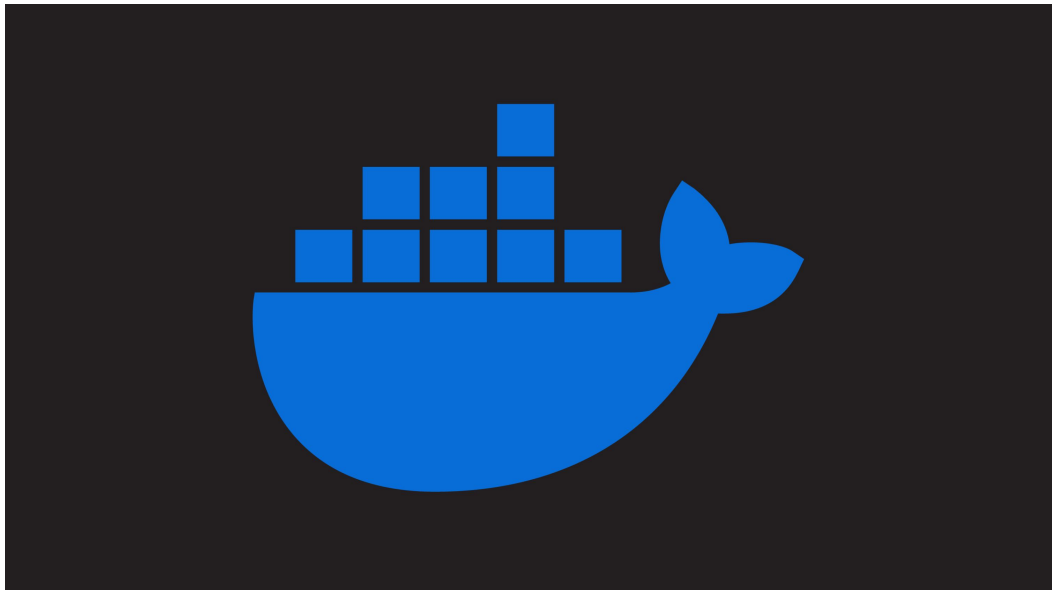


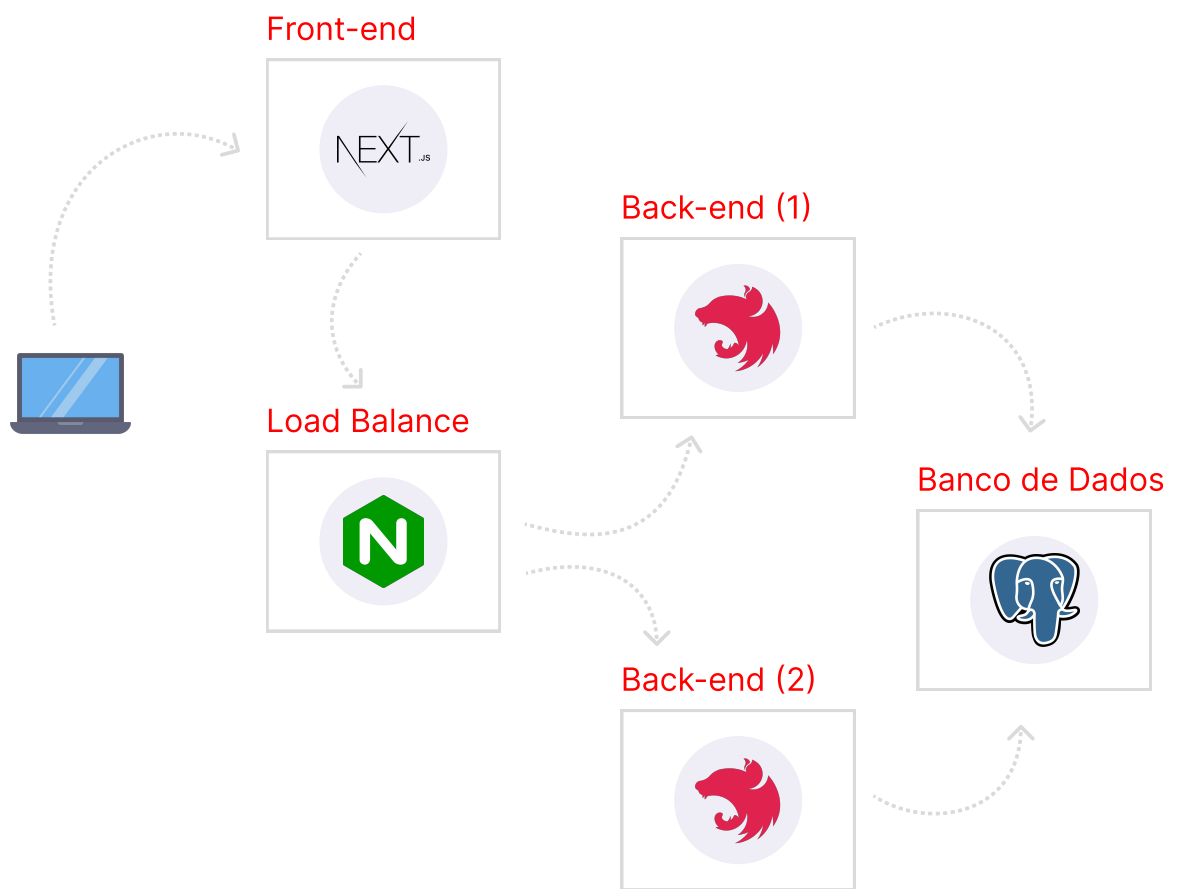
ARGUS ACADEMY

▶ Docker

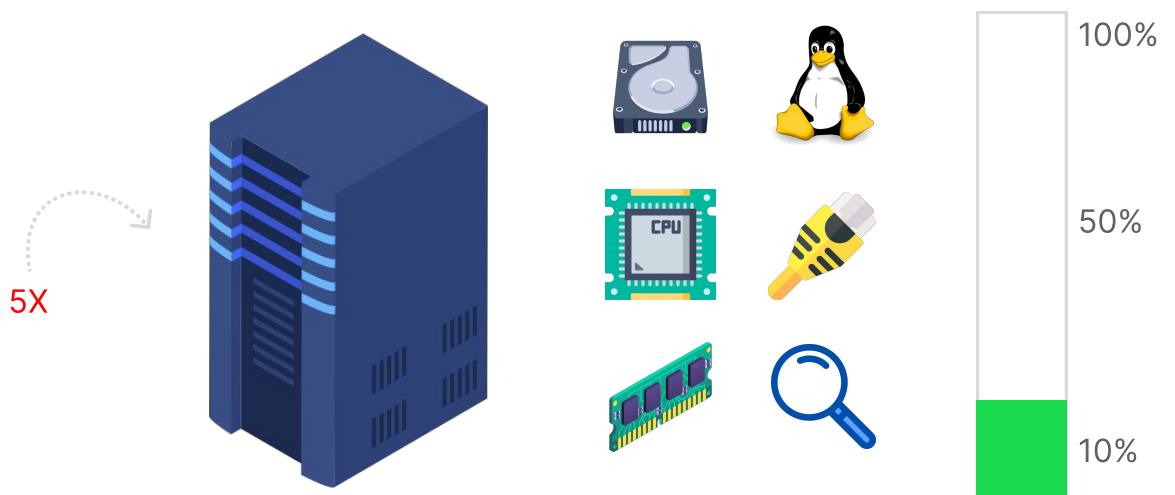
Dominando a Criação e o Gerenciamento de Containe



Aplicações vs Arquitetura de Infraestrutura

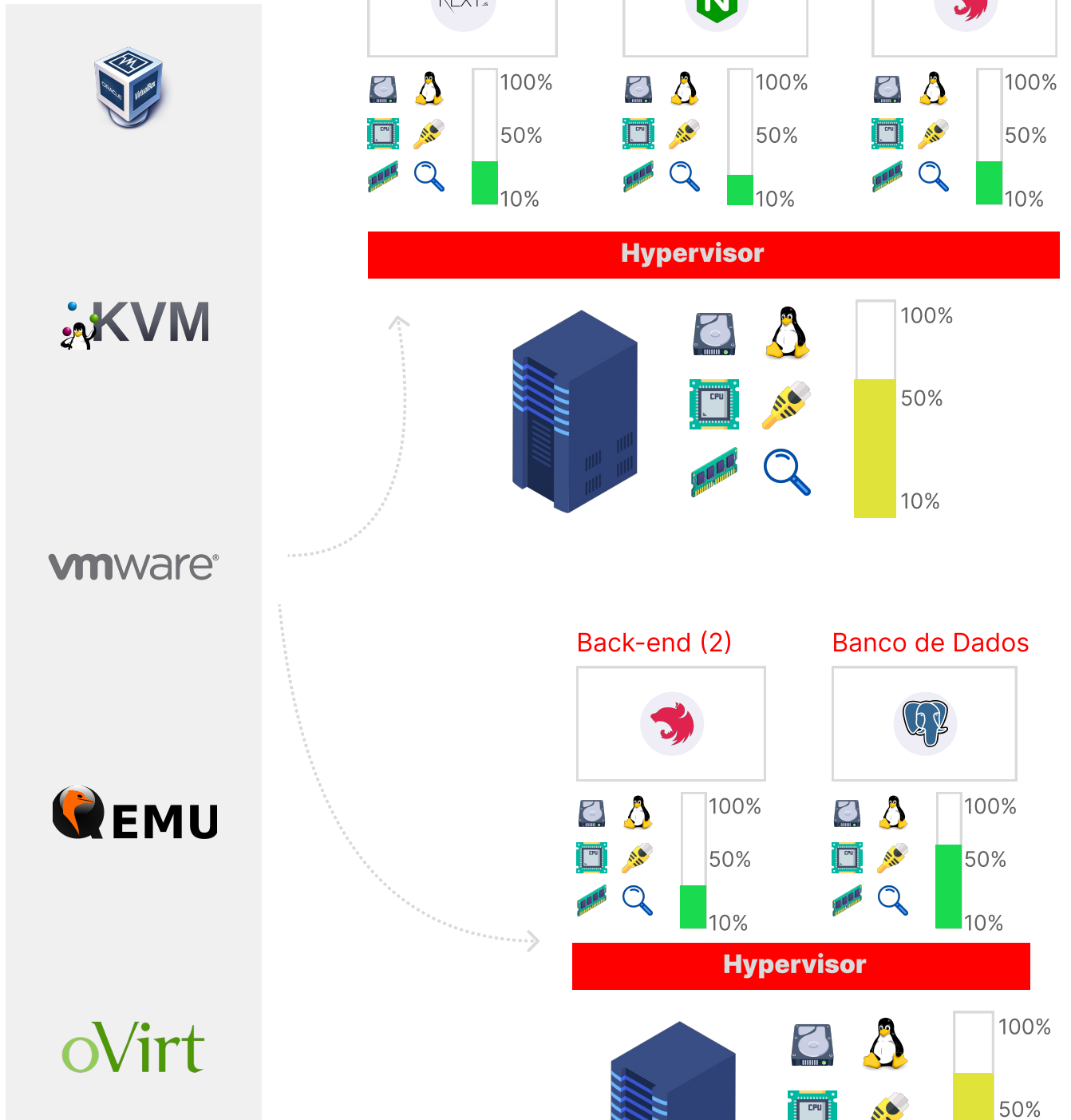


Aplicações vs Servidores Dedicados



► Introdução

Aplicações vs Máquinas Virtuais

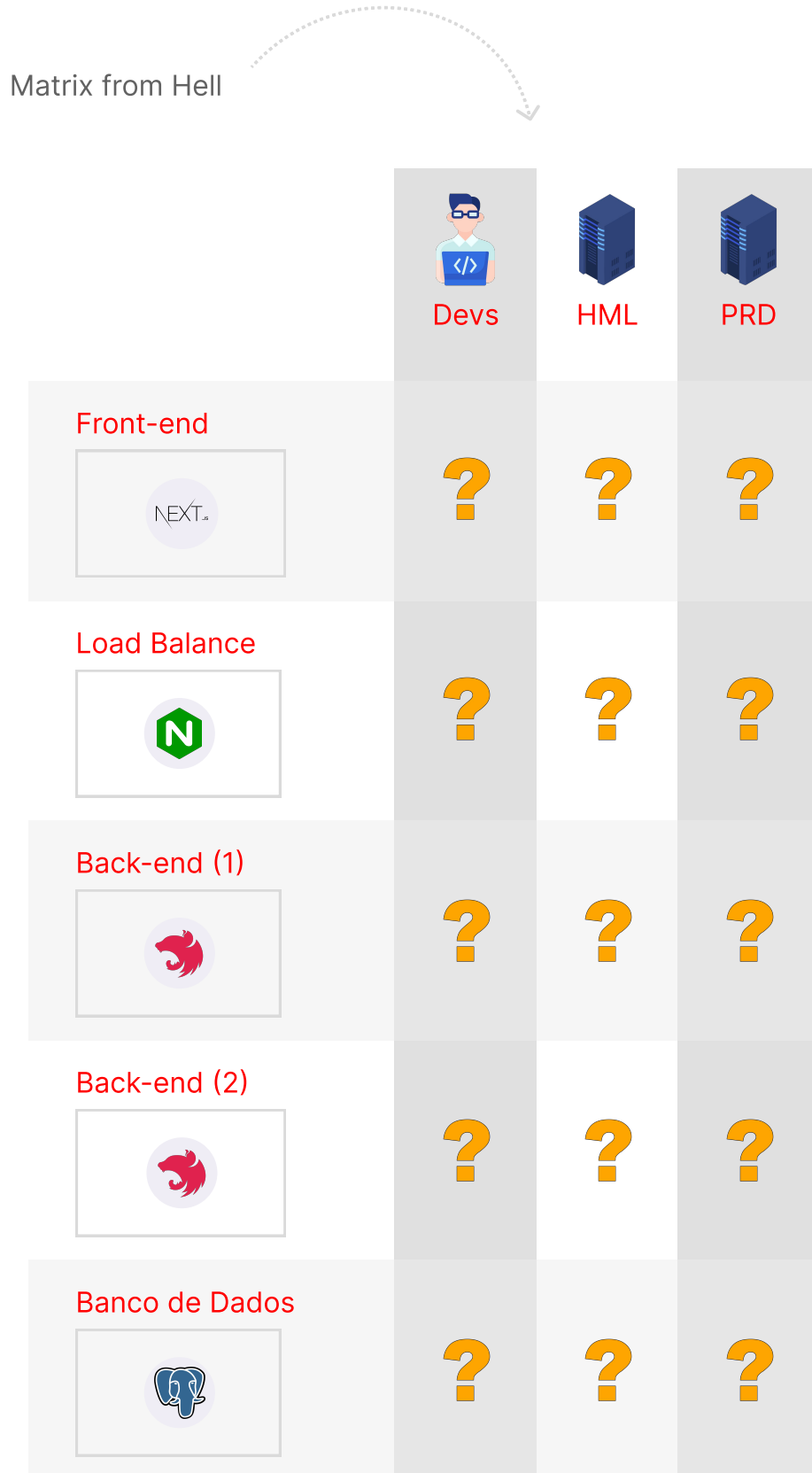


► Introdução

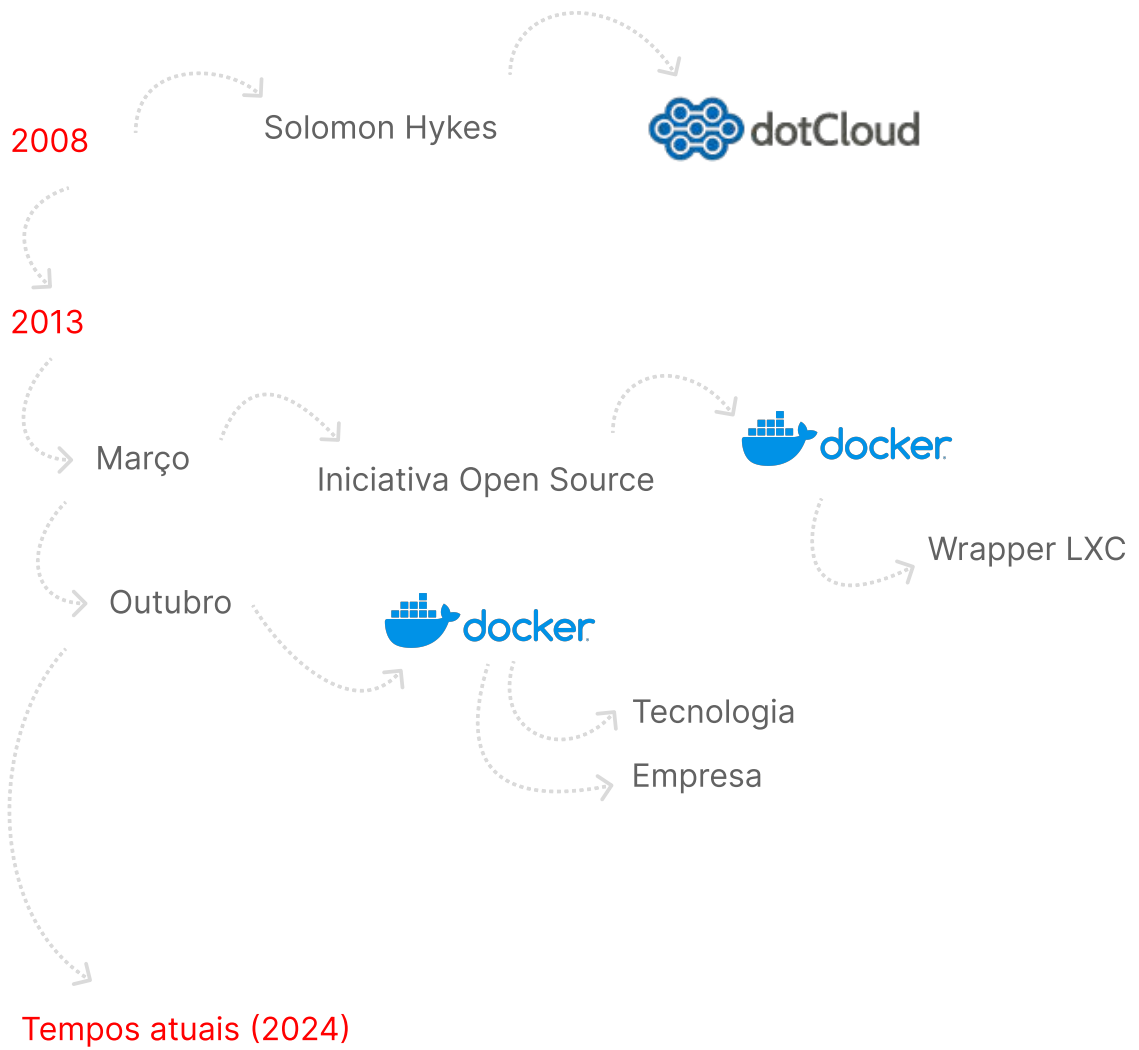
Containers são exclusividade do Docker?



Aplicações vs Containers (produtividade)

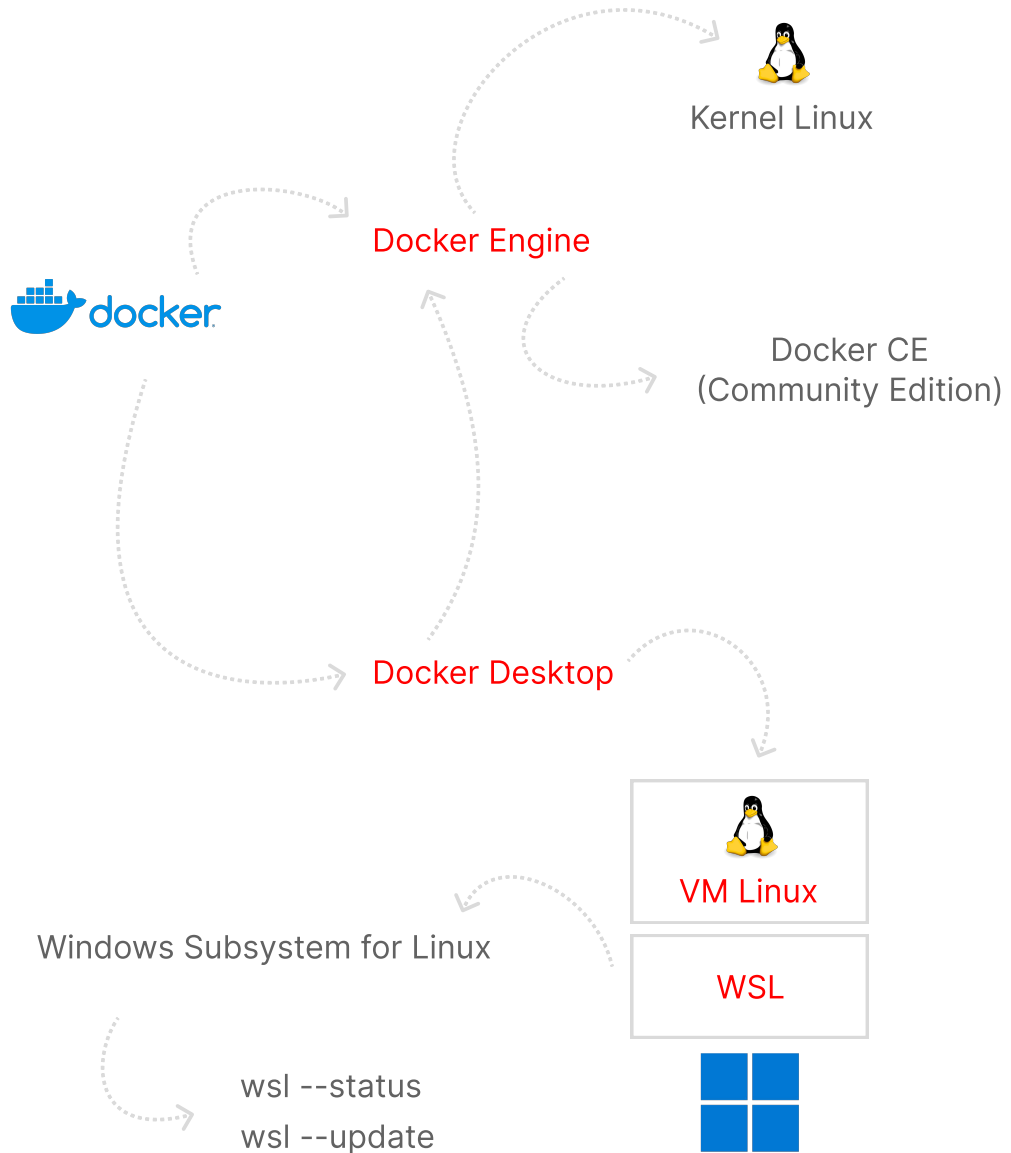


Conhecendo a empresa e a plataforma Docker



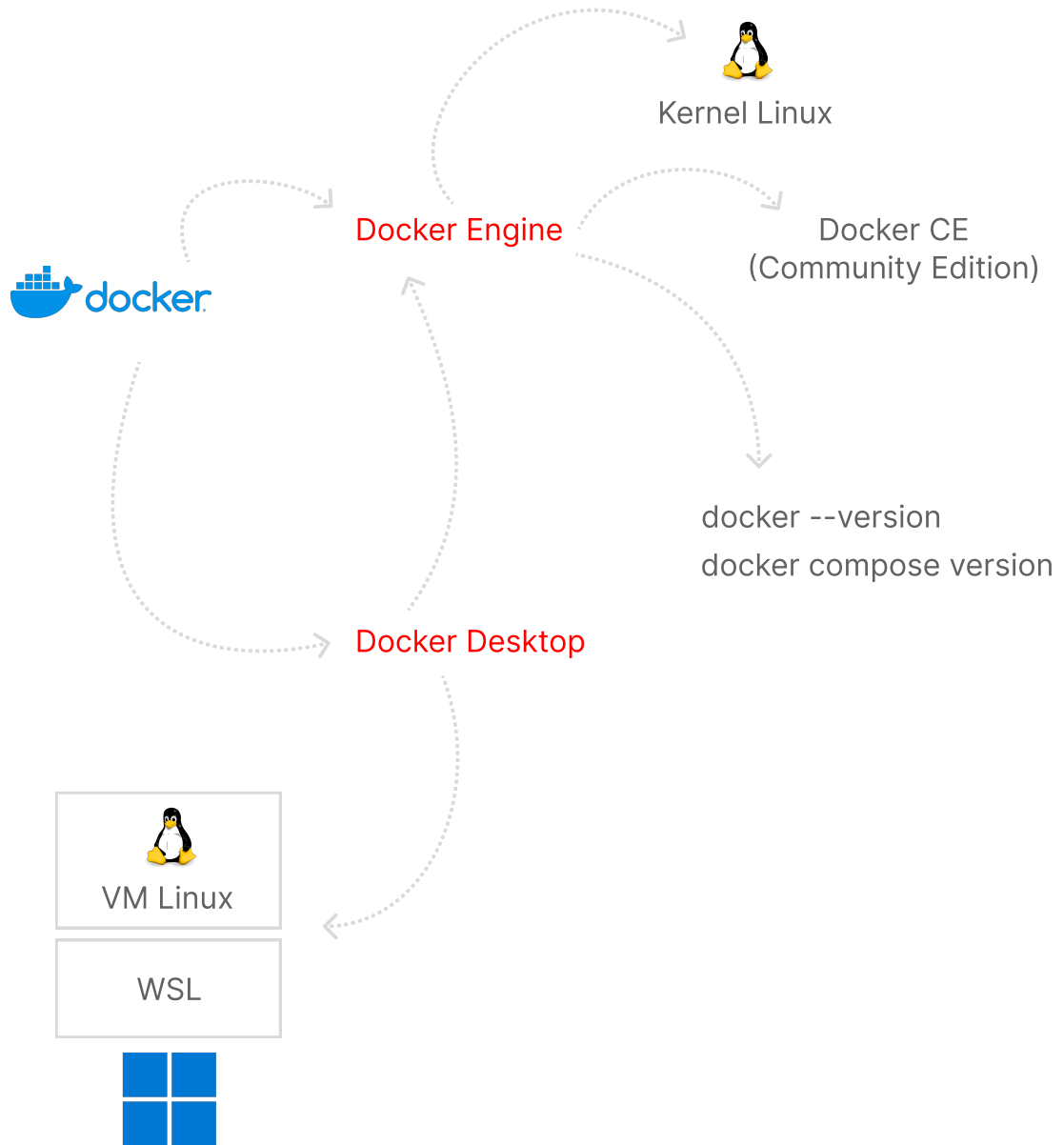
► Configurando o ambiente

[Windows 11 Pro] Configurando o WSL (Windows Subsystem for Linux)



