso em 02 Jun 2022
- SANTOS, L. **Kubernetes sem Docker? – Entendendo OCI, CRI, e o ecossistema de containers.** Disponível em https://blog.lsantos.dev/oci-cri-docker-ecossistema-de-containers/ Acesso em 02 Jun 2022
- SIMANUPANG, I. **Daemonless Container Engine.** Disponível em: https://medium.com/easyread/daemonless-container-engine-5364394b80ec. Acesso em 02 Jun 2022
- SIMPLILEARN. **What is Docker: Advantages and Components** Disponível em https://www.simplilearn.com/tutorials/docker-tutorial/what-is-docker Acesso em 02 Jun 2022
- SITE24X7. **How do containers work.** Disponível em https://www.site24x7.com/learn/containers/how-containers-work.html#:~:text=will%20utilize%20it.-,Containers%20are%20an%20abstraction%20in%20the%20application%20layer%2C%20whereby%20code,running%20as%20an%20isolated%20process. Acesso em 02 Jun 2022
- SUPALOV, V. **What are Docker image layers?** Disponível em https://vsupalov.com/docker-image-layers/ Acesso em 02 Jun 2022
- VELICHKO, I. **Why and How to Use Containerd From Command Line.** Disponível em https://iximiuz.com/en/posts/containerd-command-line-clients/ Acesso em 02 Jun 2022
- VMWARE. **What is container orchestration?** Disponível em https://www.vmware.com/topics/glossary/content/container-orchestration.html#:~:text=Container%20orchestration%20is%20the%20automation,networking%2C%20load%20balancing%20and%20more. Acesso em 02 Jun 2022
- WALKER, J. **What is Podman and How does it differ from Docker?** Disponível em https://www.howtogeek.com/devops/what-is-podman-and-how-does-it-differ-from-docker/#:~:text=In%20Podman%2C%20containers%20can%20form,to%20the%20Kubernetes%20Pod%20concept.&text=The%20Pod%20concept%20is%20powerful,and%20manage%20them%20in%20unison. Acesso em 02 Jun 2022
- WAVEWORKS. **Comparing Container Orchestrators: 6 choices analyzed.** Disponível em https://www.weave.works/blog/comparing-container-orchestration/ Acesso em 02 Jun 2022

OBRIGADO.

UNISINOS

DESAFIE
O AMANHÃ.