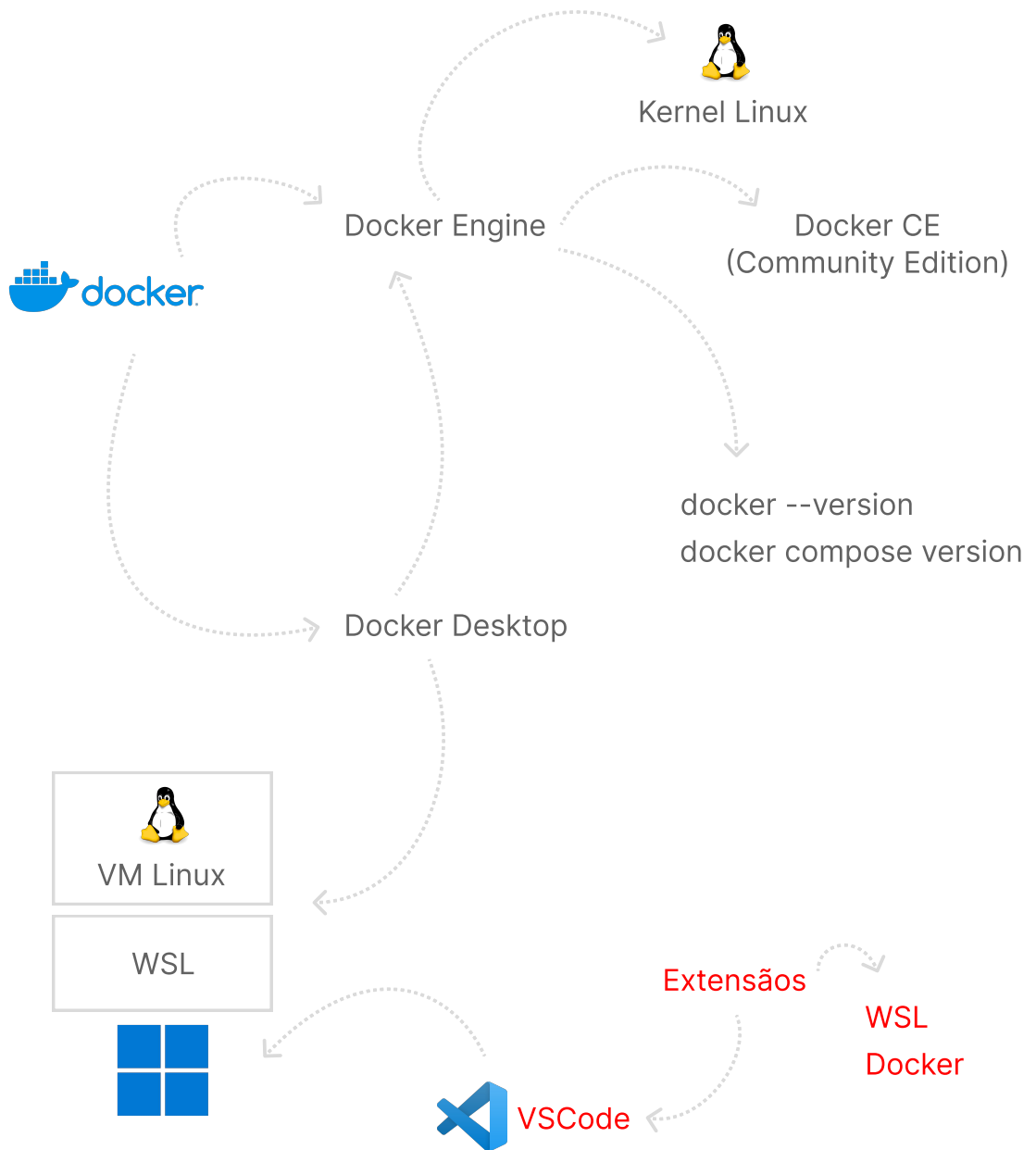
► Configurando o ambiente

[Windows 11 Pro] Instalando o Docker Desktop



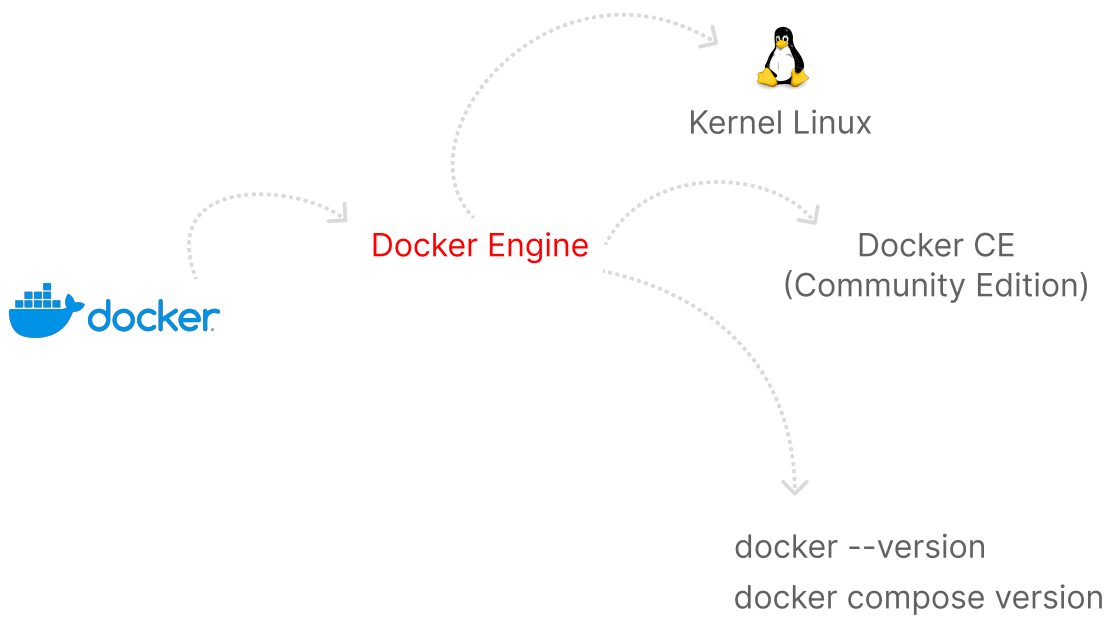
► Configurando o ambiente

[Windows 11 Pro] Instalando o VSCode



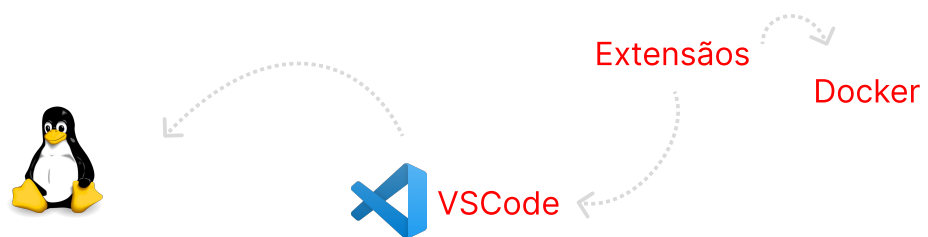
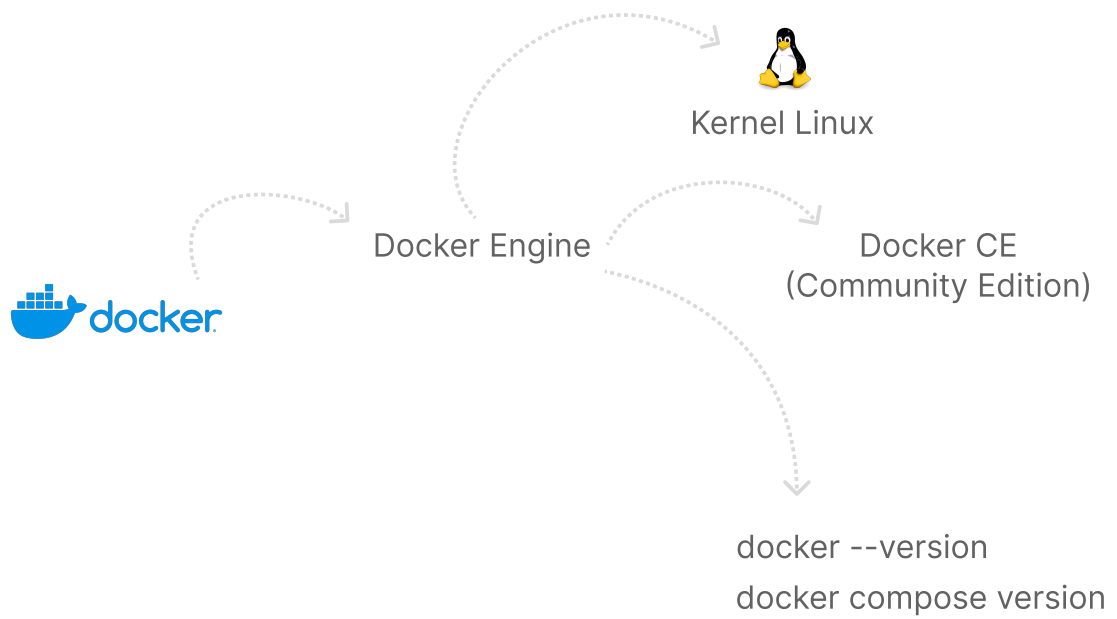
► Configurando o ambiente

[Linux Ubuntu 22] Instalando o Docker Engine



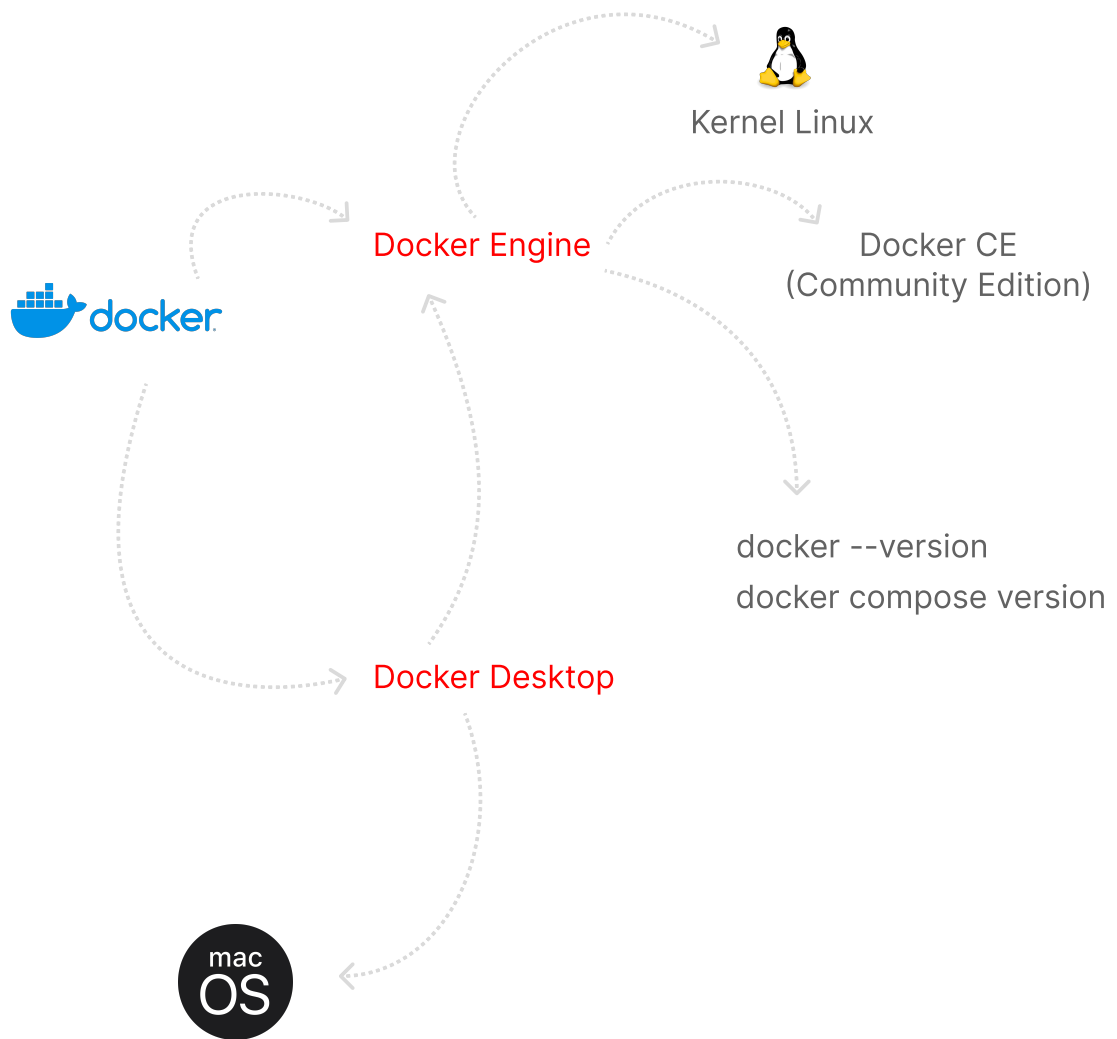
► Configurando o ambiente

[Windows 11 Pro] Instalando o VSCode



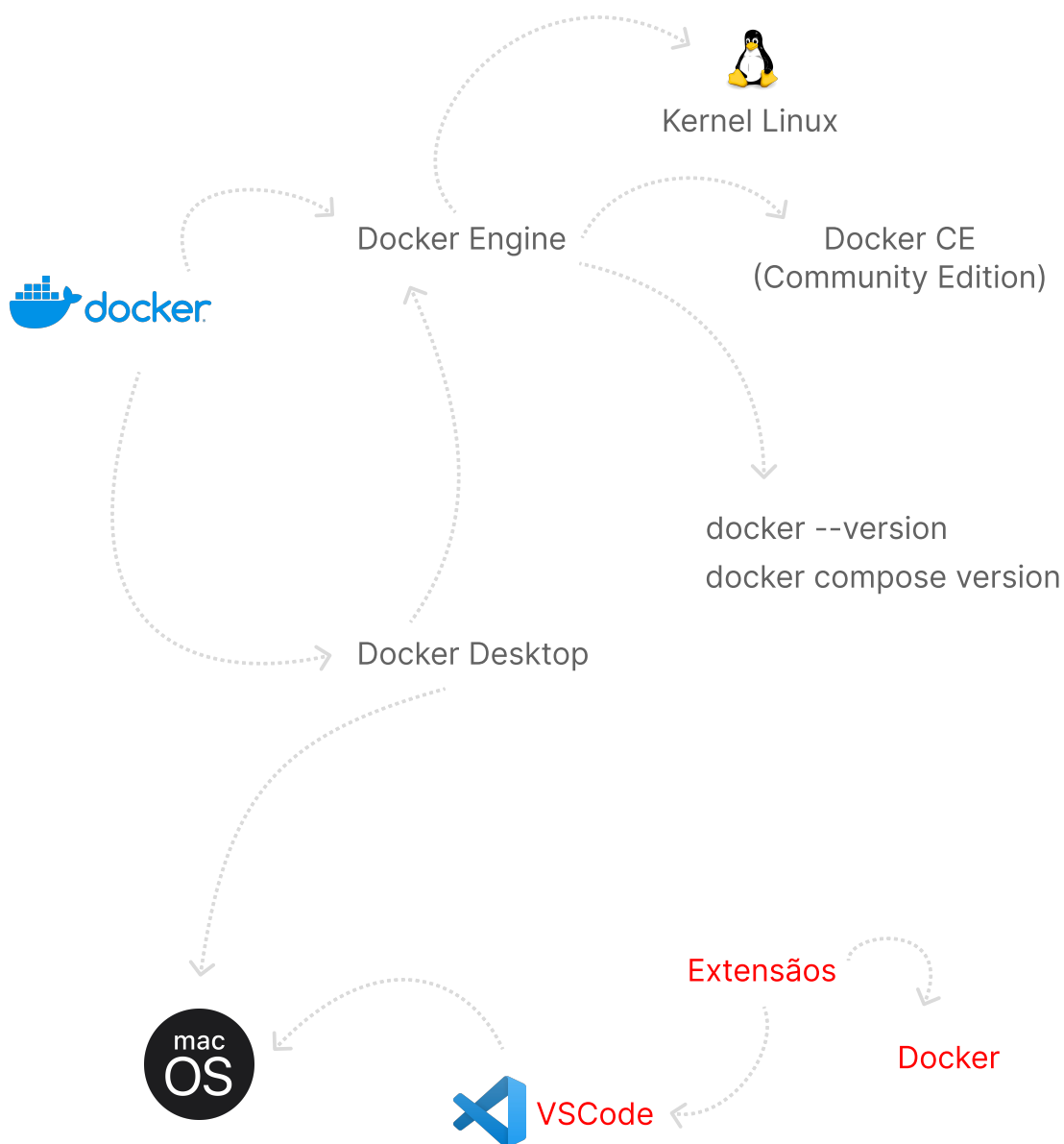
► Configurando o ambiente

[MacOS] Instalando o Docker Desktop

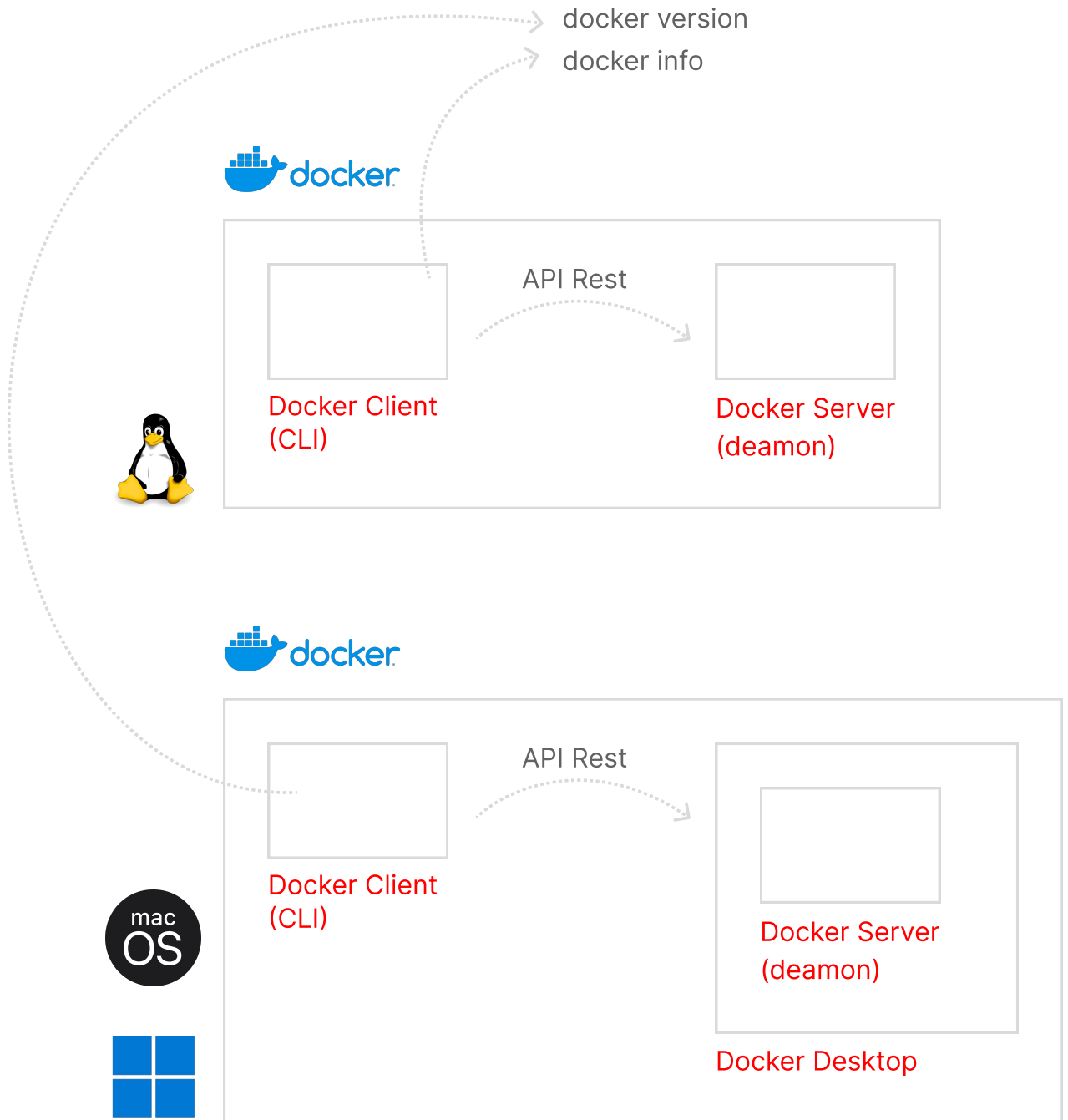


► Configurando o ambiente

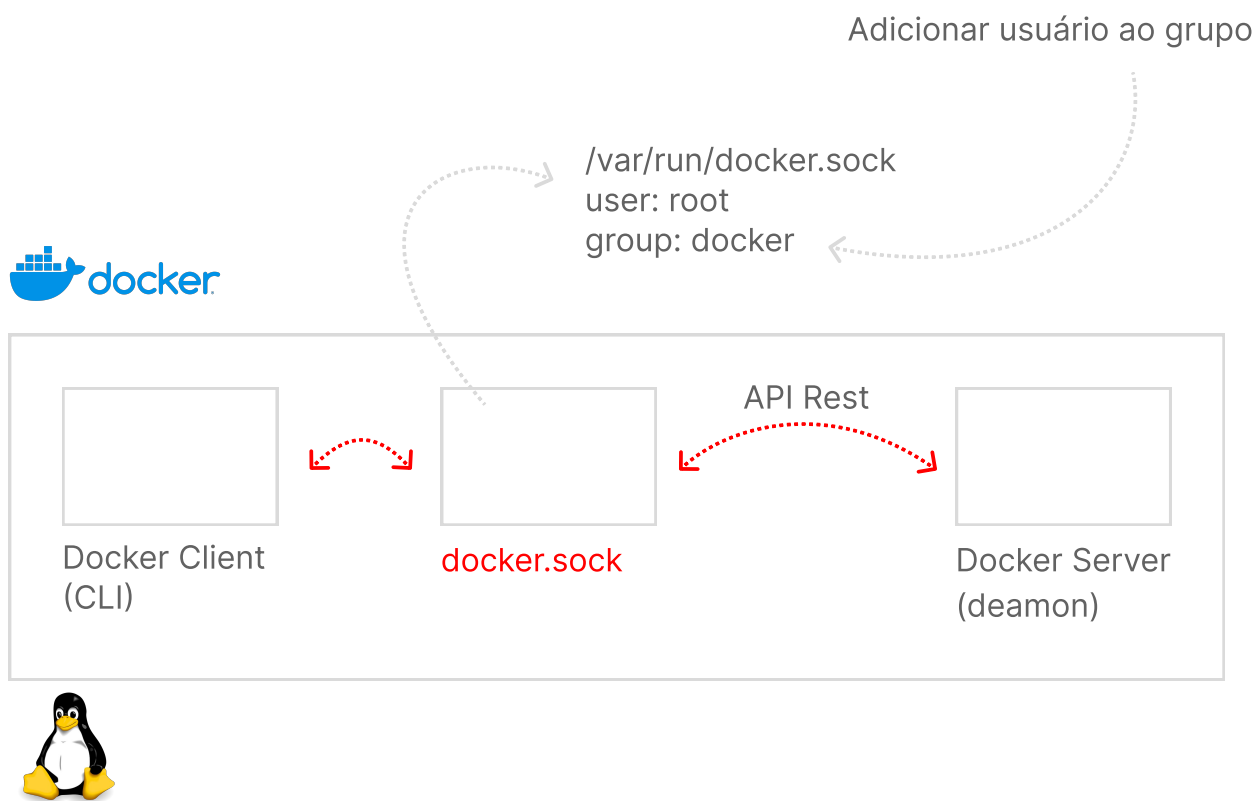
[MacOS] Instalando o VSCode



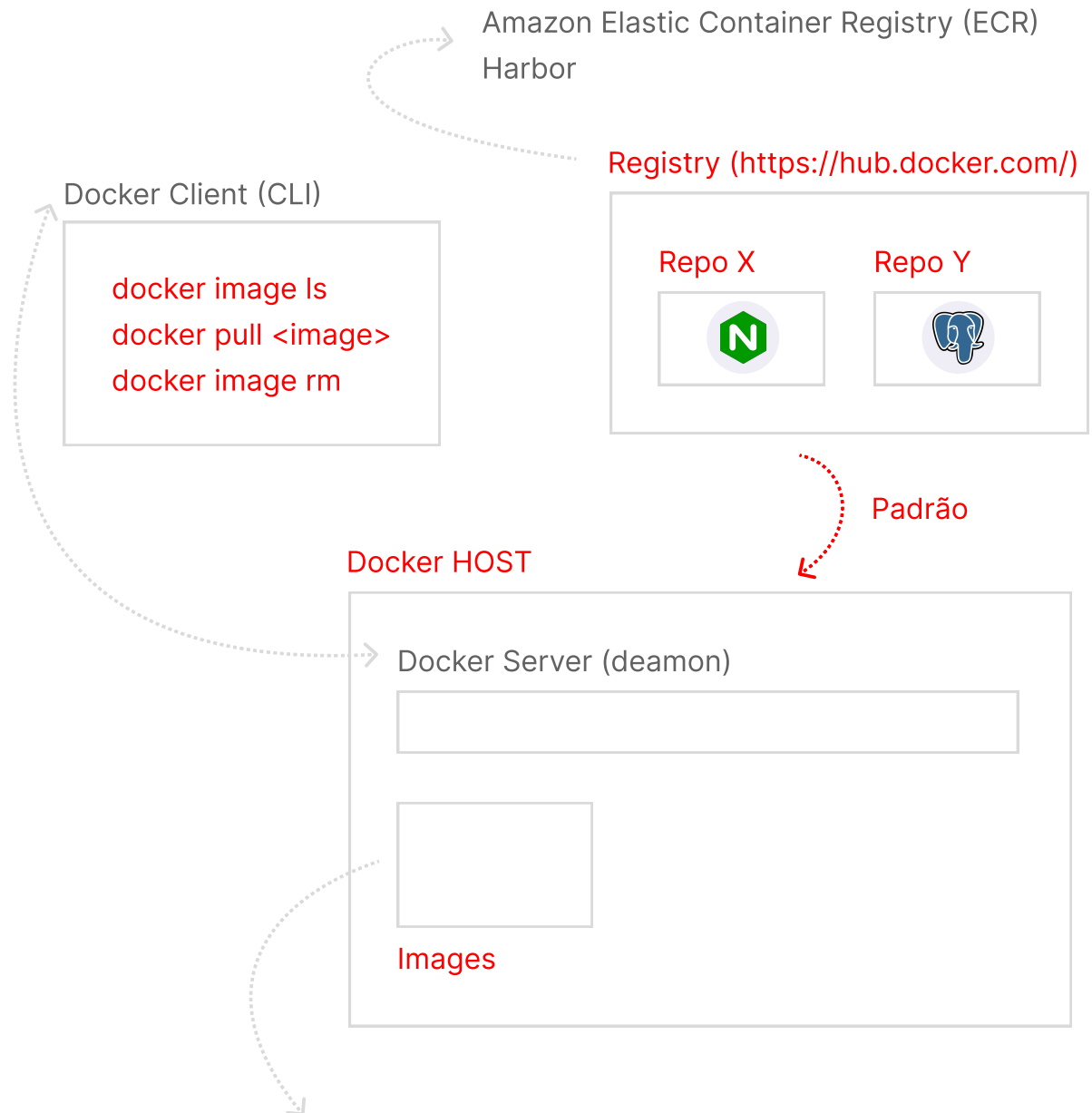
Conhecendo o Docker Client e o Docker Server



Configurando o acesso ao Docker Sock no Linux

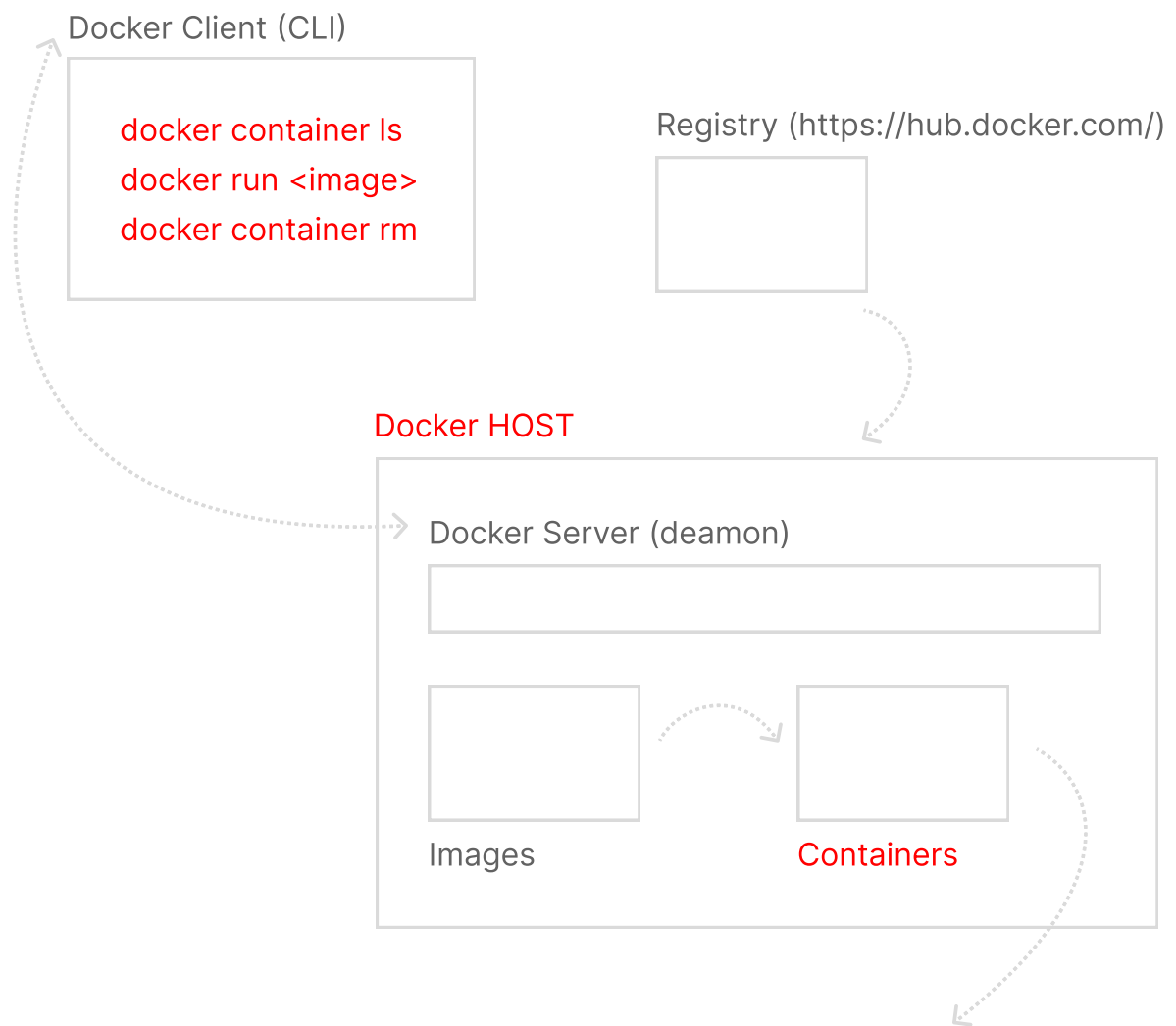


Introdução as Images e ao Docker Hub



Uma imagem é um arquivo executável e auto-contido, com tudo o que necessário para executar o seu objetivo

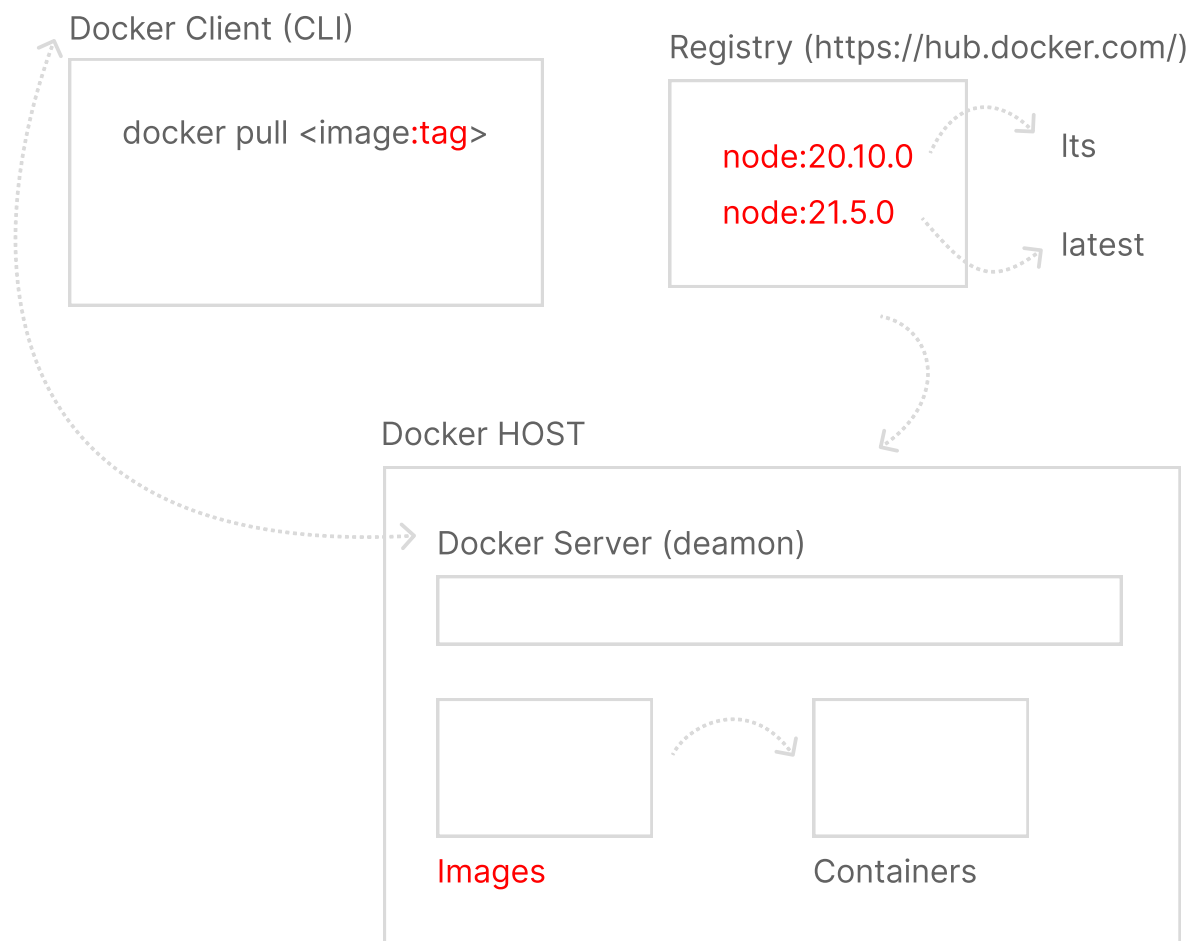
Introdução aos Containers



Um container (contêiner) é a instância de uma imagem em execução

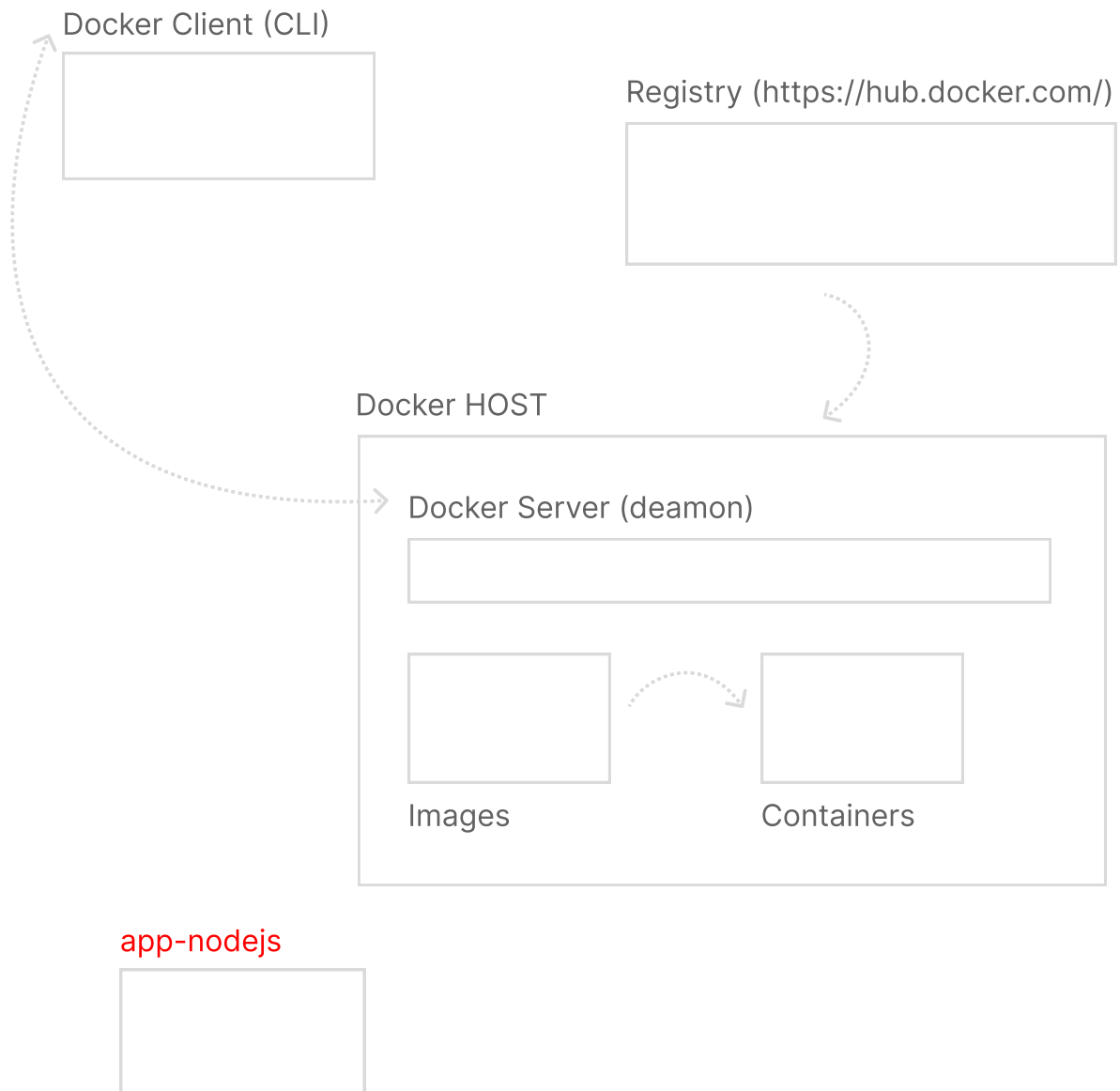
► Images

Baixando imagens de tags específicas



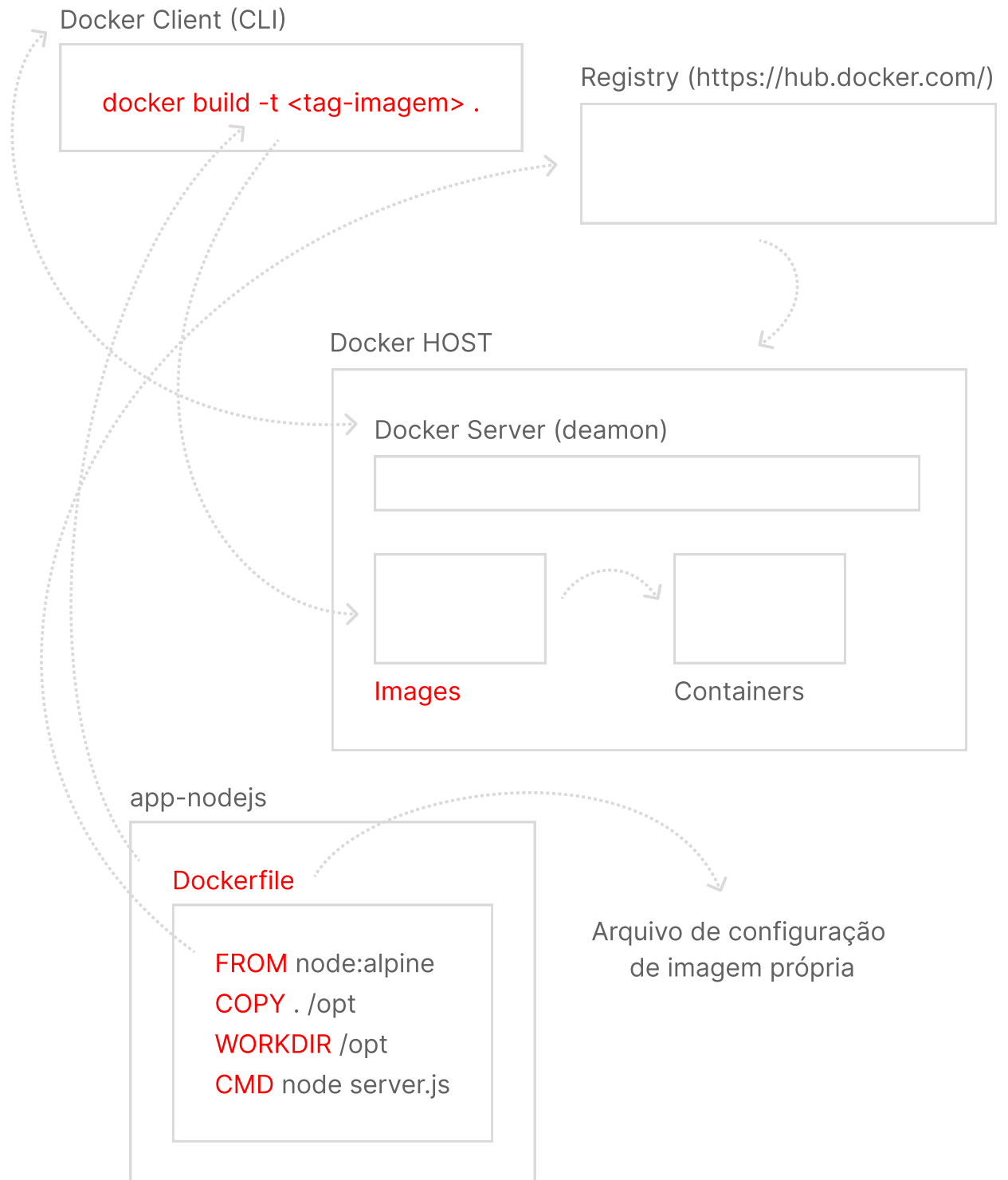
► Images

Criando uma imagem: Criando uma aplicação em NodeJS



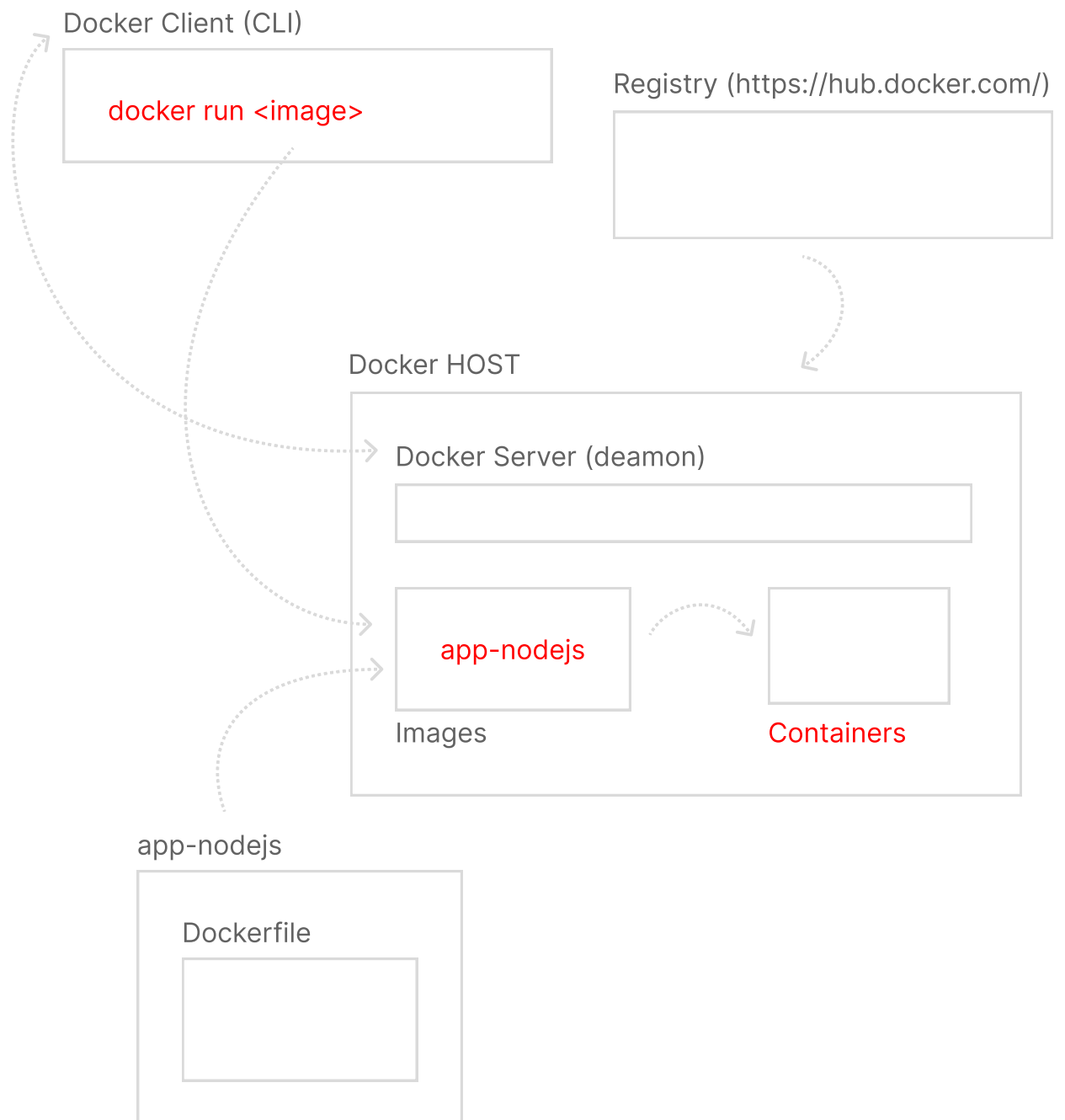
► Images

Criando uma imagem: Dockerfile e o build da imagem



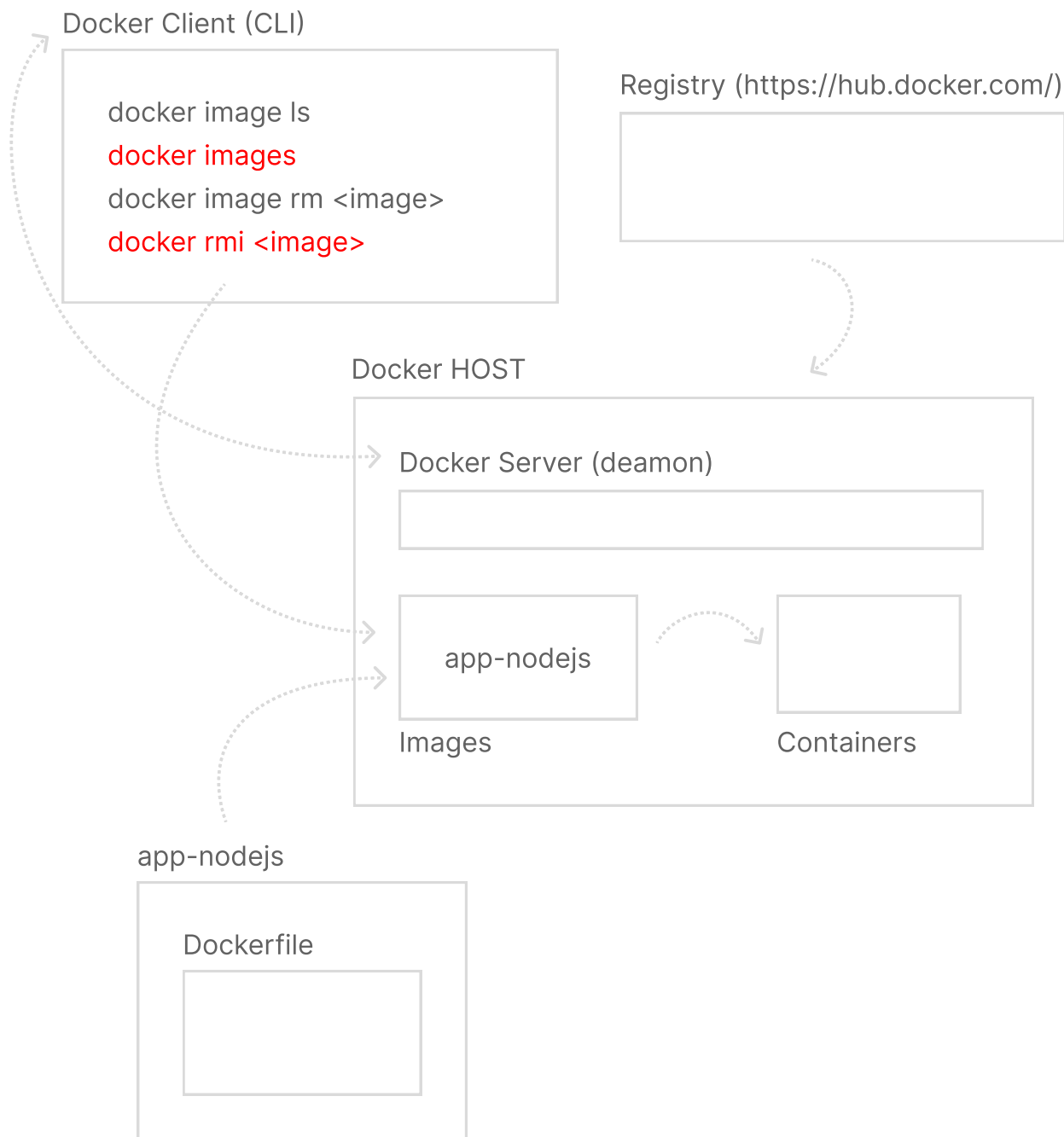
► Images

Criando uma imagem: Executando um container à partir da imagem

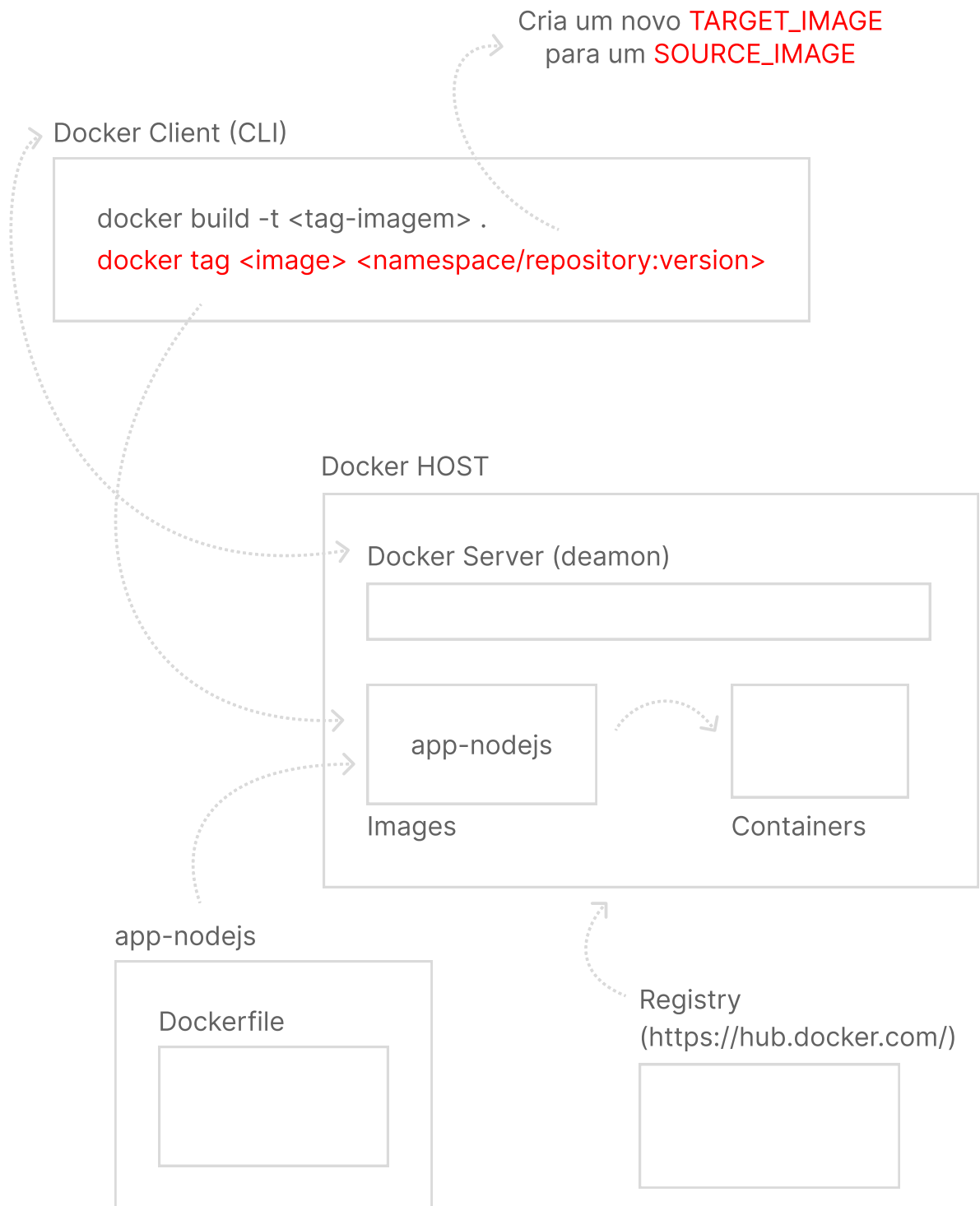


► Images

Comandos opcionais para listagem e remoção de imagens

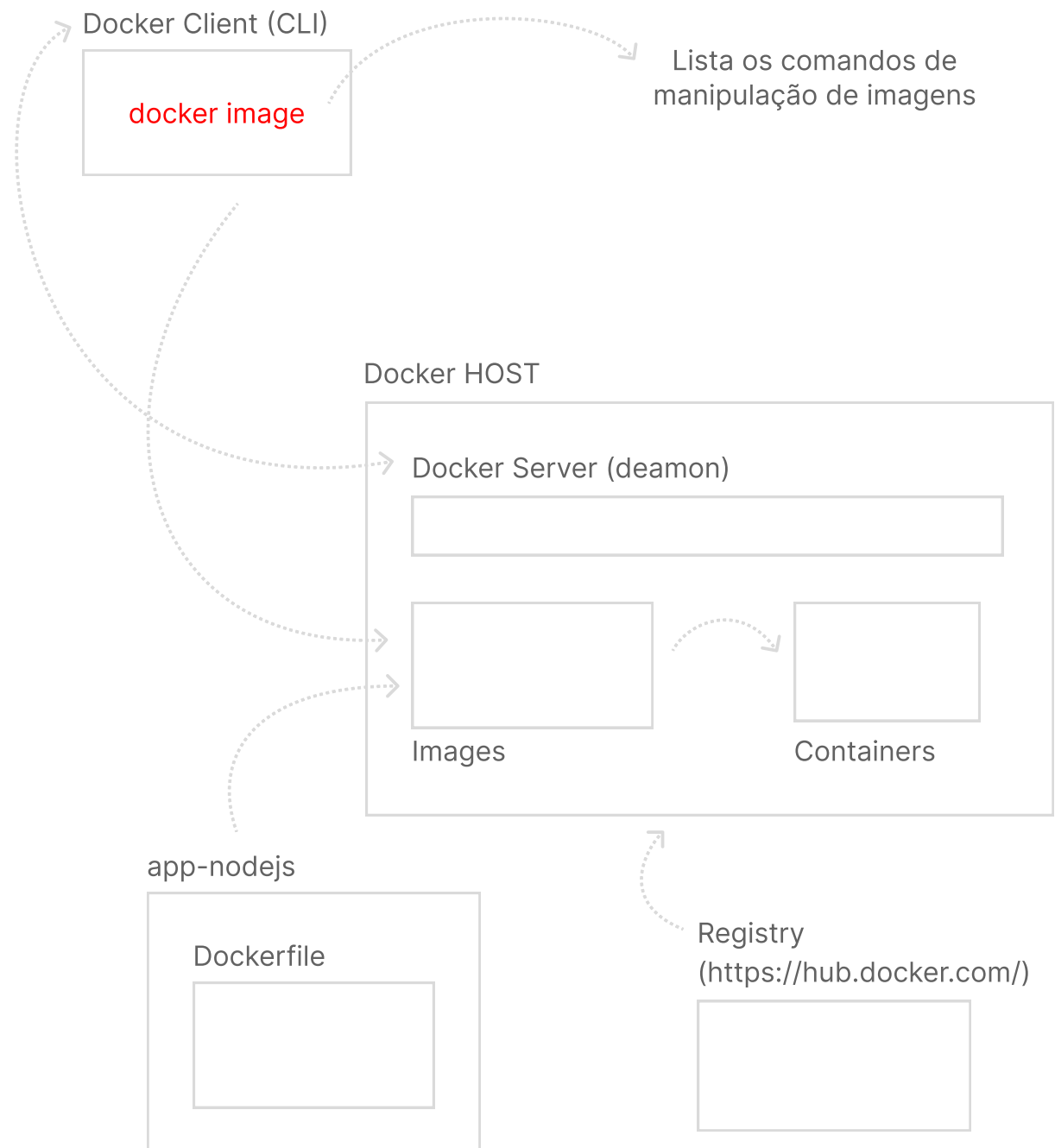


Etiquetando imagens



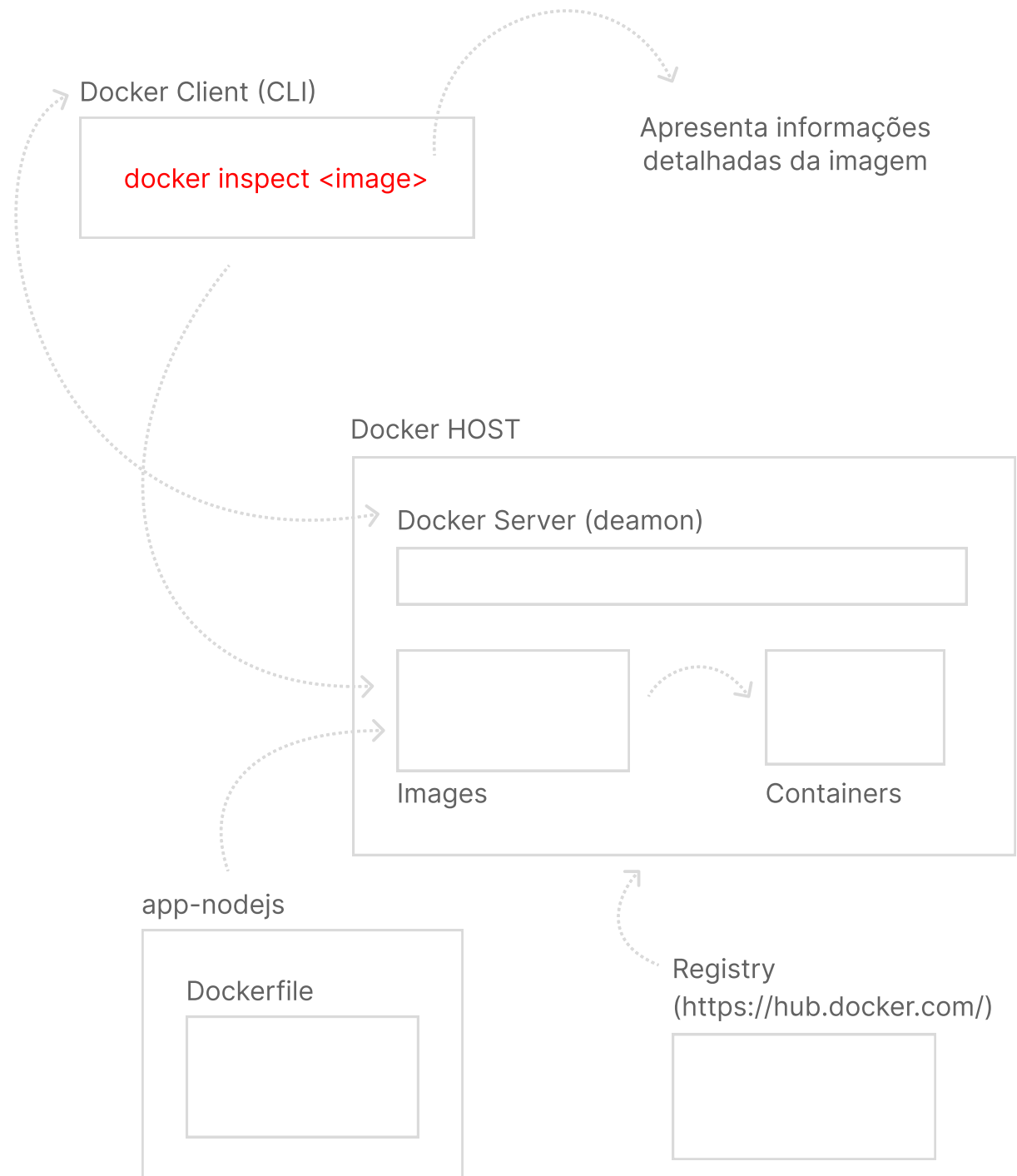
► Images

Exibindo os comandos de manipulação de imagens



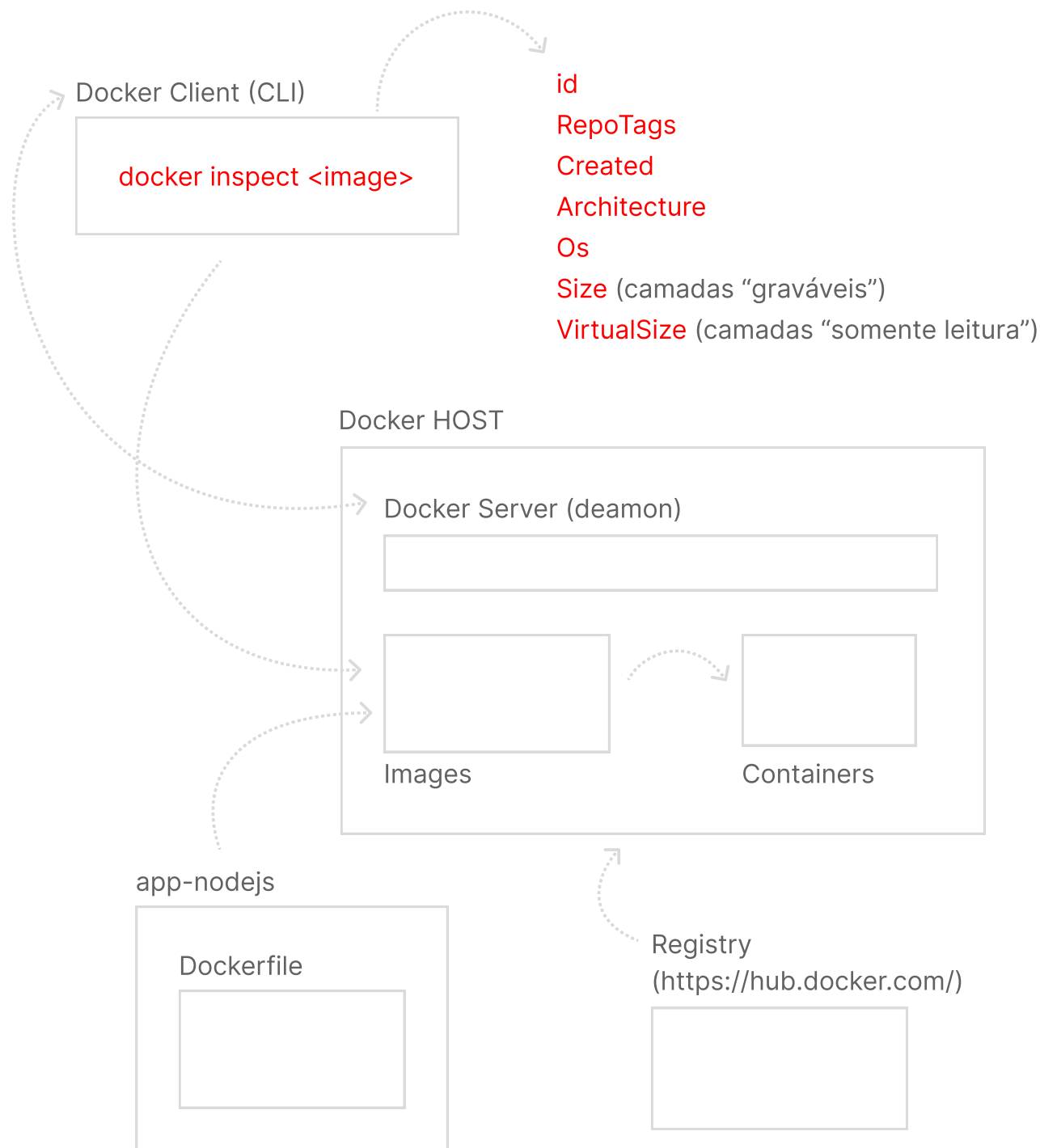
► Images

Inspecionando imagens

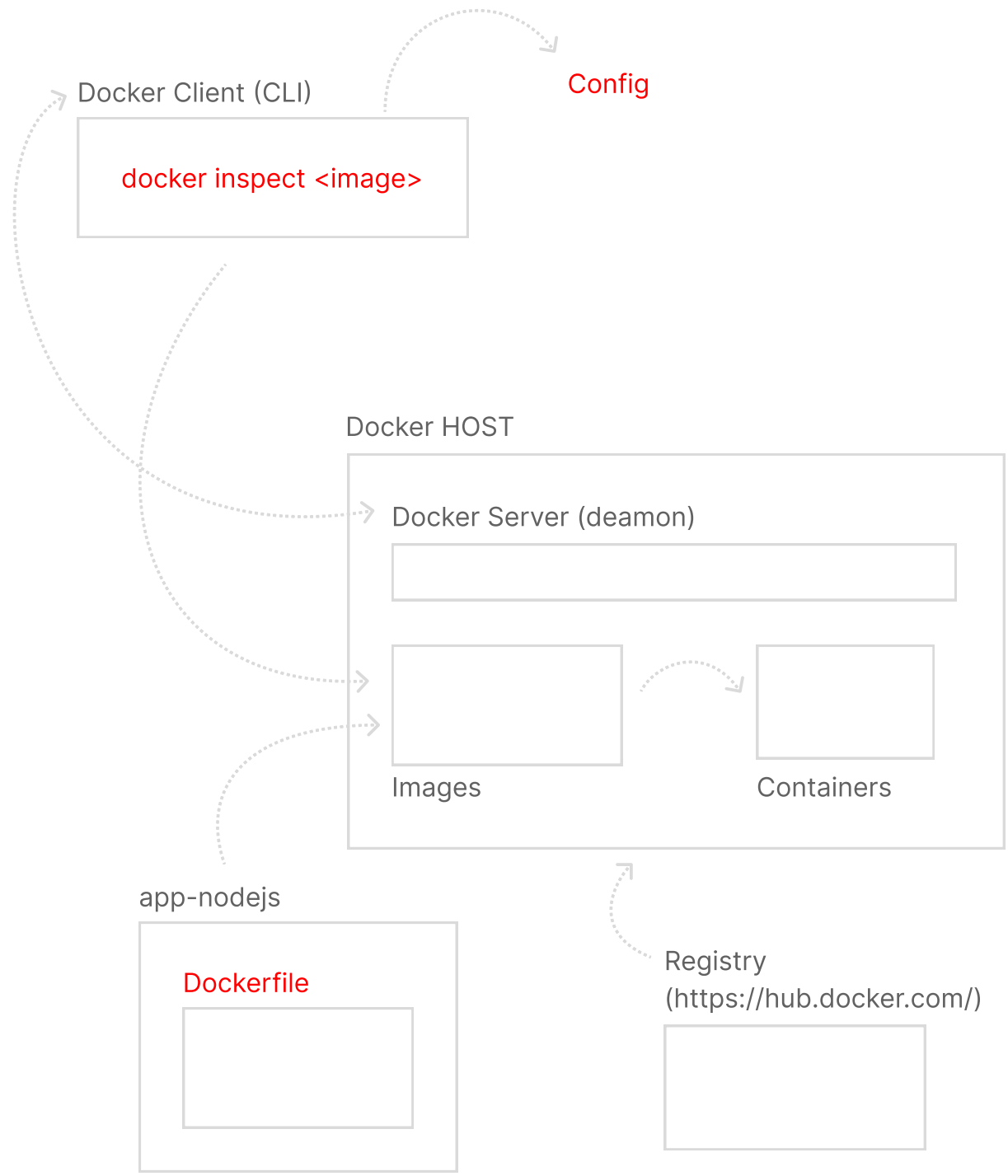


► Images

Inspecionando imagens - Detalhes 1

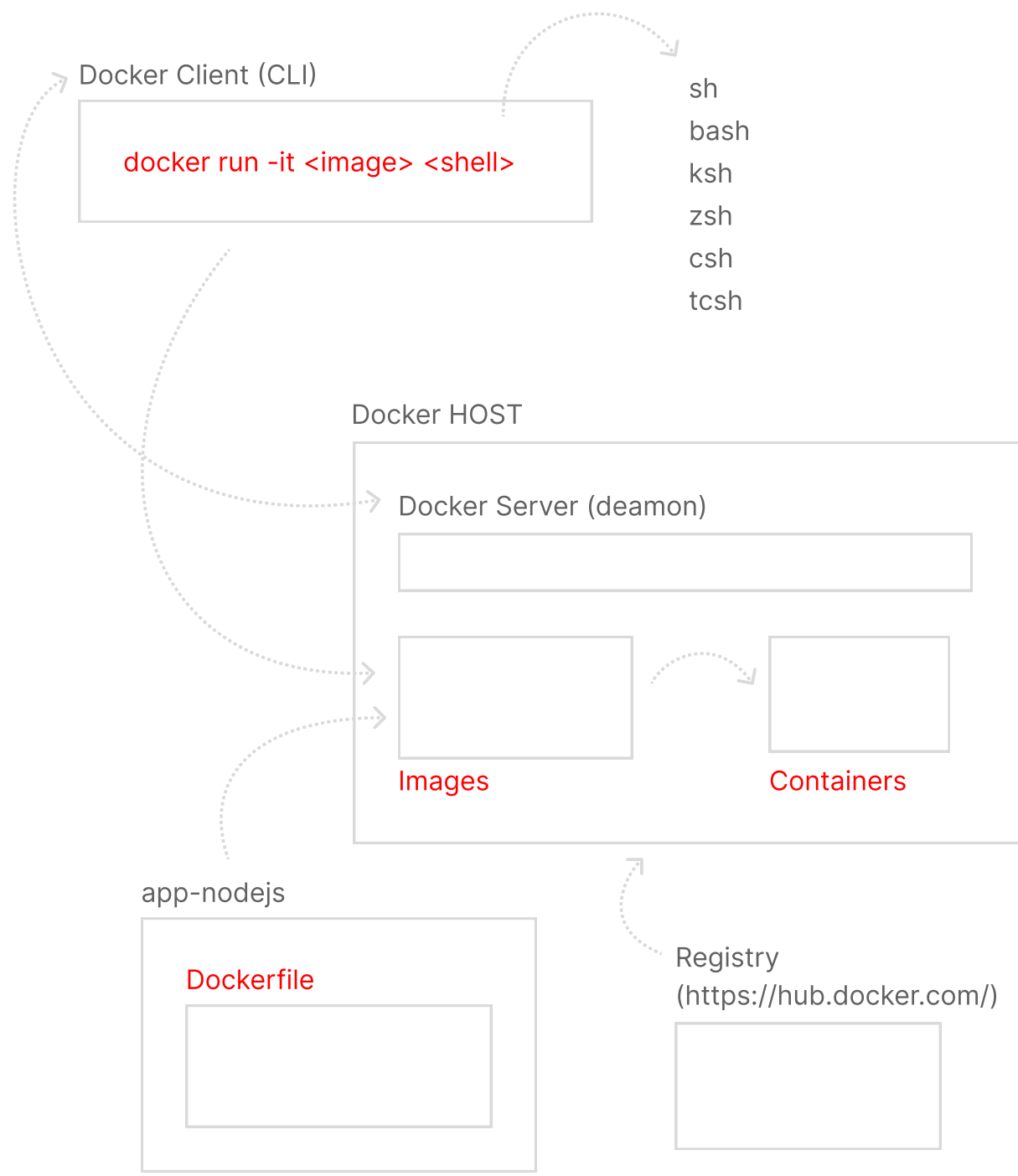


Inspecionando imagens - Detalhes 2



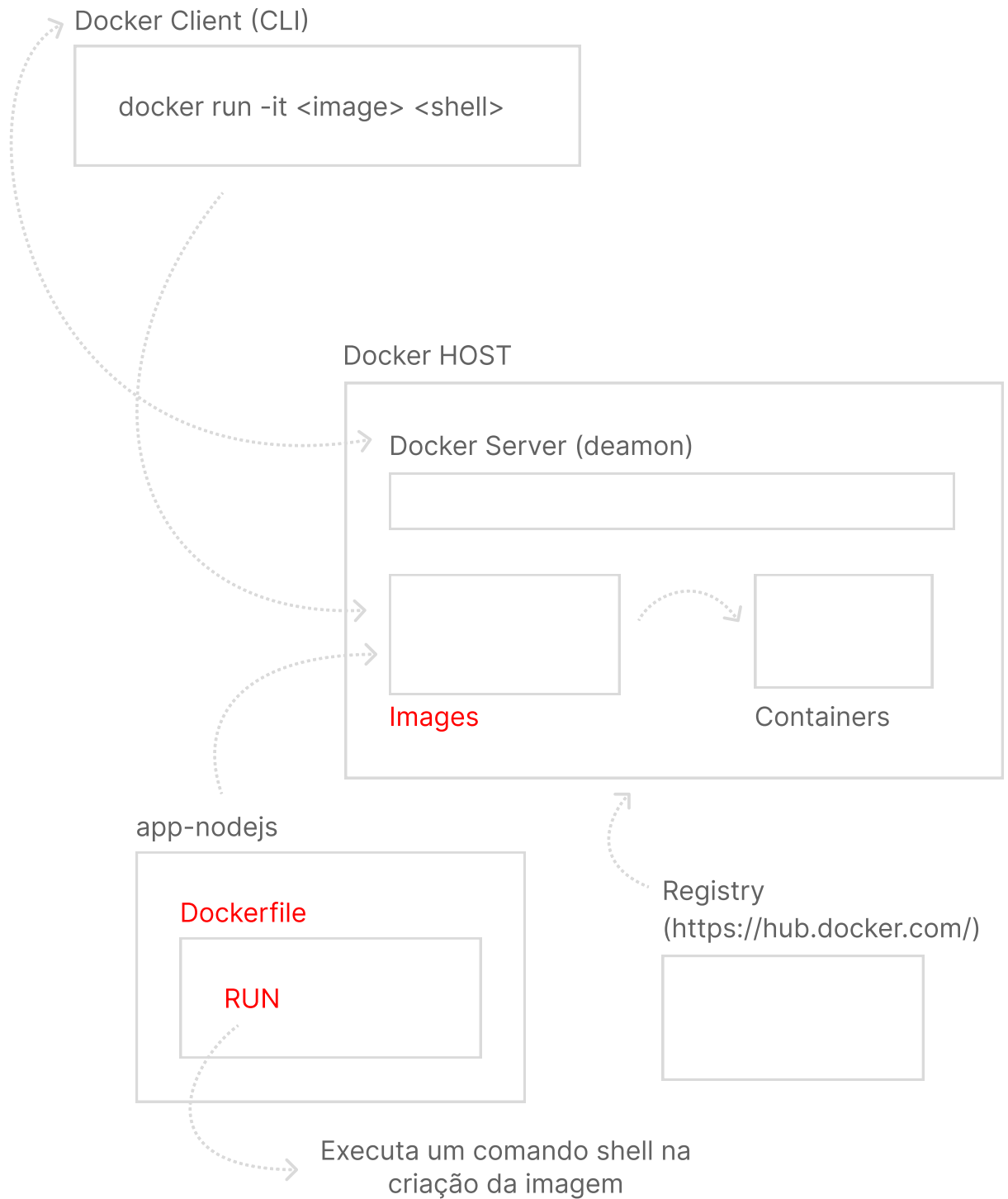
► Images

Acessando o container durante a inicialização via terminal interativo



► Images

Dockerfile - RUN



Dockerfile - RUN multilinha

Dockerfile

```
RUN comando1
RUN comando2
RUN comando3
```

Dockerfile

```
RUN comando1 && \
    comando2 && \
    comando3
```

Dockerfile

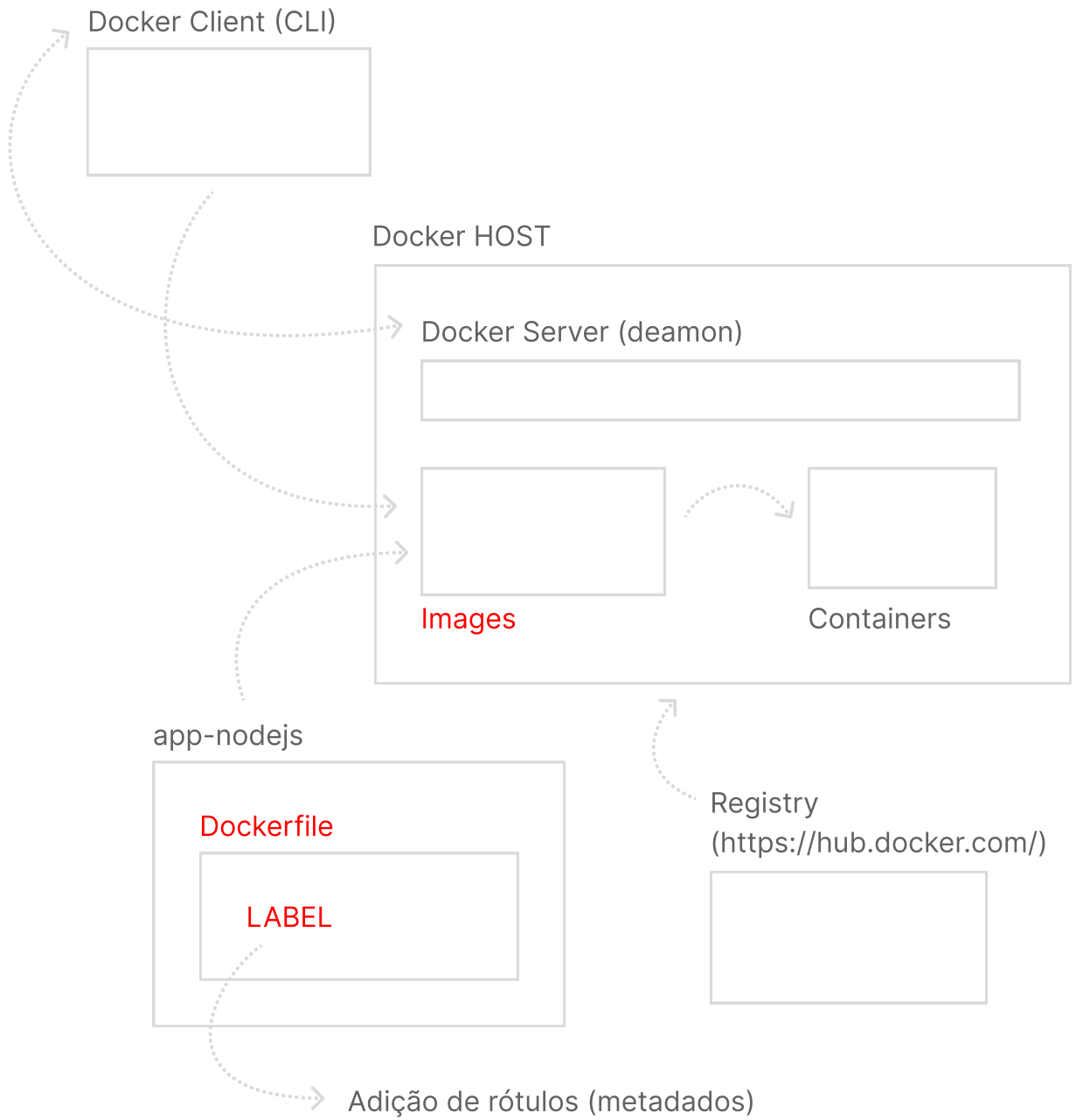
```
RUN comando1 \
    && comando2 \
    && comando3
```

Dockerfile

```
RUN <<EOF
    comando1
    comando2
    comando3
EOF
```

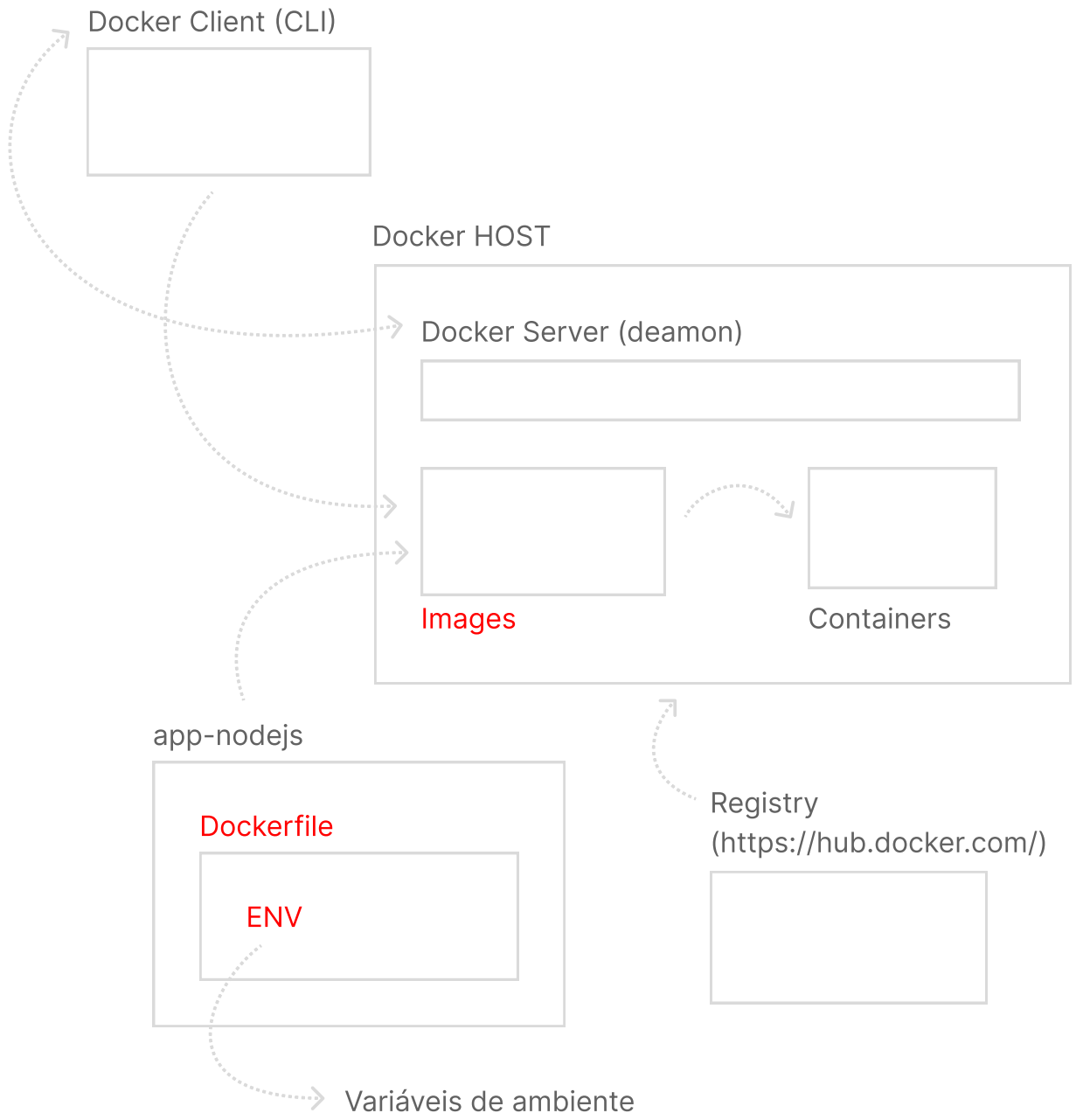
► Images

Dockerfile - LABEL



► Images

Dockerfile - ENV



Dockerfile - ENV reutilizando variáveis de ambiente no Dockerfile

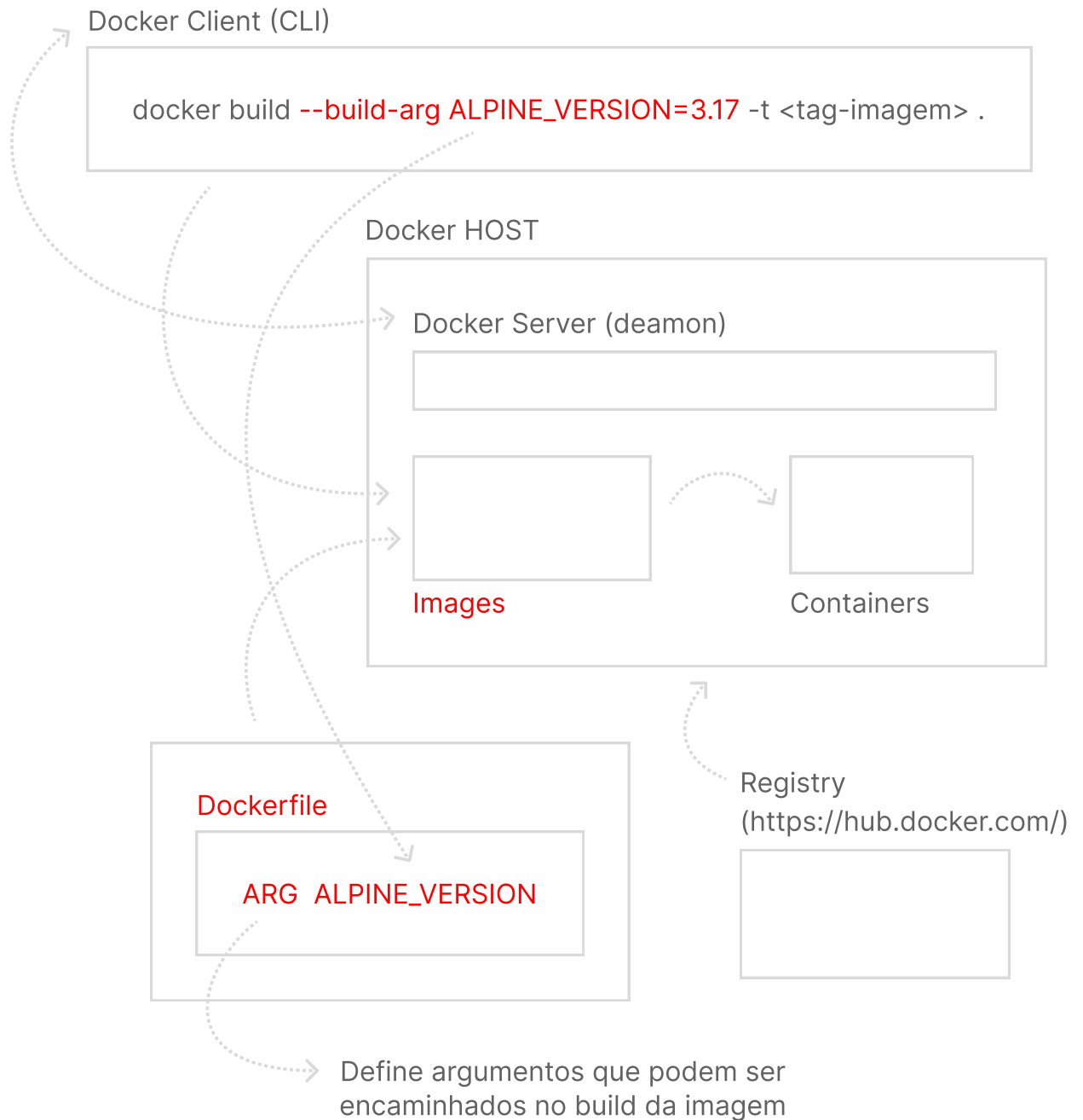
Dockerfile

```
ENV VARIABEL=VALOR
```

```
LABEL ROTULO=${VALOR}
```



Dockerfile - ARG



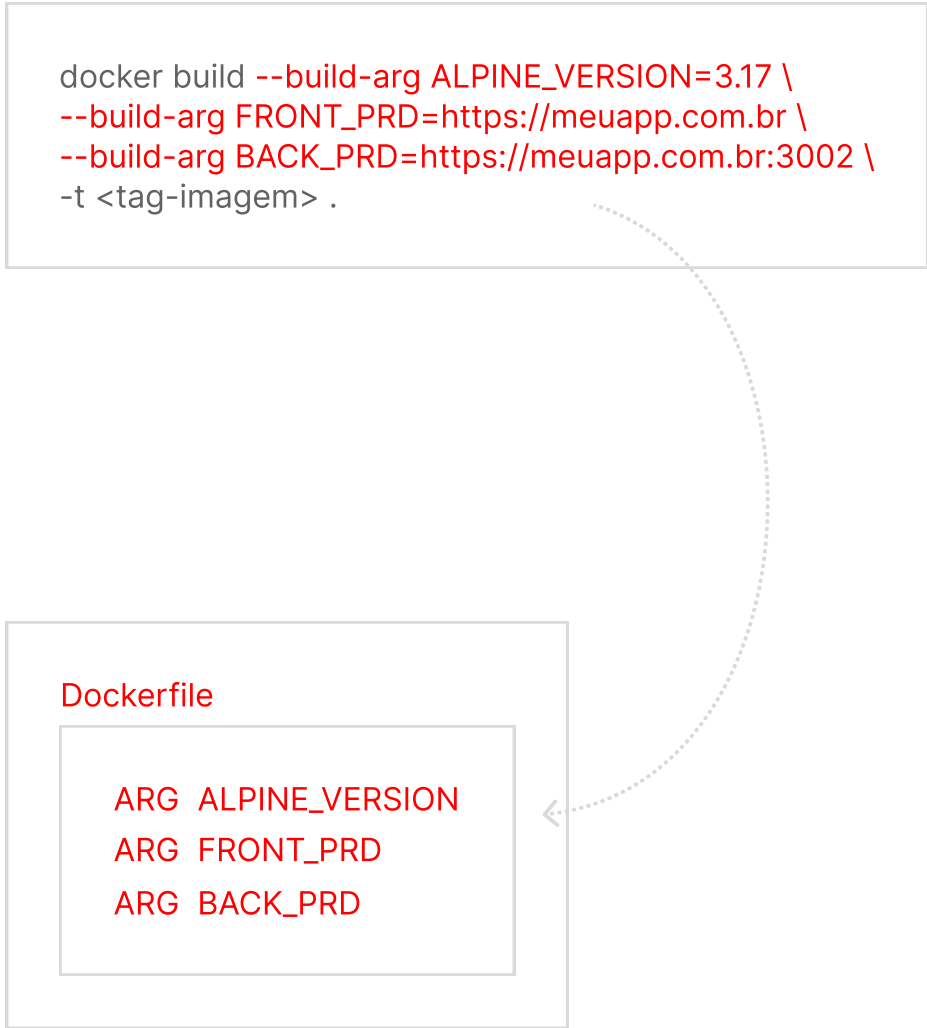
Dockerfile - ARG múltiplos argumentos

Docker Client (CLI)

```
docker build --build-arg ALPINE_VERSION=3.17 \  
--build-arg FRONT_PRD=https://meuapp.com.br \  
--build-arg BACK_PRD=https://meuapp.com.br:3002 \  
-t <tag-imagem> .
```

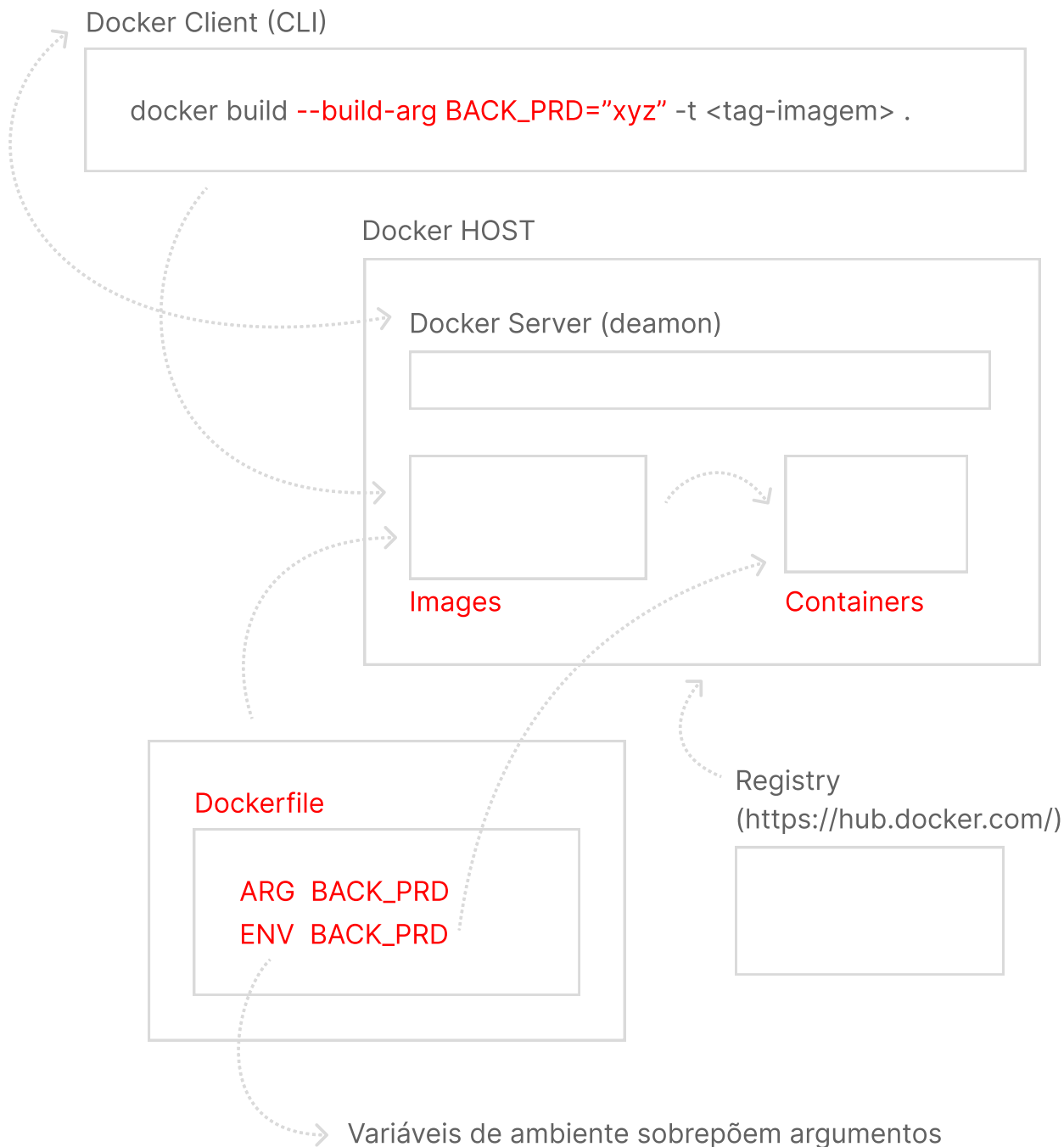
Dockerfile

```
ARG ALPINE_VERSION  
ARG FRONT_PRD  
ARG BACK_PRD
```



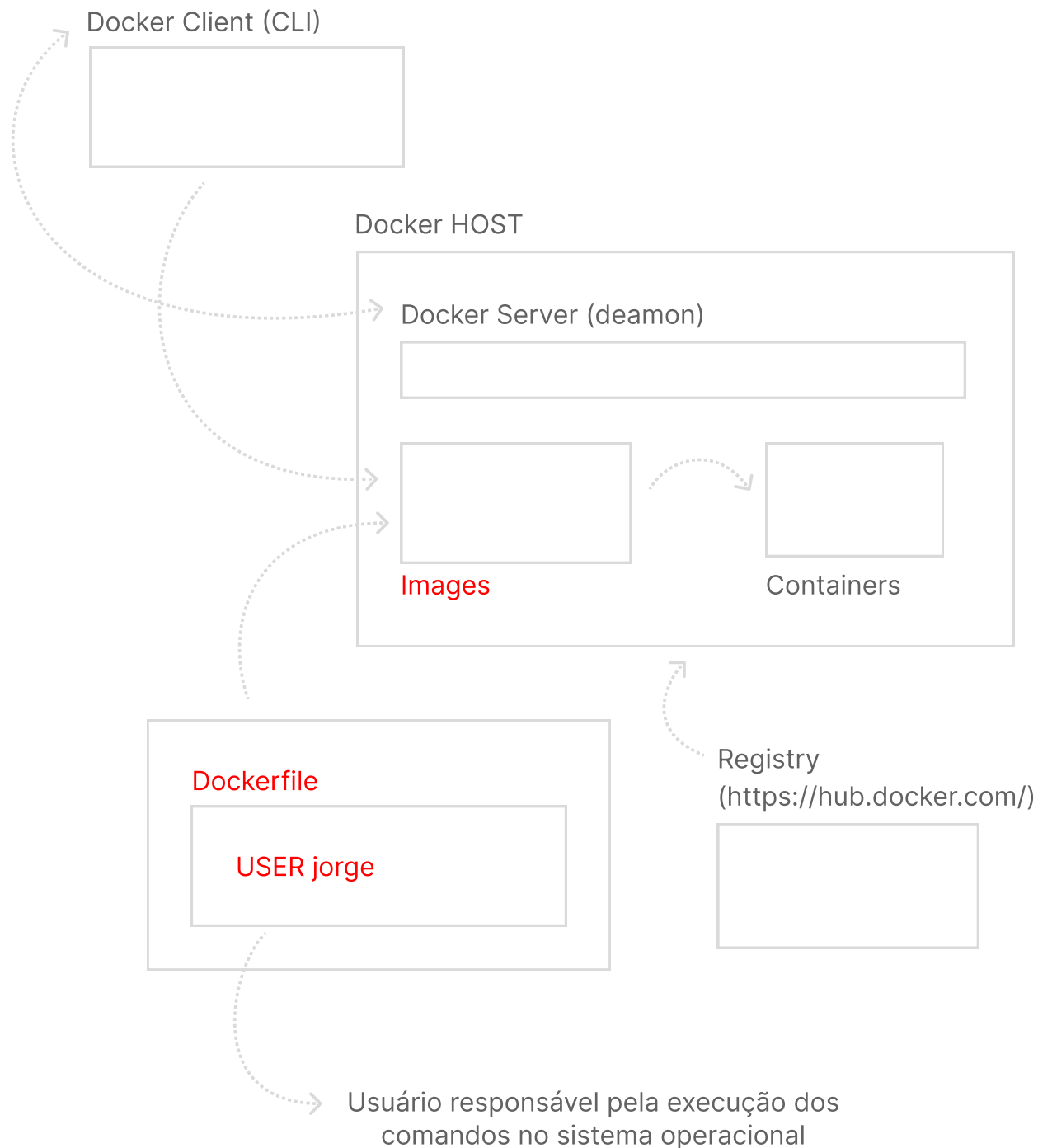
► Images

Dockerfile - ARG vs ENV



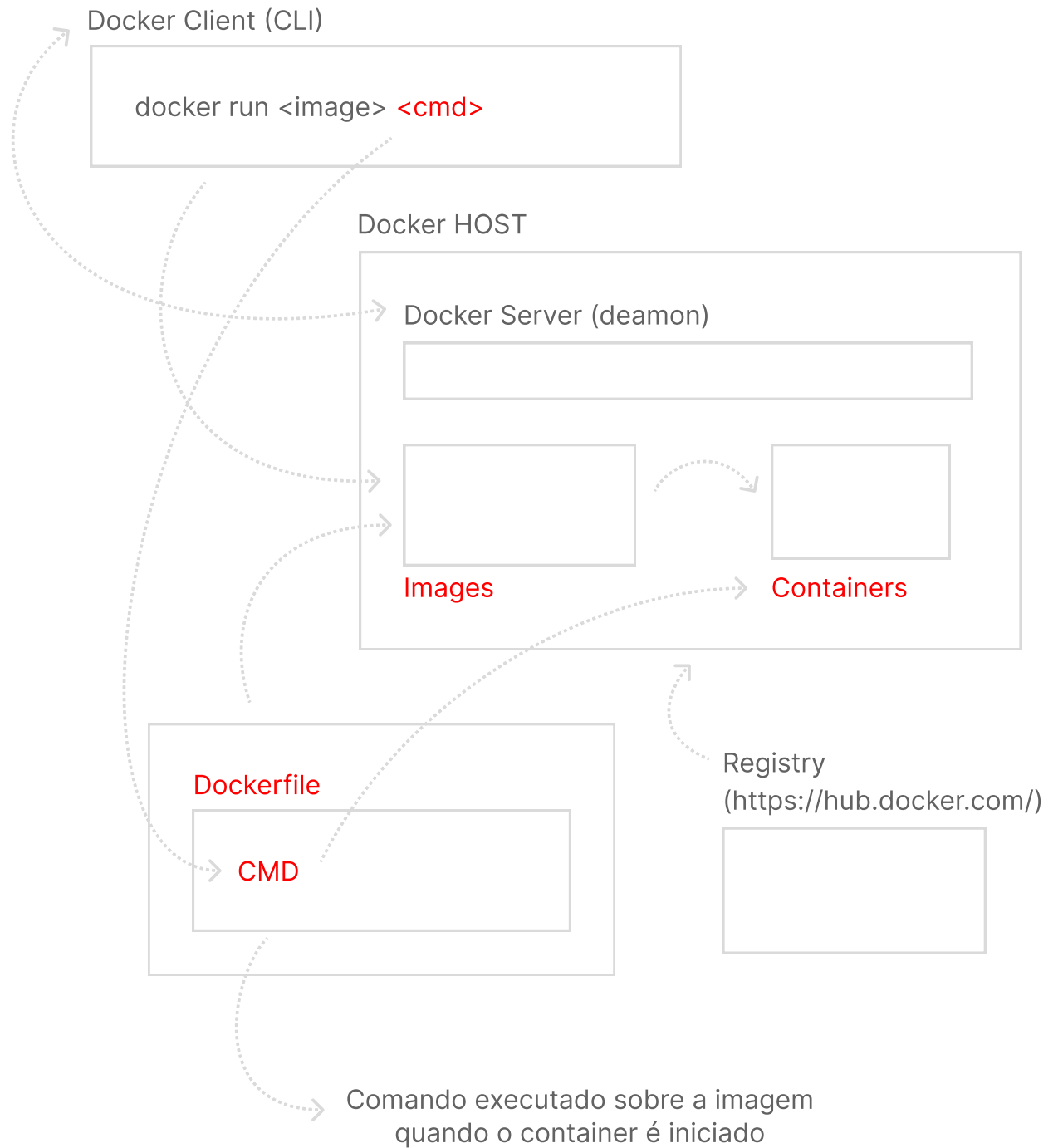
► Images

Dockerfile - USER

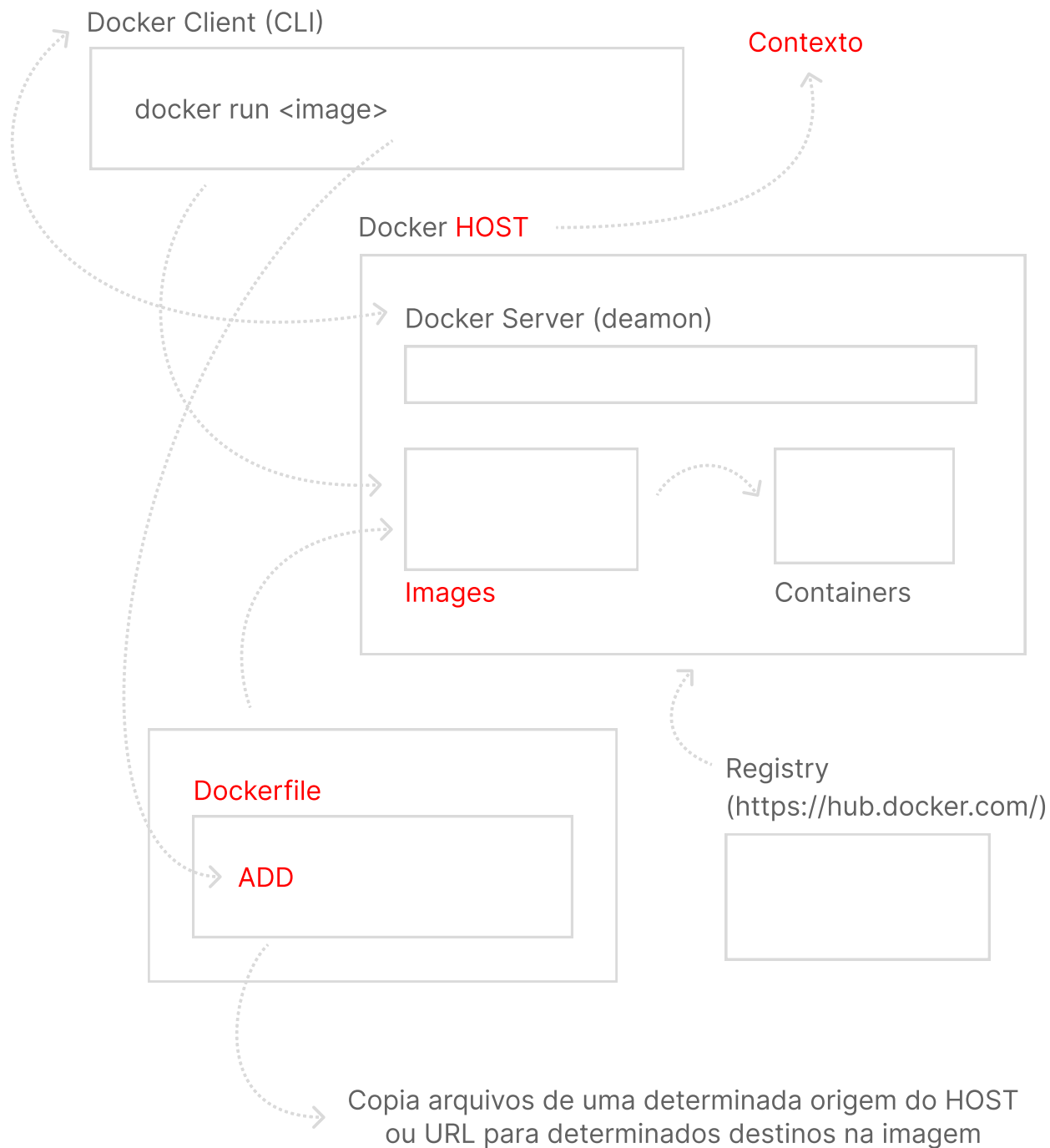


► Images

Dockerfile - CMD

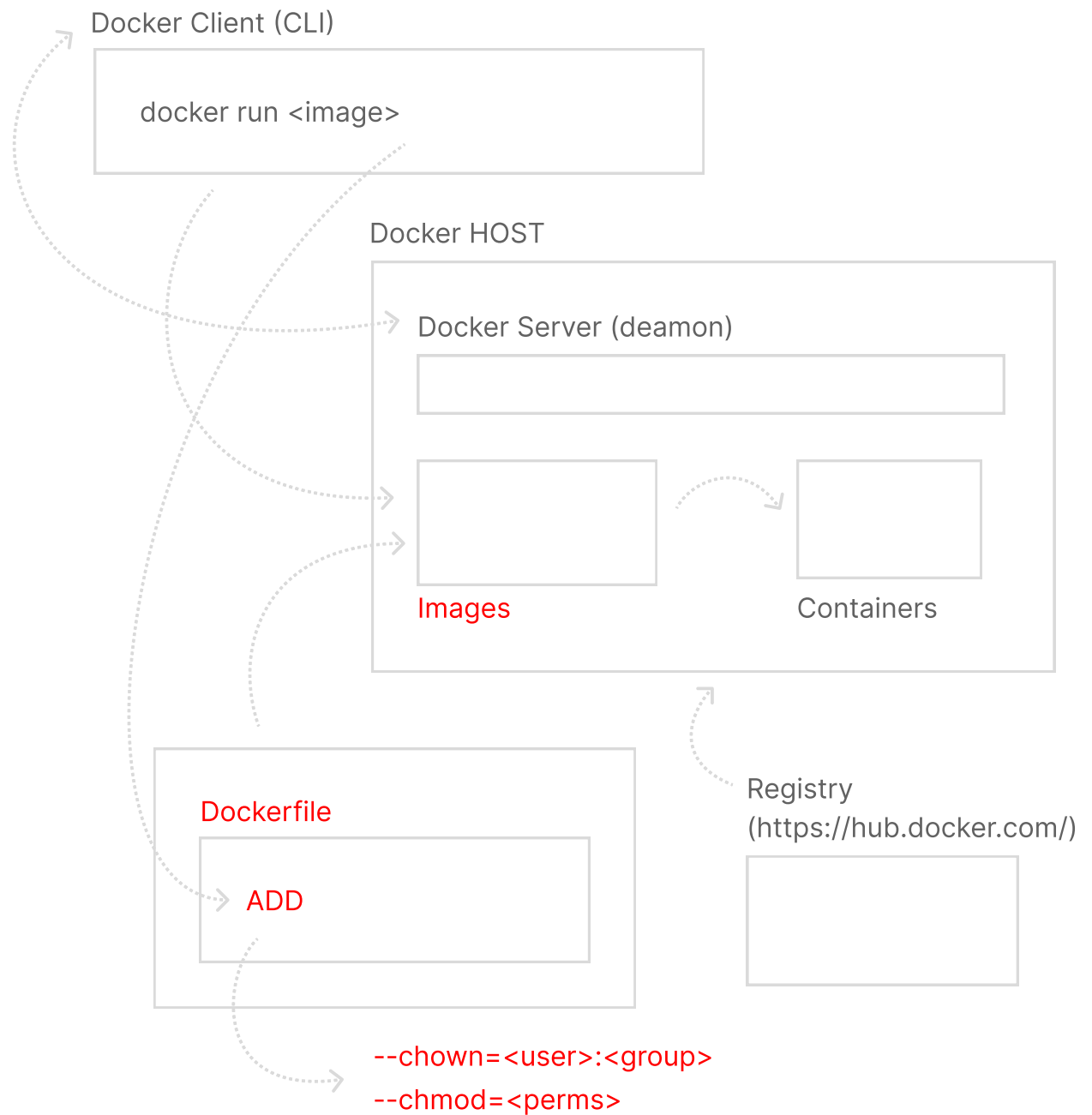


Dockerfile - ADD



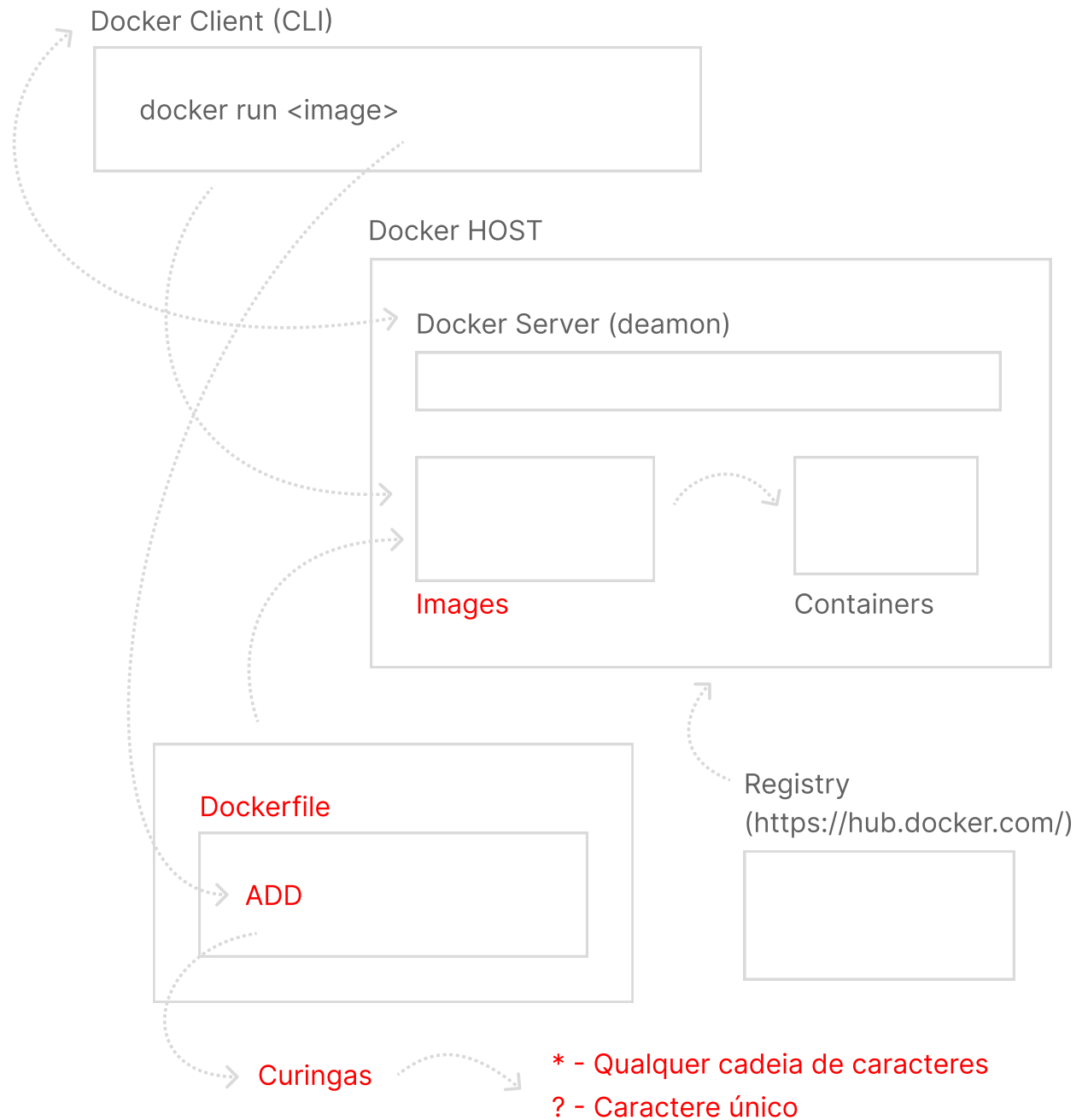
► Images

Dockerfile - ADD ajustando permissões



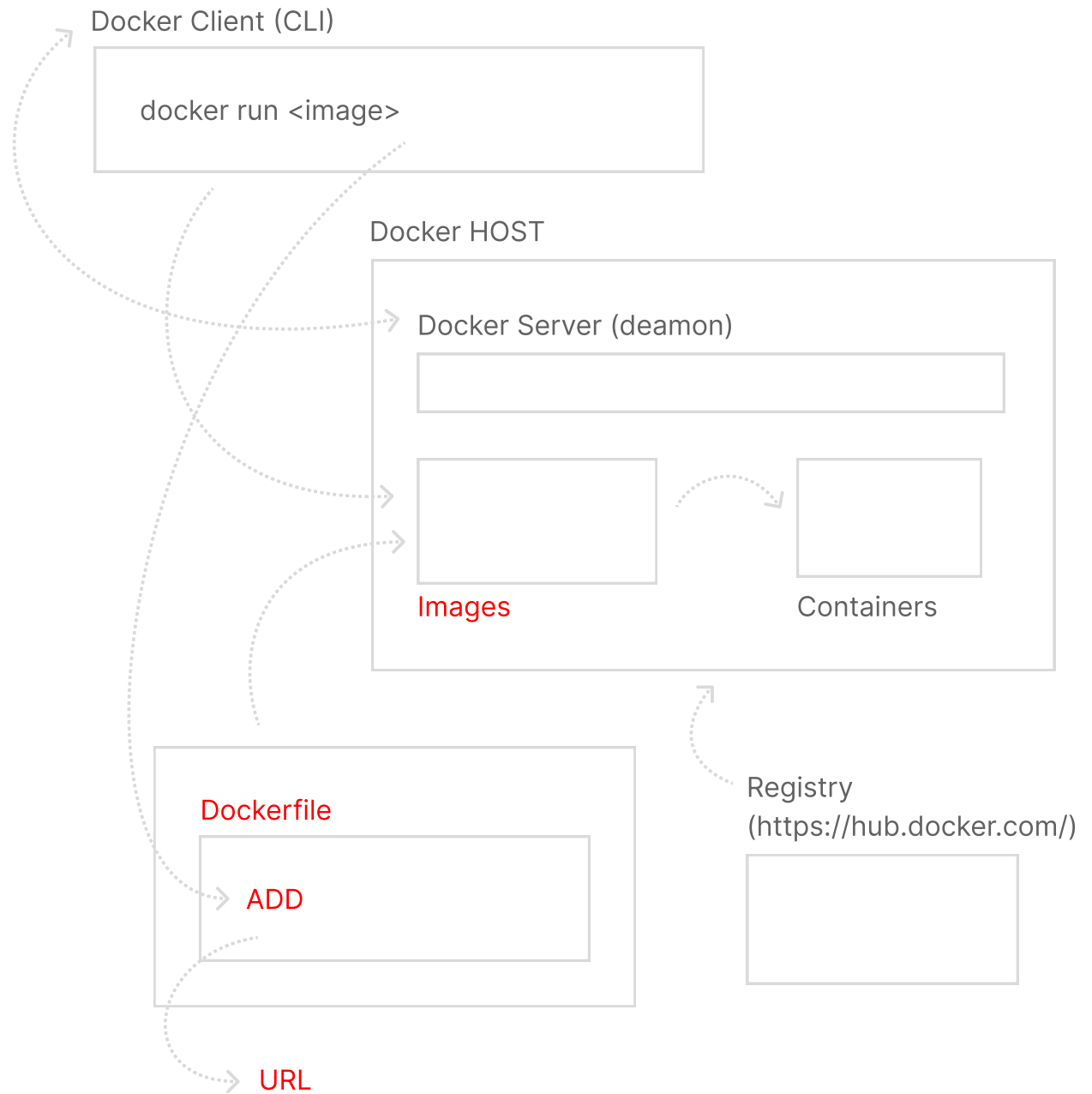
► Images

Dockerfile - ADD curingas e arquivos zipados



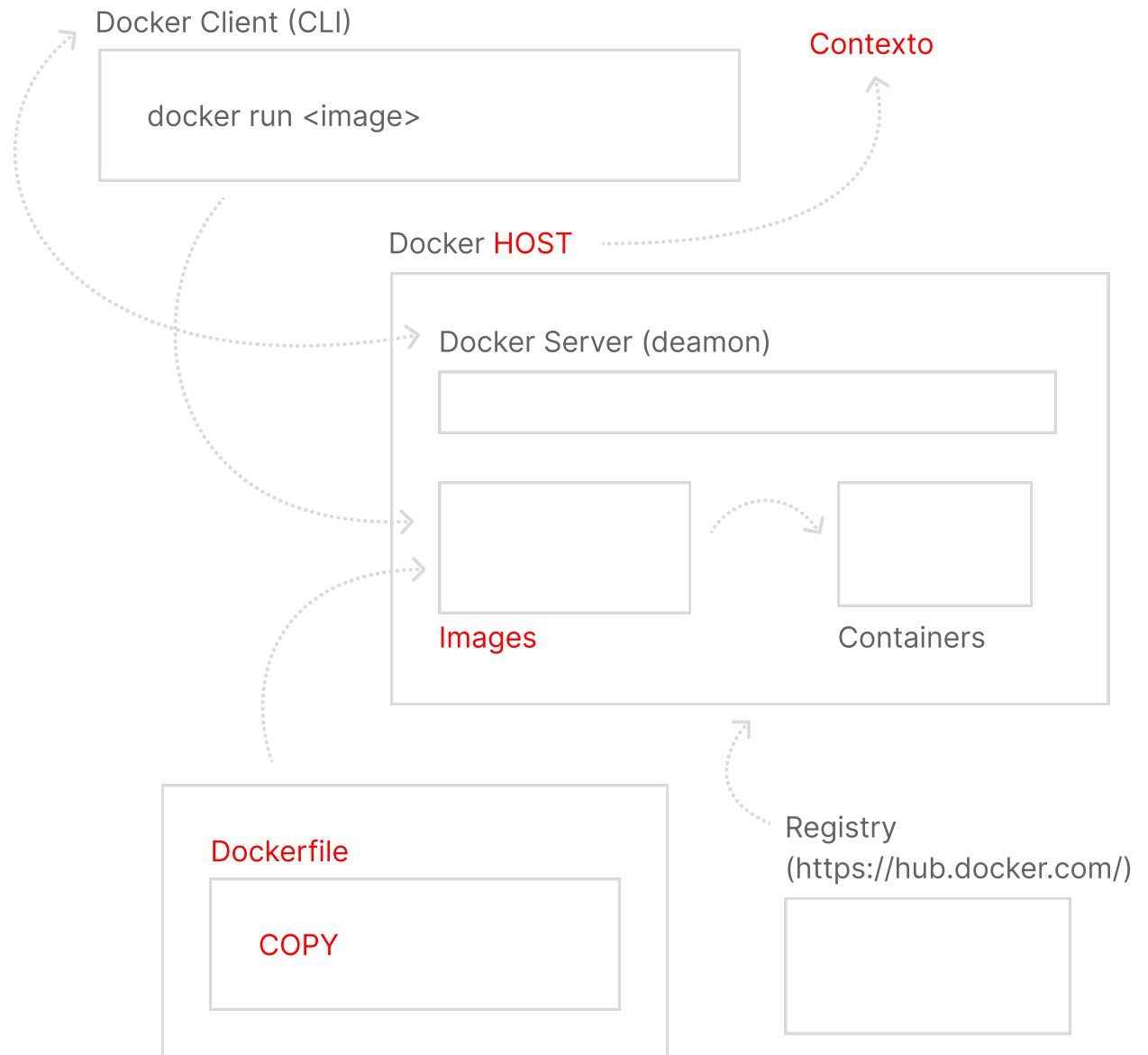
► Images

Dockerfile - ADD copiando arquivos via URL

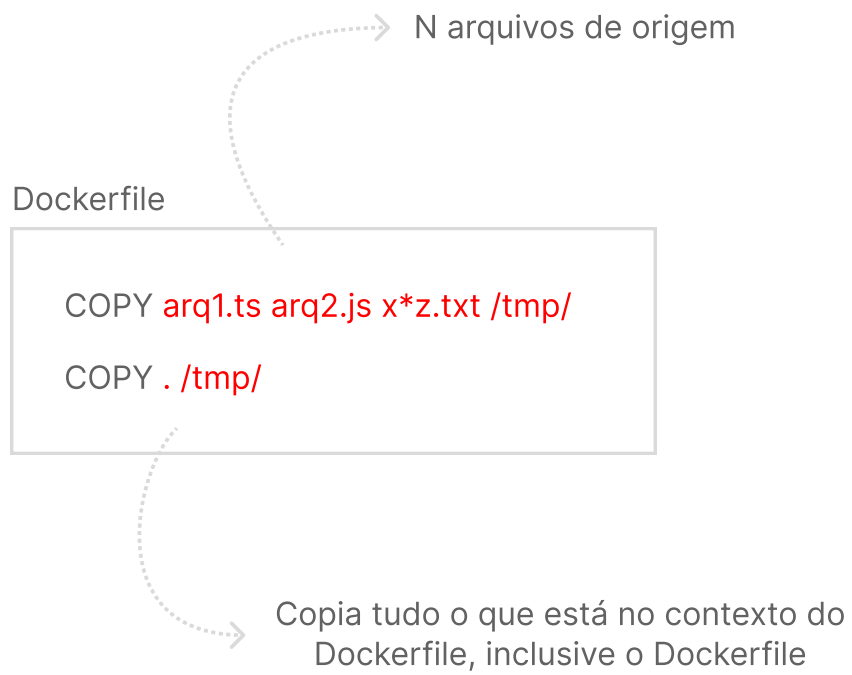


► Images

Dockerfile - COPY

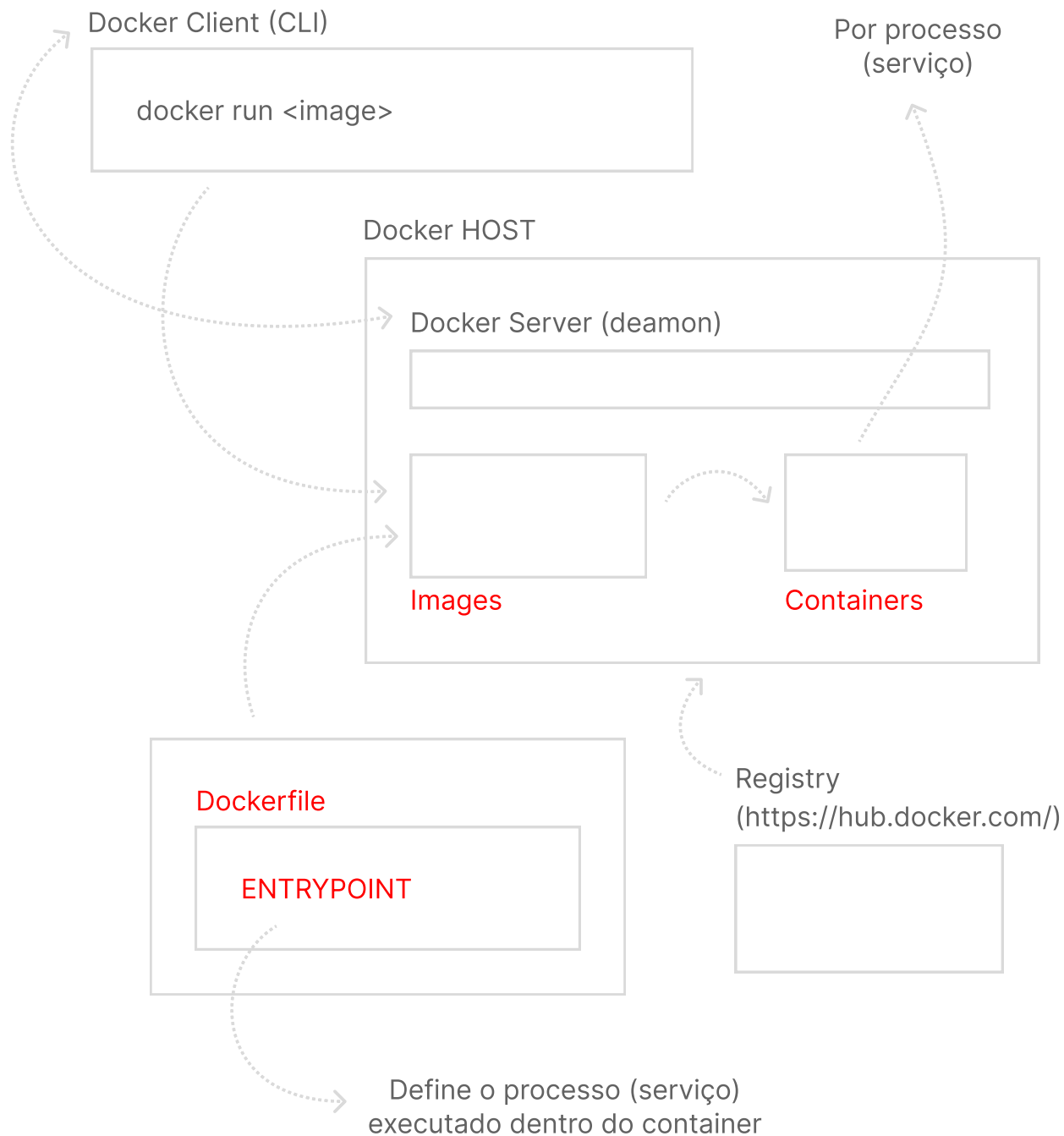


Dockerfile - COPY múltiplos arquivos



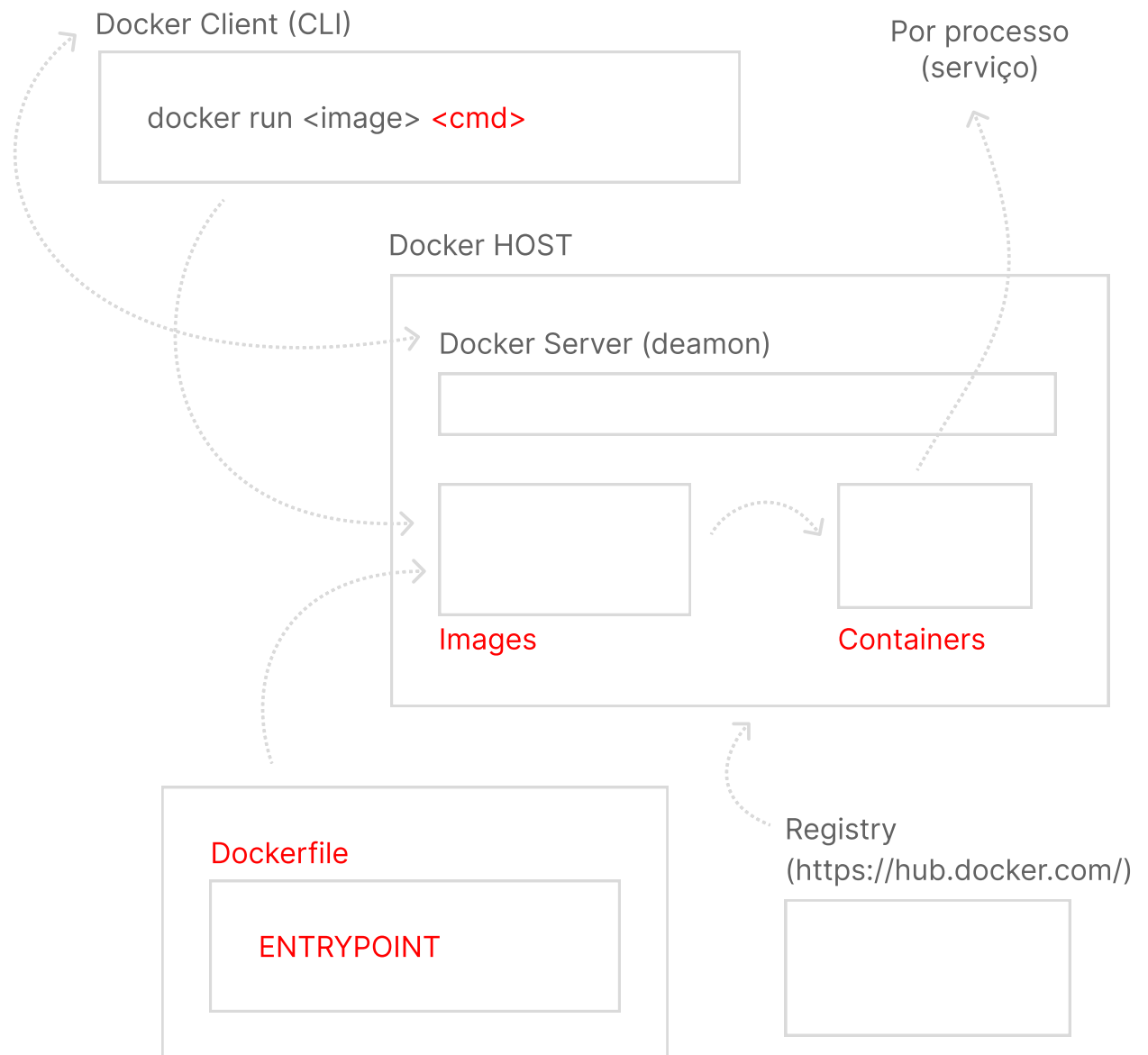
► Images

Dockerfile - ENTRYPOINT



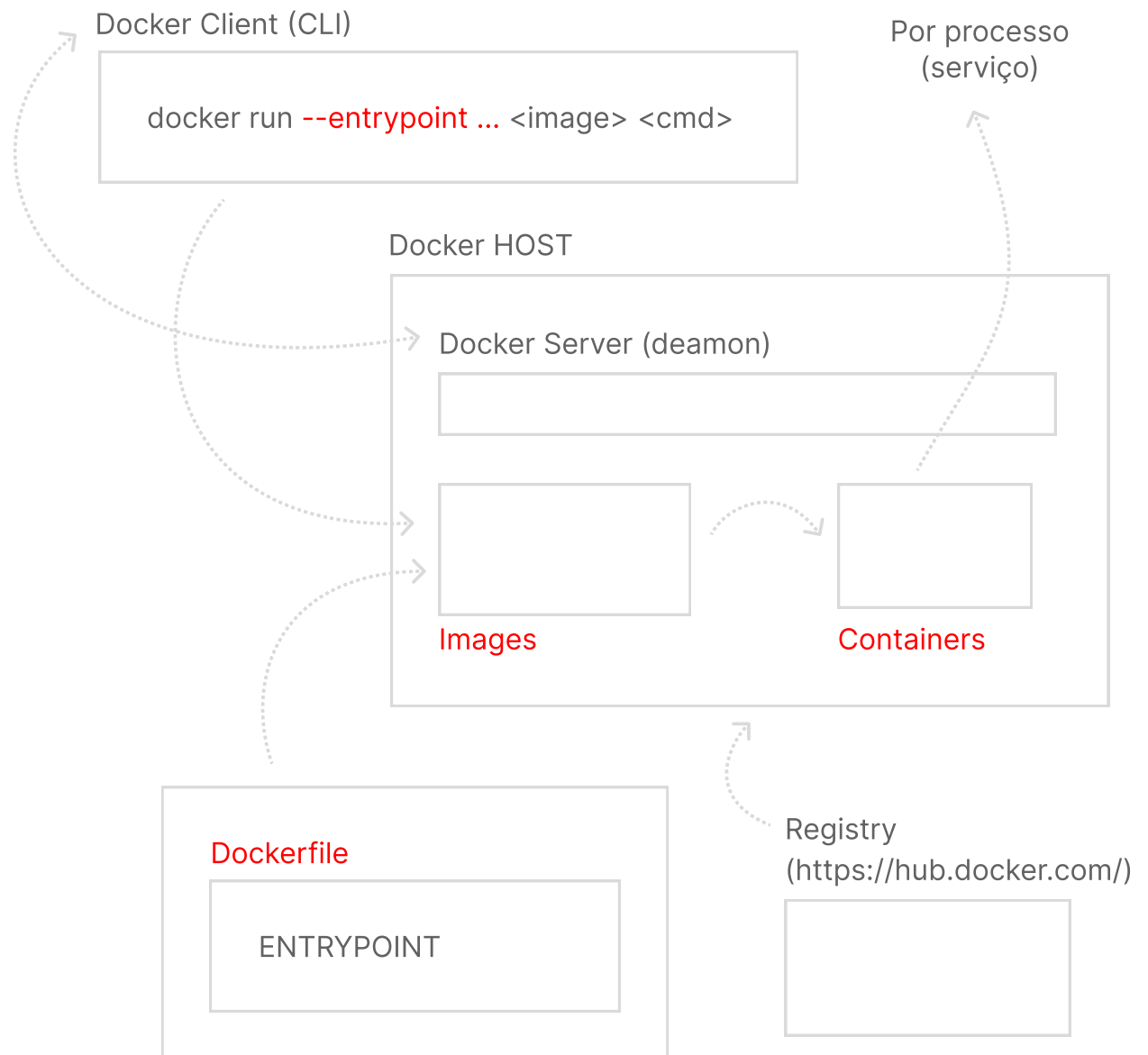
► Images

Dockerfile - ENTRYPOINT combinado com CMD



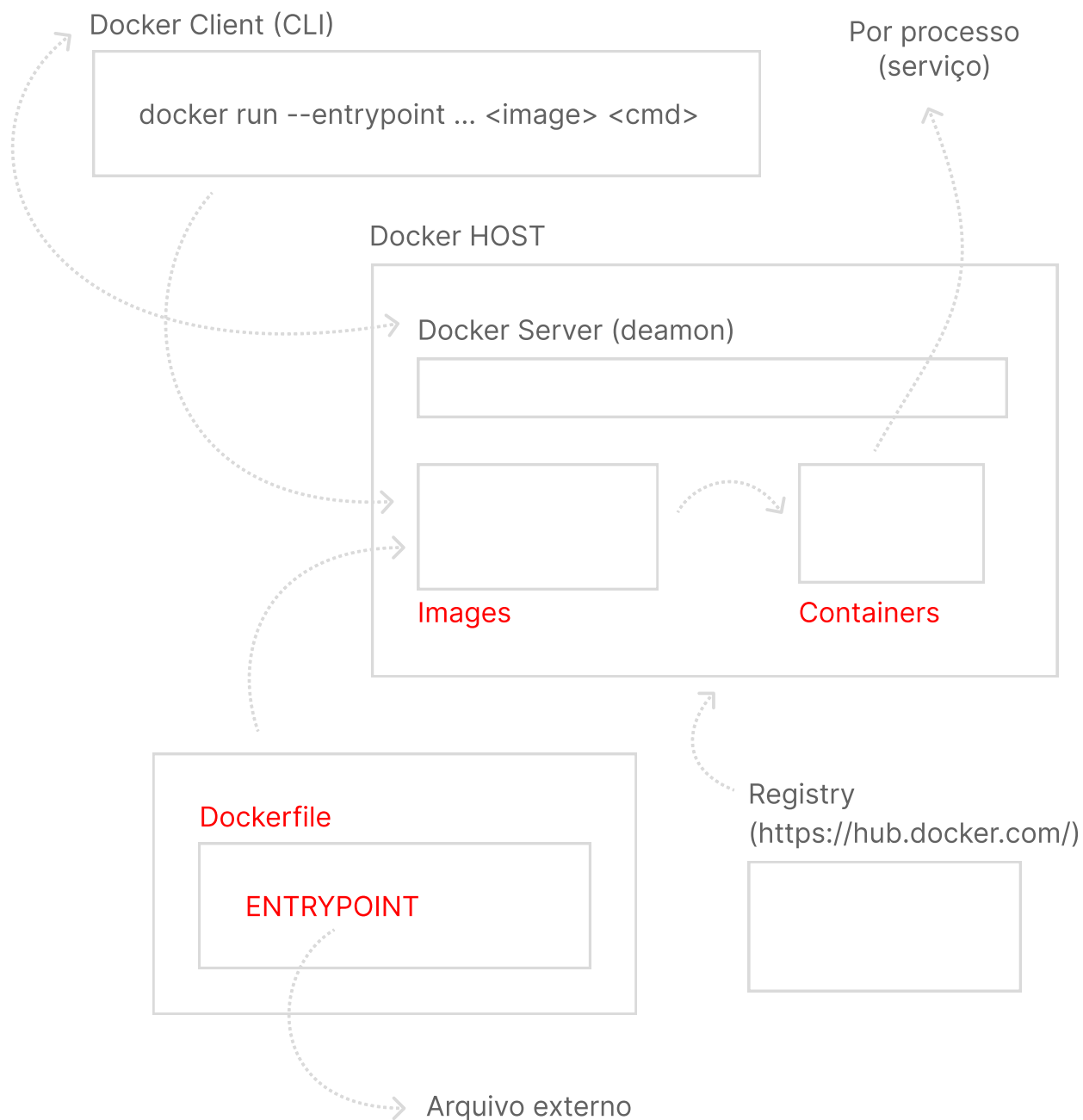
► Images

Dockerfile - ENTRYPOINT via terminal



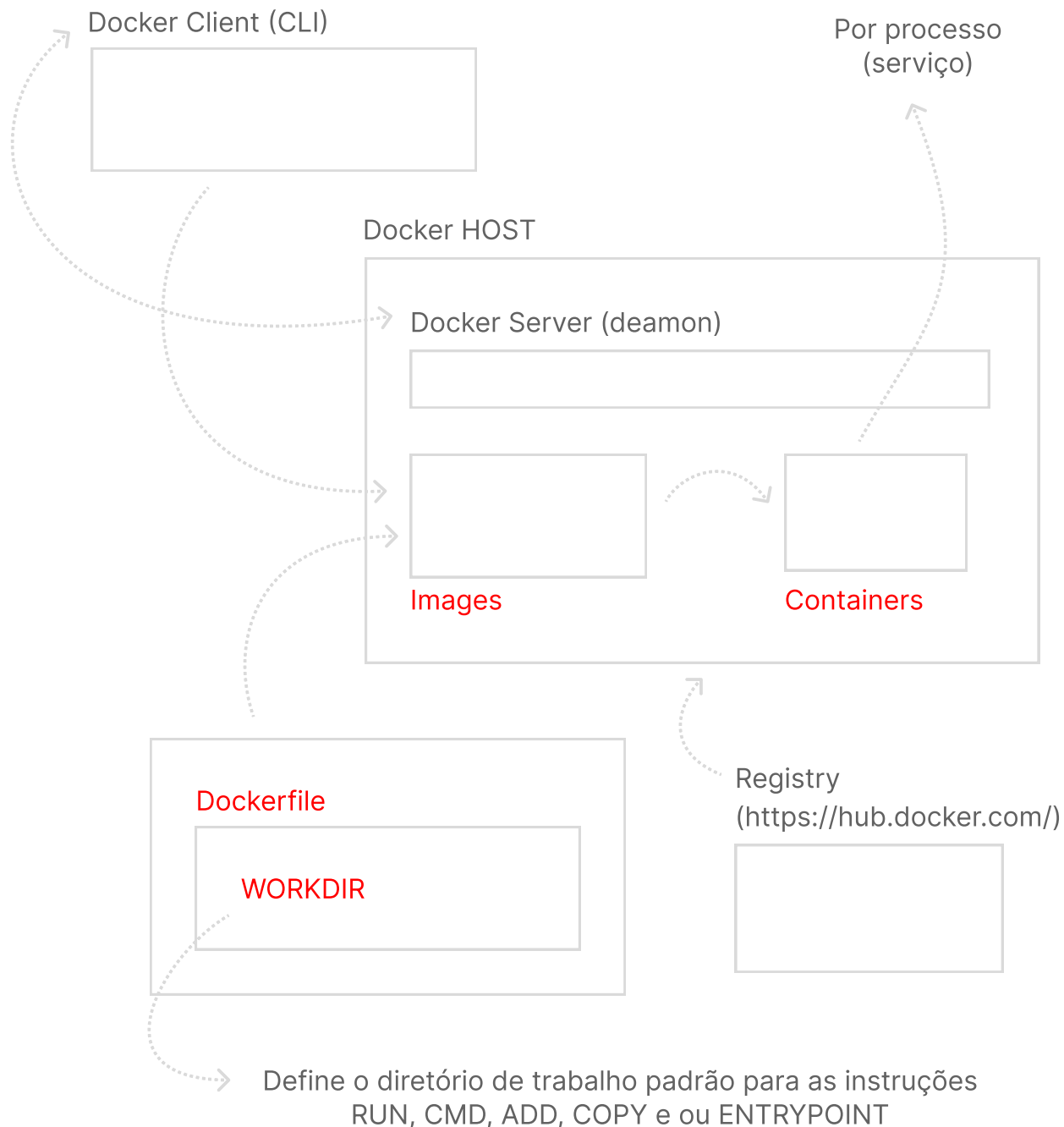
► Images

Dockerfile - ENTRYPOINT via arquivo externo



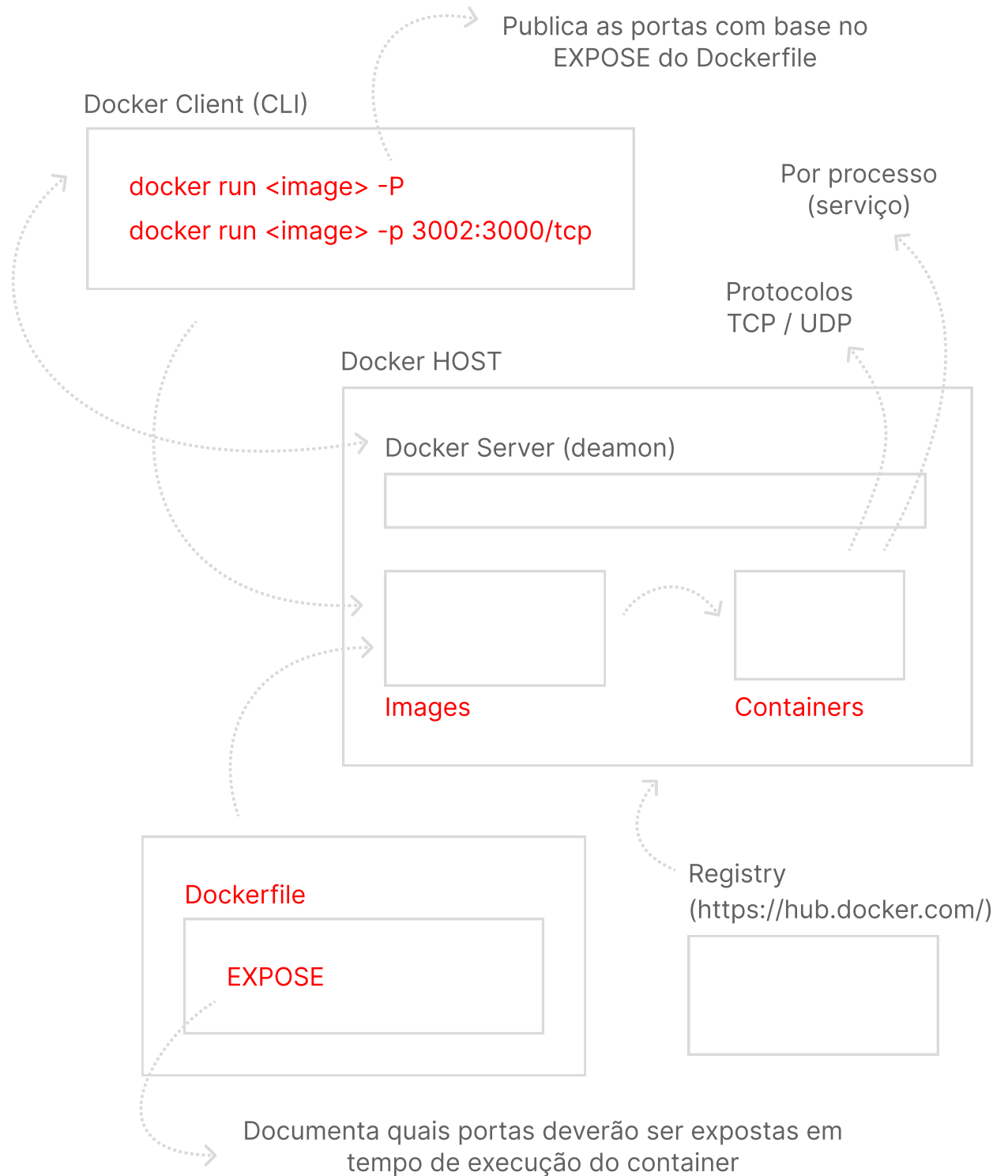
► Images

Dockerfile - WORKDIR

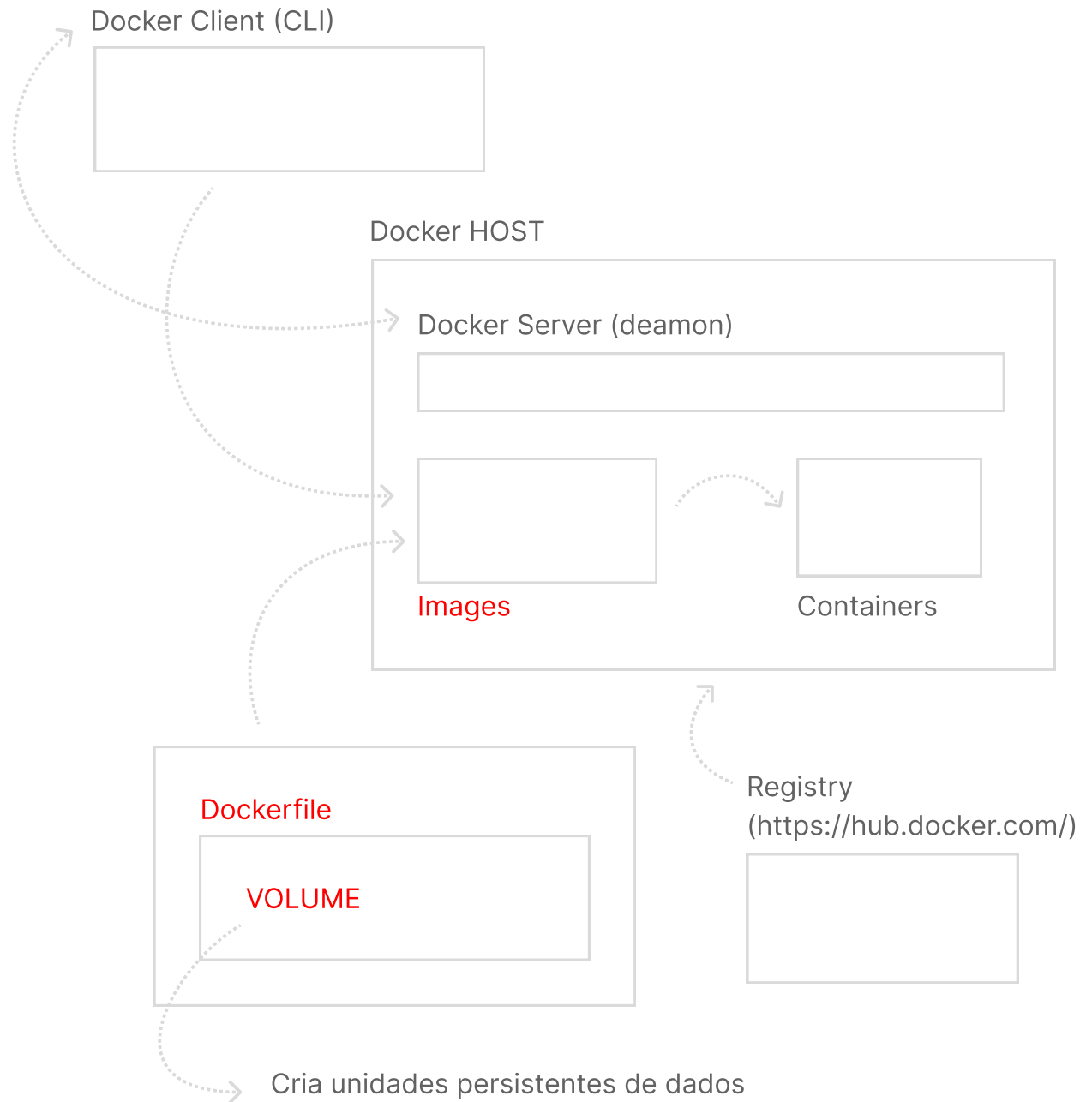


► Images

Dockerfile - EXPOSE

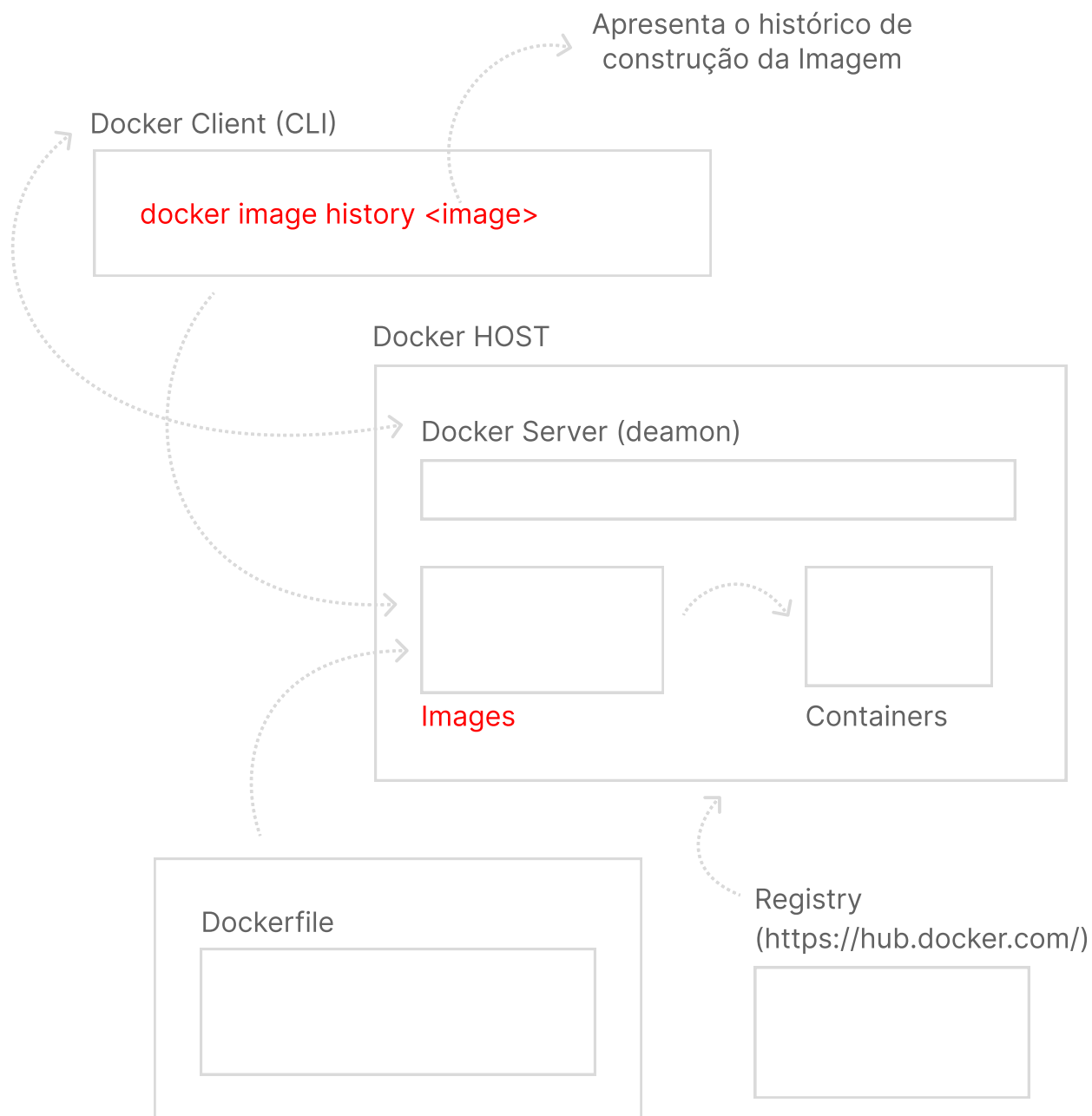


Dockerfile - VOLUME

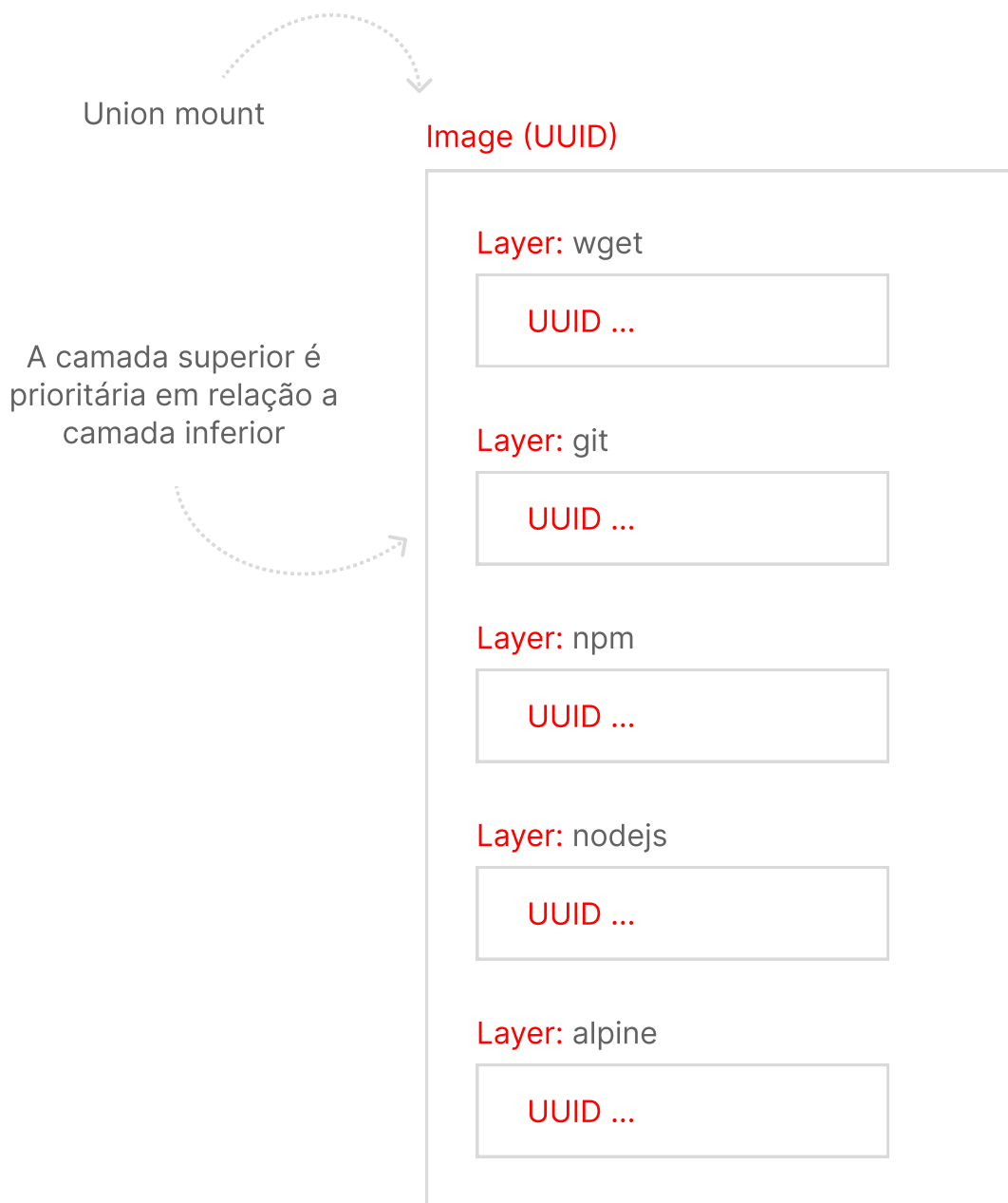


► Images

Consultando o histórico de construção da imagem



Camadas e cache de camadas



Dockerfile e as instruções que geram camadas



Afeta a estrutura de arquivos da imagem

Geram camadas

FROM

RUN

ADD

COPY

WORKDIR

VOLUME

Não geram camadas

LABEL

ENV

ARG

USER

CMD

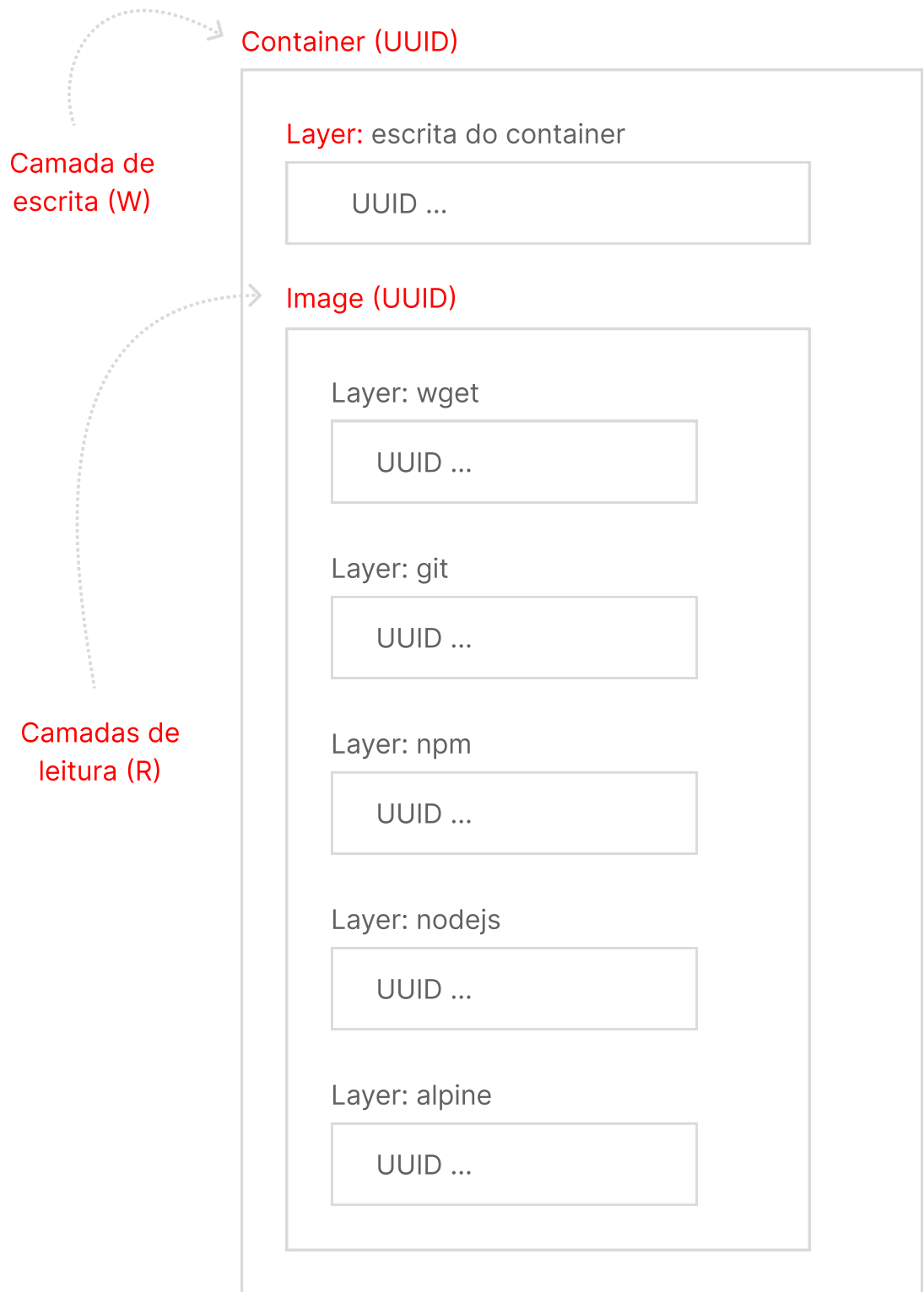
ENTRYPOINT

EXPOSE

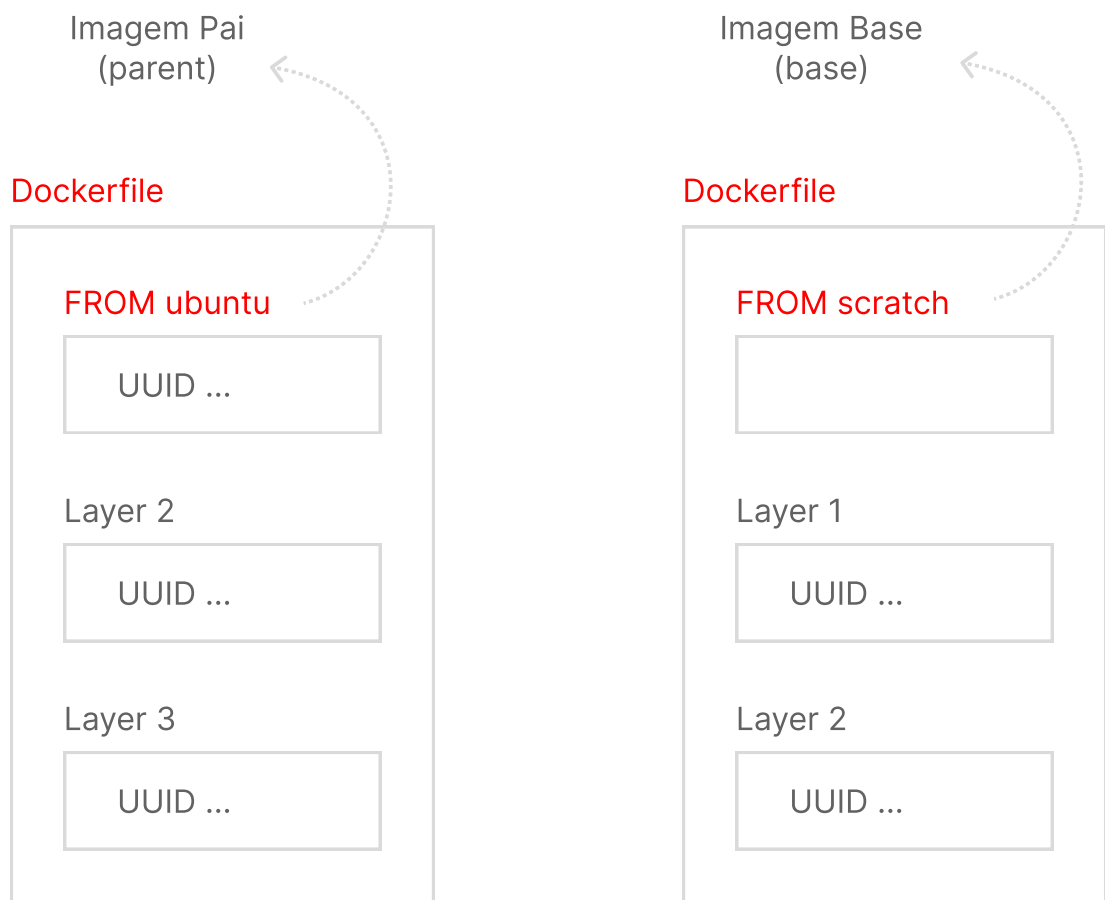
Dockerfile e a herança de instruções entre camadas



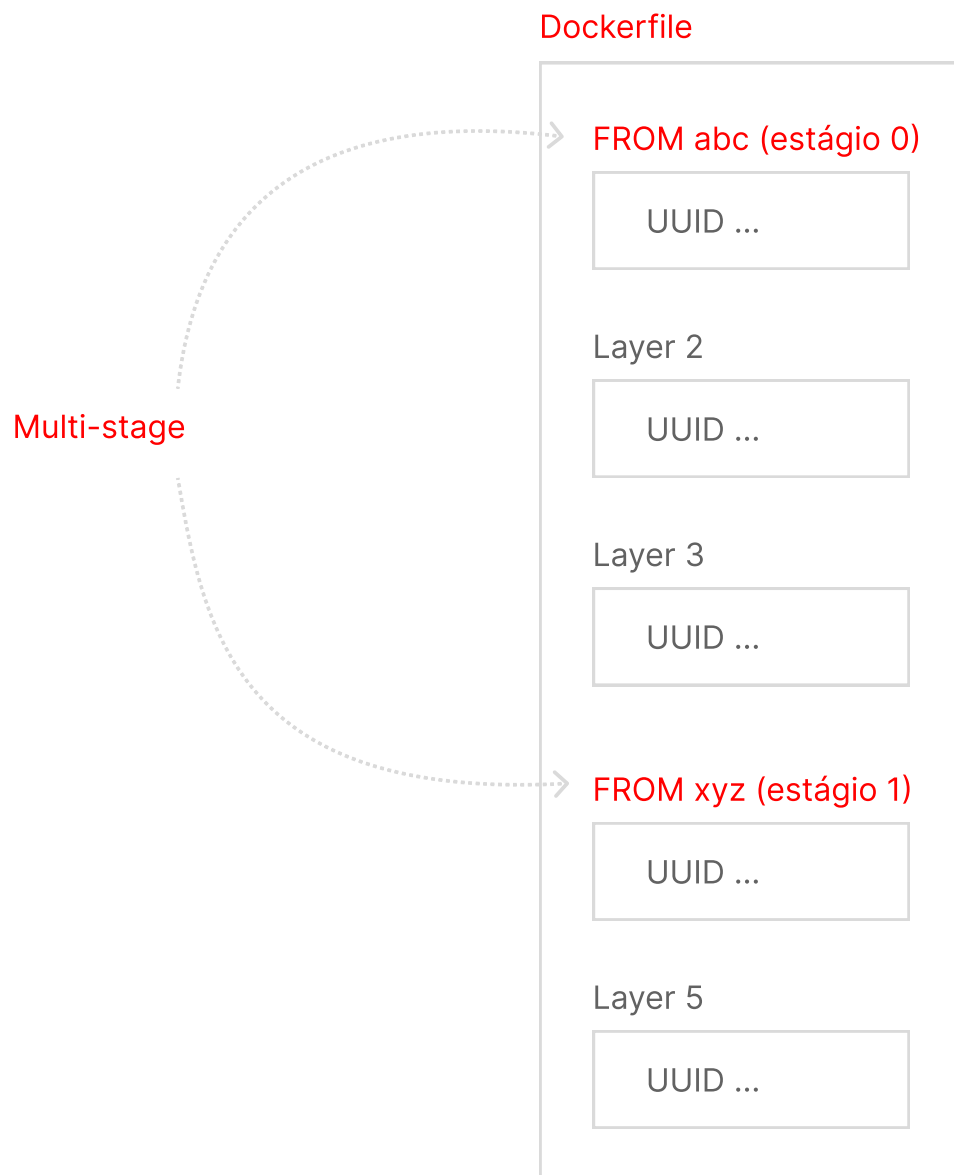
Camadas de leitura da Imagem e escrita do Container



Criando Imagens com FROM scratch parte 1

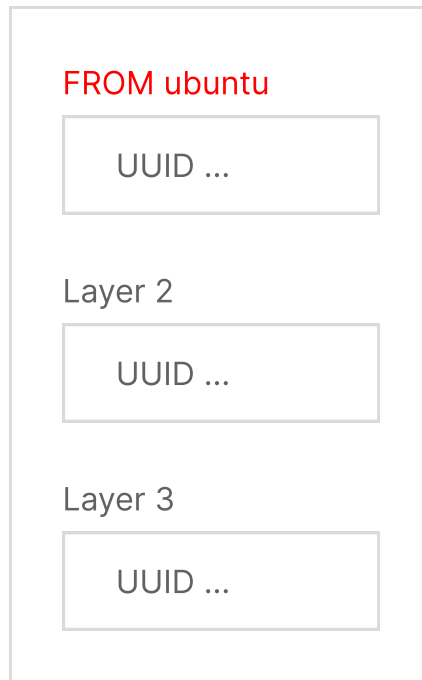


Multi-stage Build - Introdução

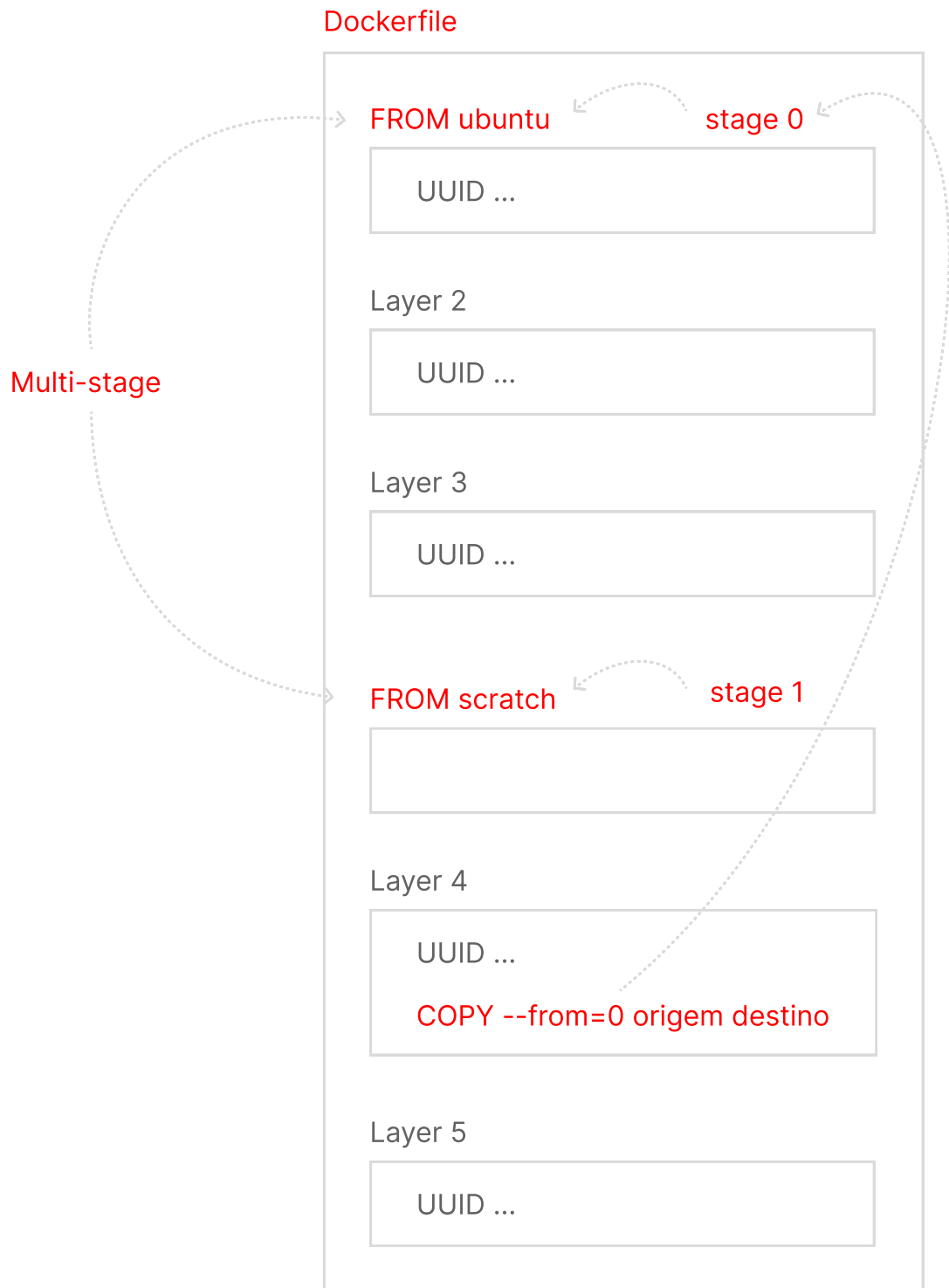


Multi-stage Build - Criando uma Imagem Multi-stage parte 1

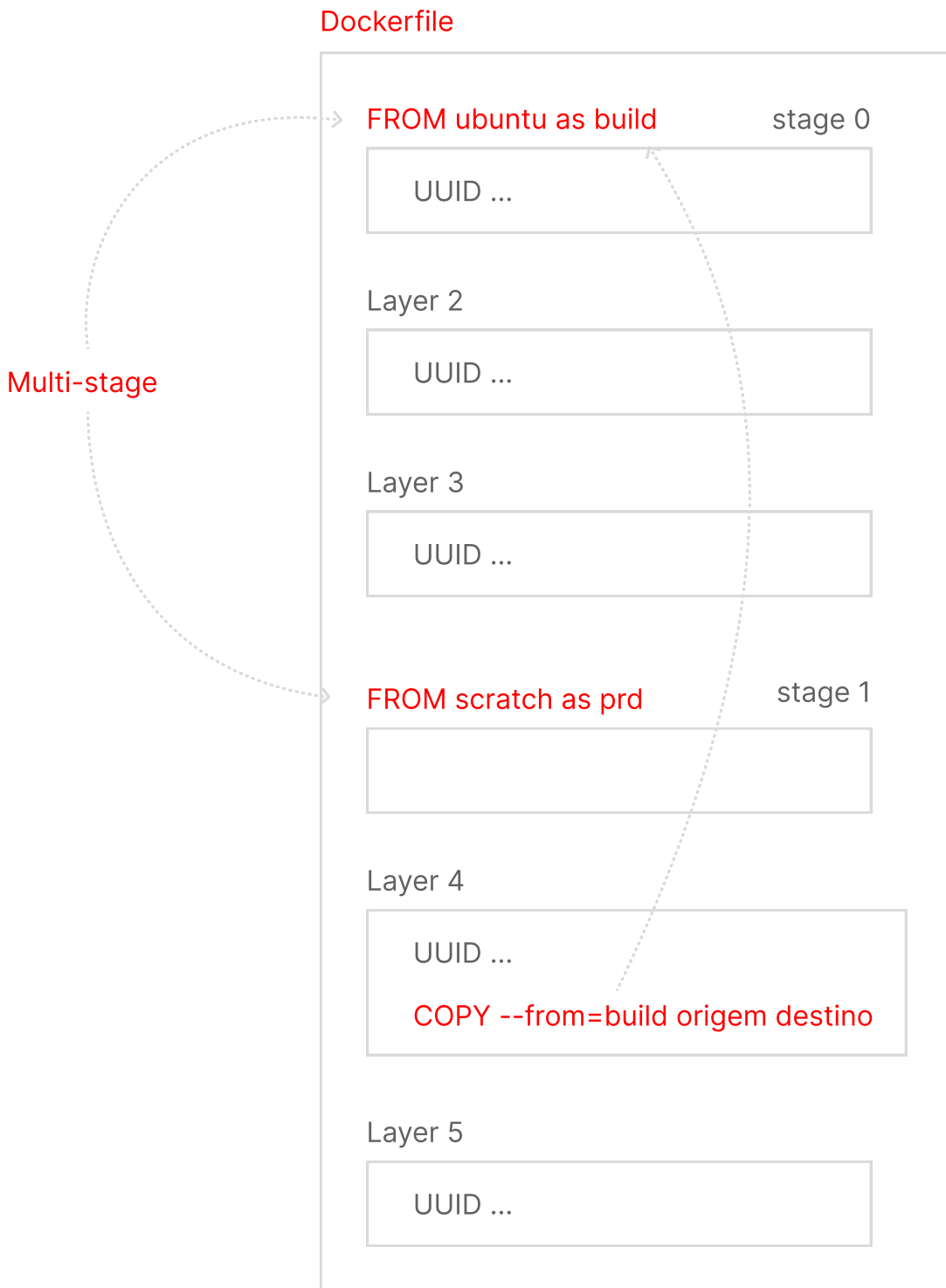
Dockerfile



Multi-stage Build - Criando uma Imagem Multi-stage parte 2

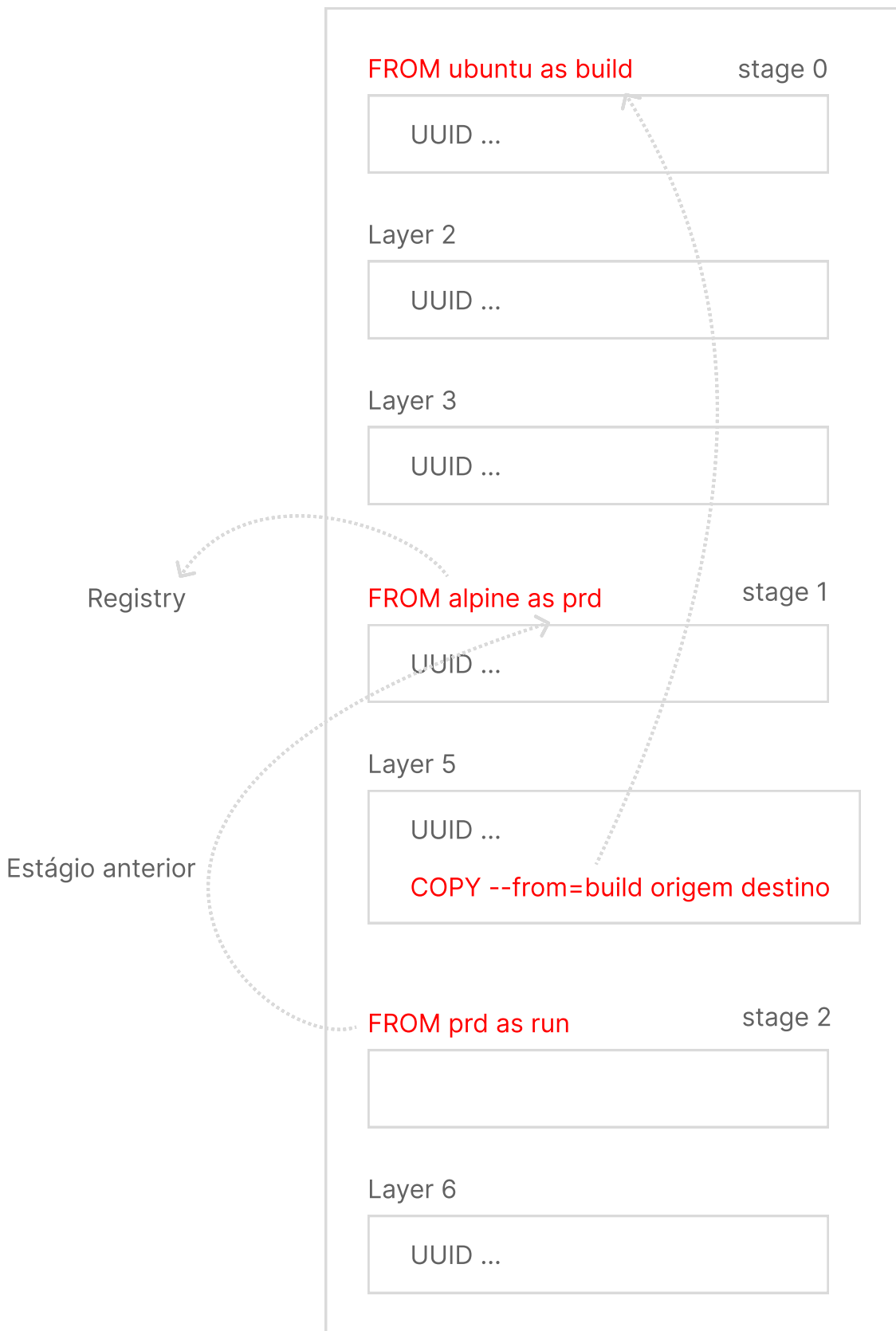


Multi-stage Build - Nomeando estágios



Multi-stage Build - Estágios baseados em estágios anteriores ou imagens

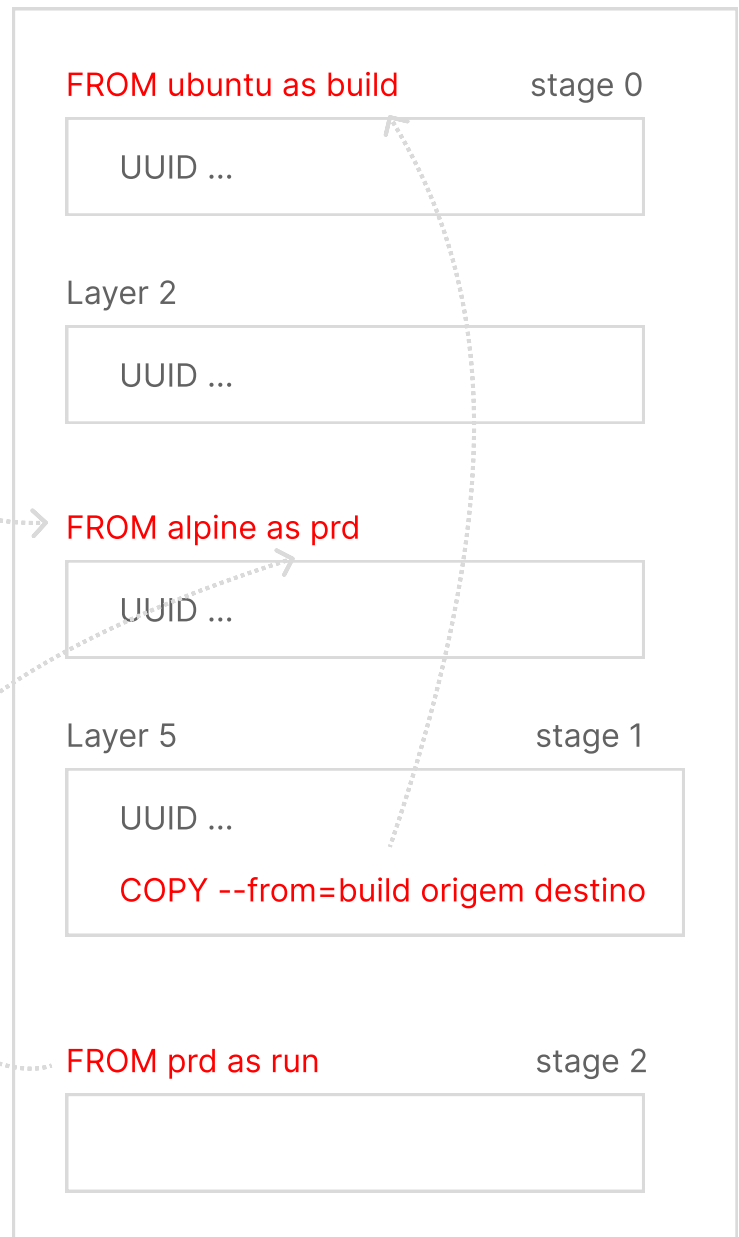
Dockerfile



Multi-stage Build - Build até estágio específico

docker build --target prd -t <imagem> .

Dockerfile



► Images

Exportar, transferir e importar Imagens manualmente

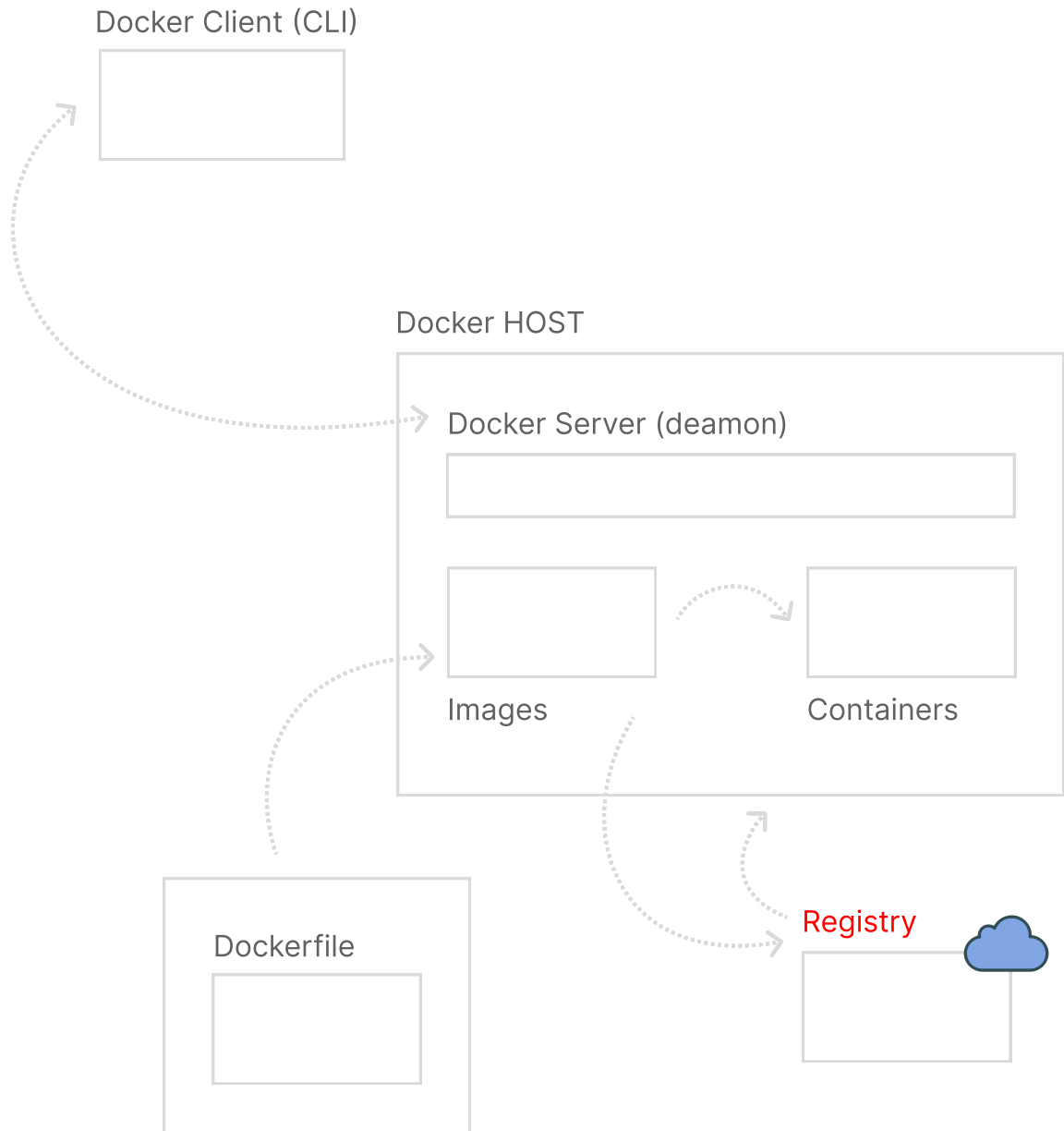
docker image **save -o nome_arquivo_imagem.tar** <imagem>



docker image **load -i nome_arquivo_imagem.tar**

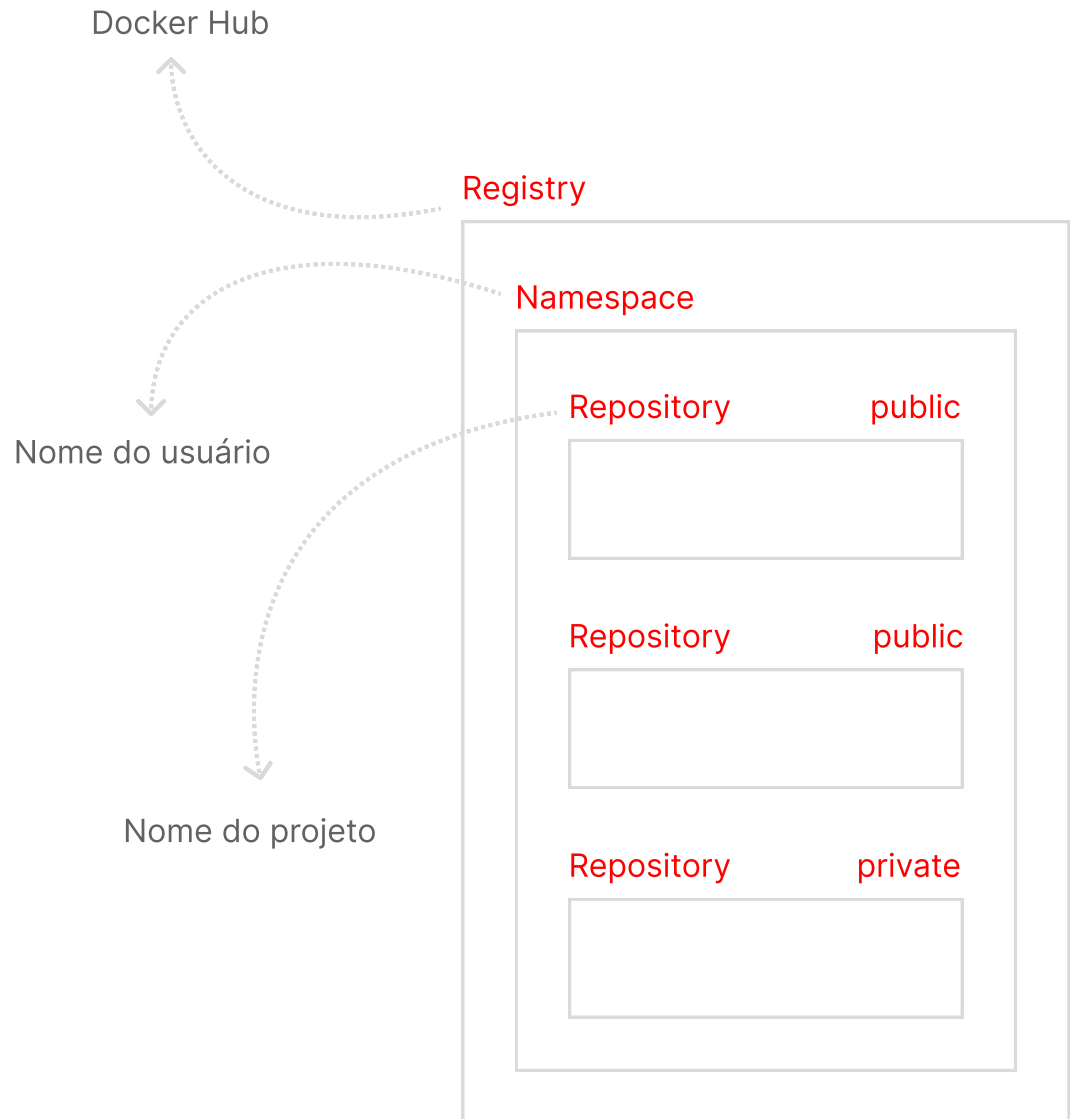
► Registries

Criando uma conta no Docker Hub

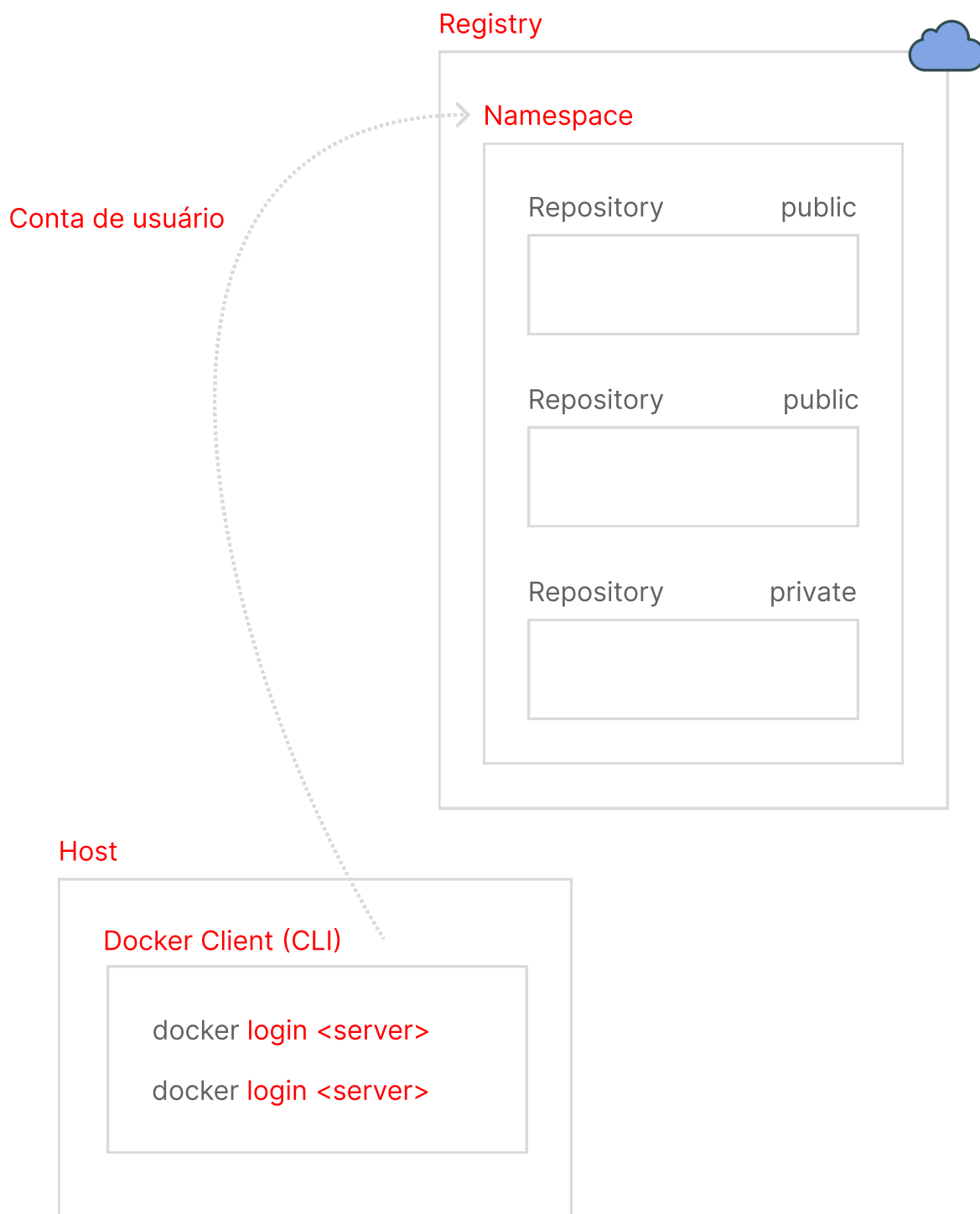


► Registries

Criando um repositório (público e privado)

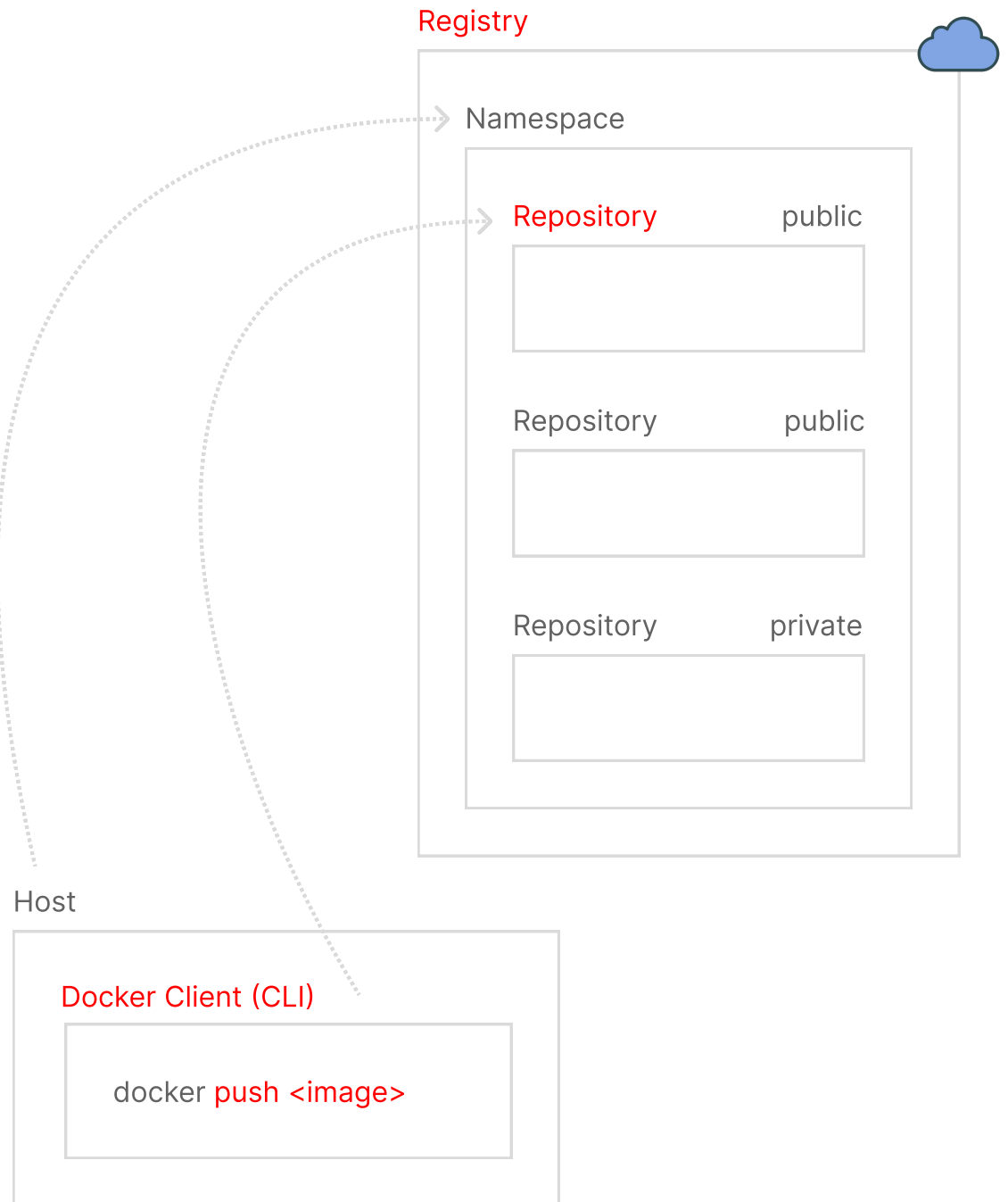


Login e logout no Docker Hub via terminal



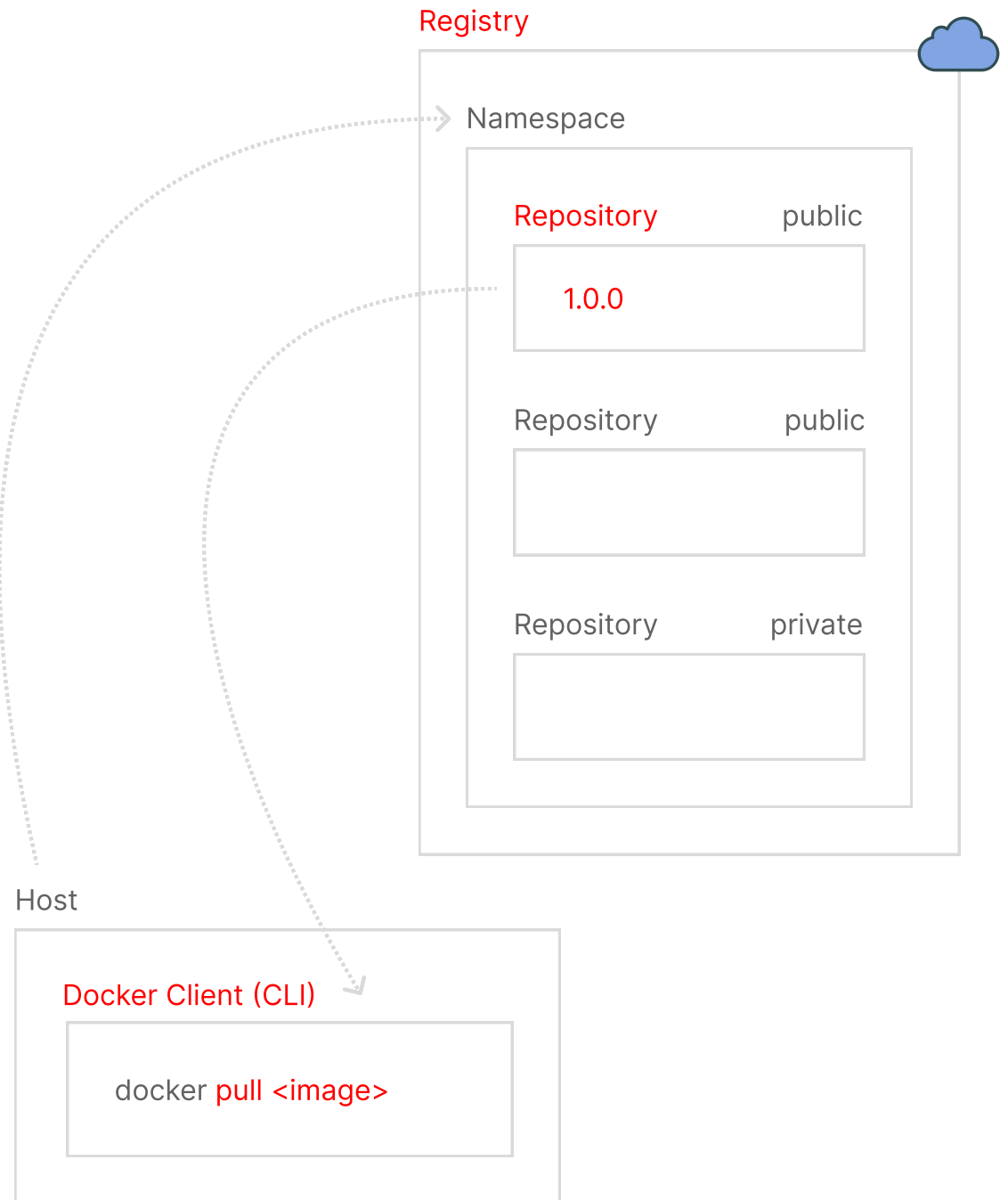
► Registries

Enviando imagem para o registry (push)

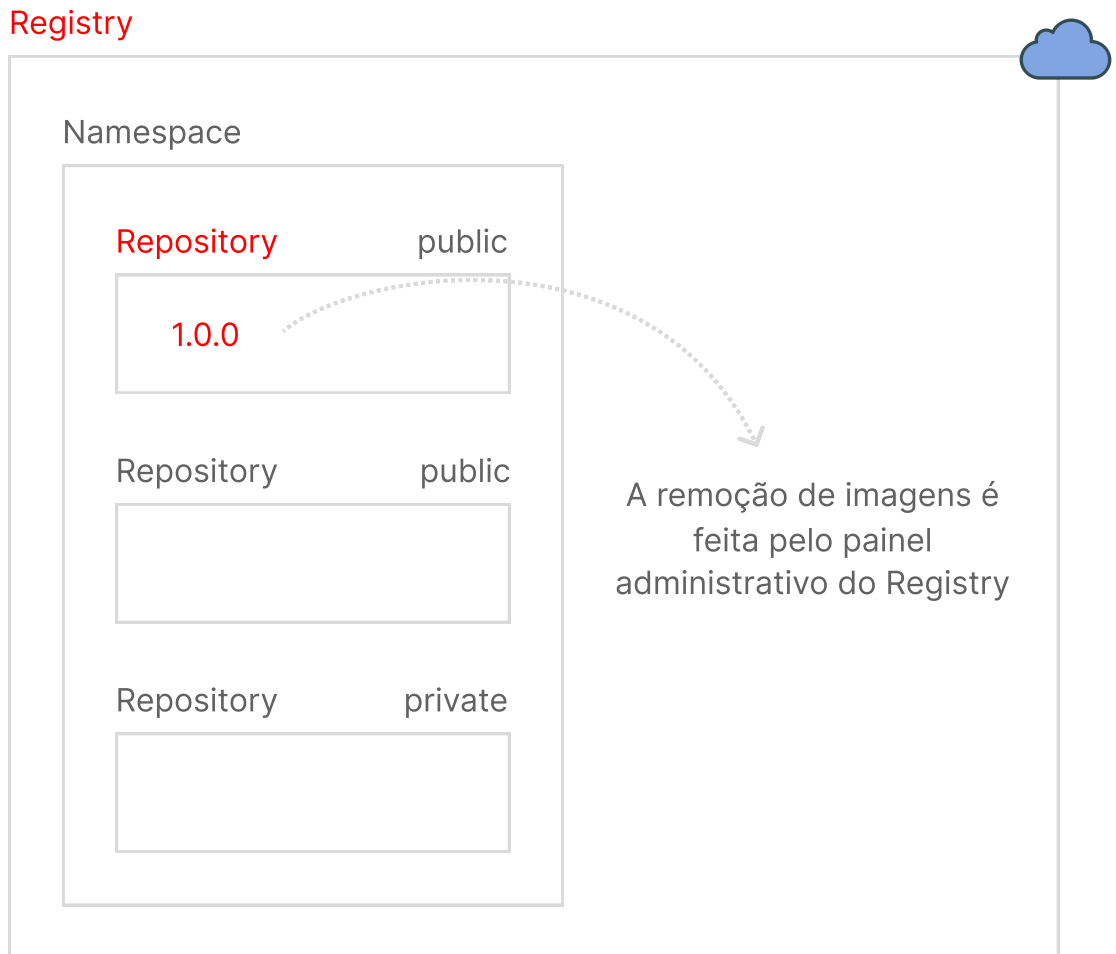


► Registries

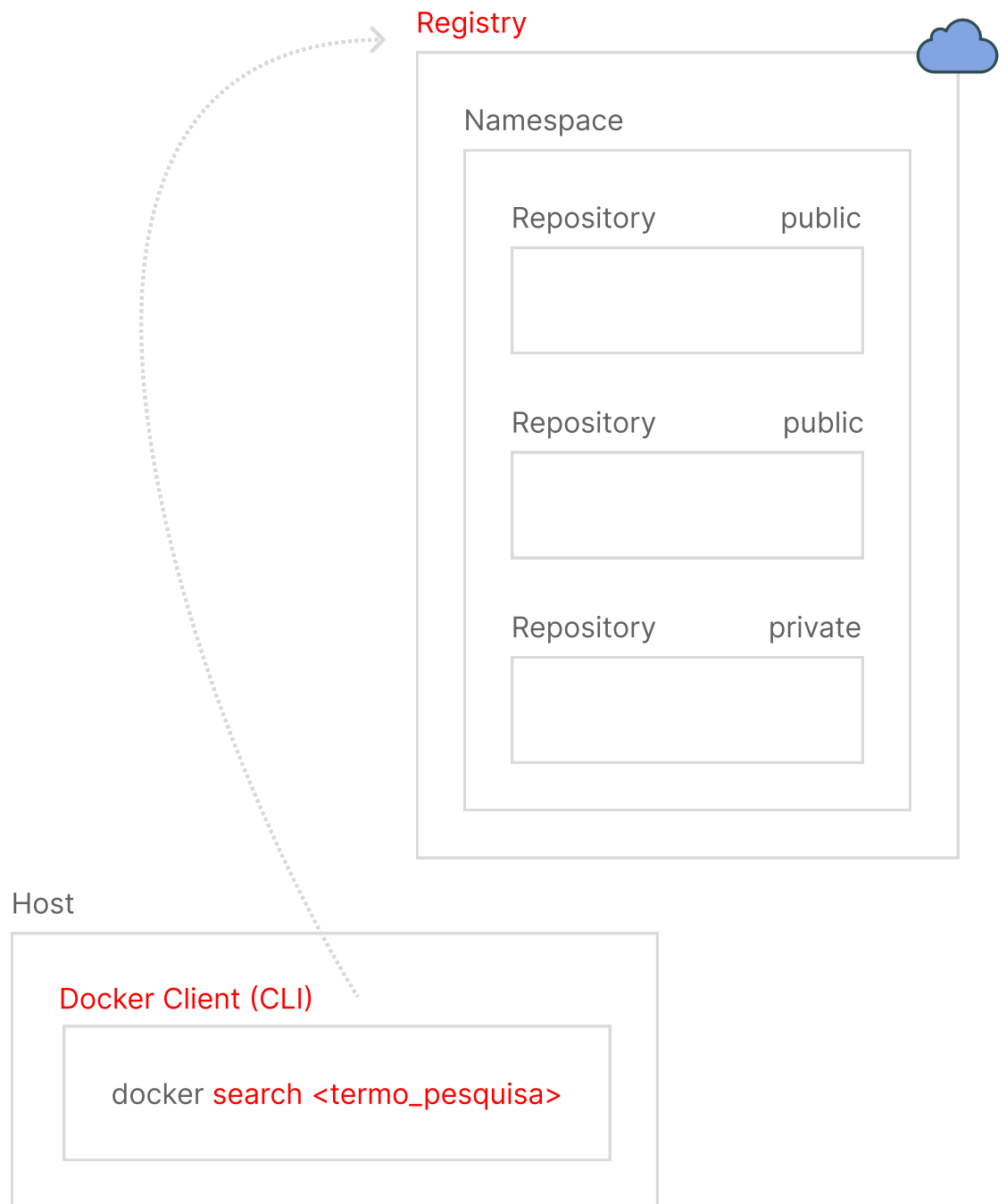
Baixando imagem do registry (pull)



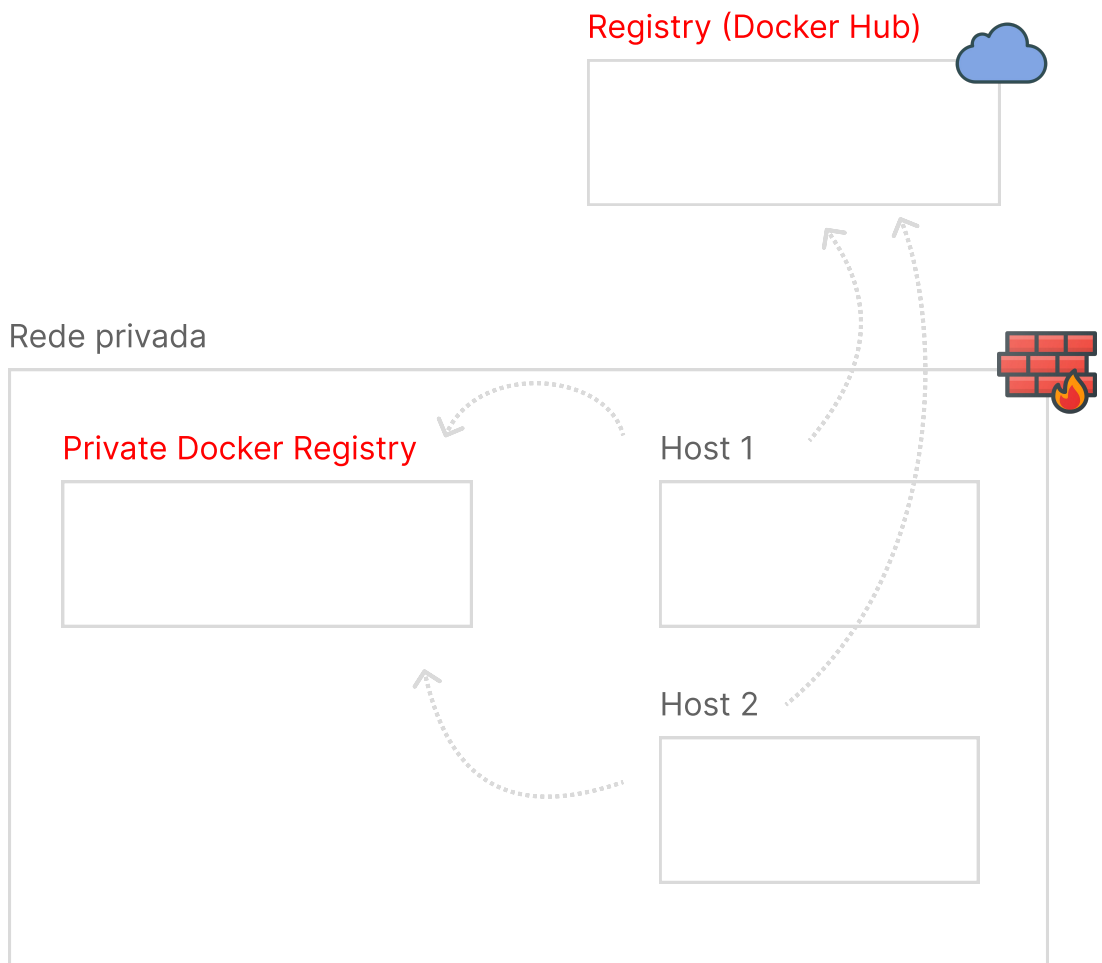
Removendo imagem do registry



Pesquisando por imagens no registry via CLI

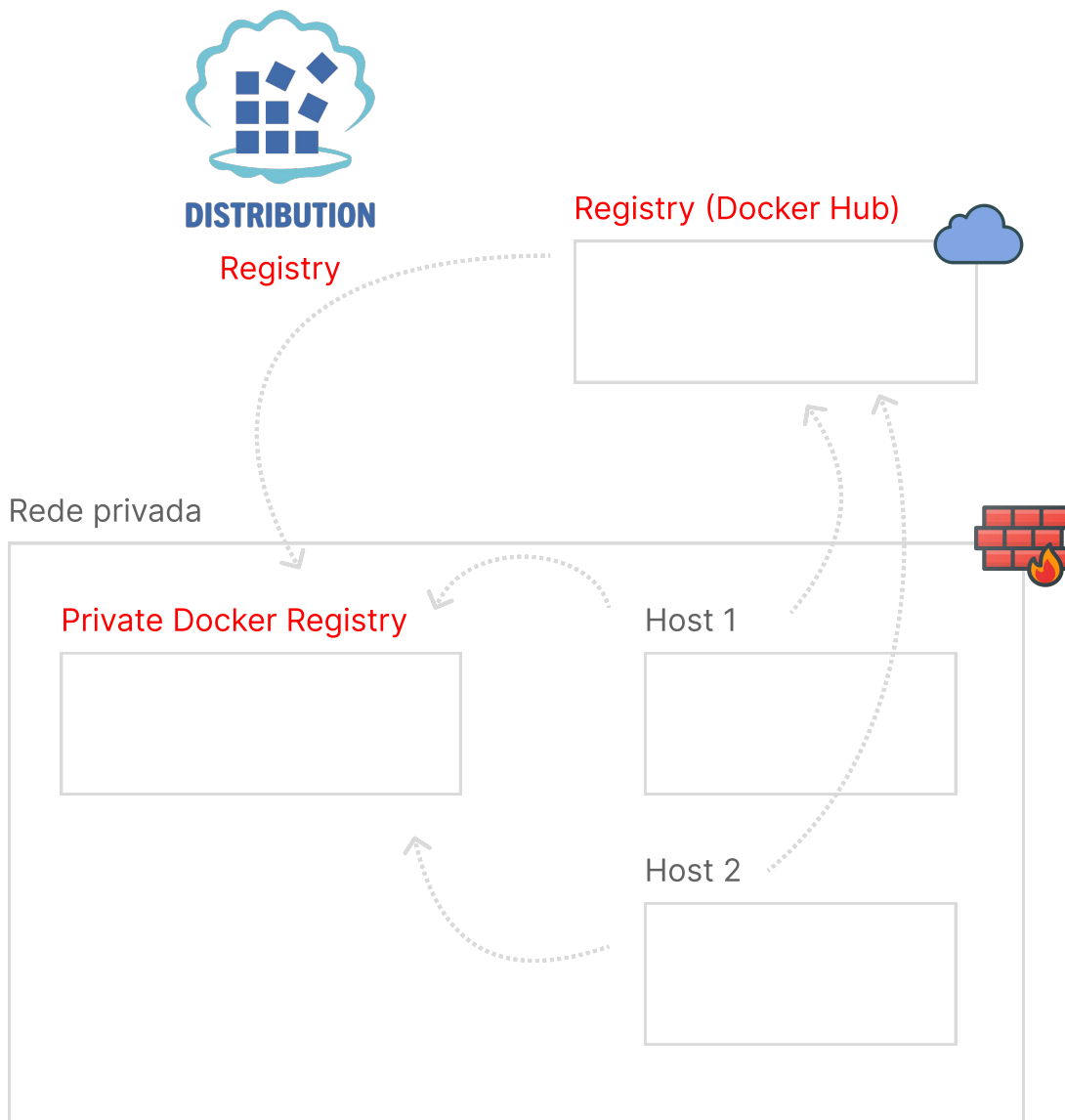


Criando um Registry Privado

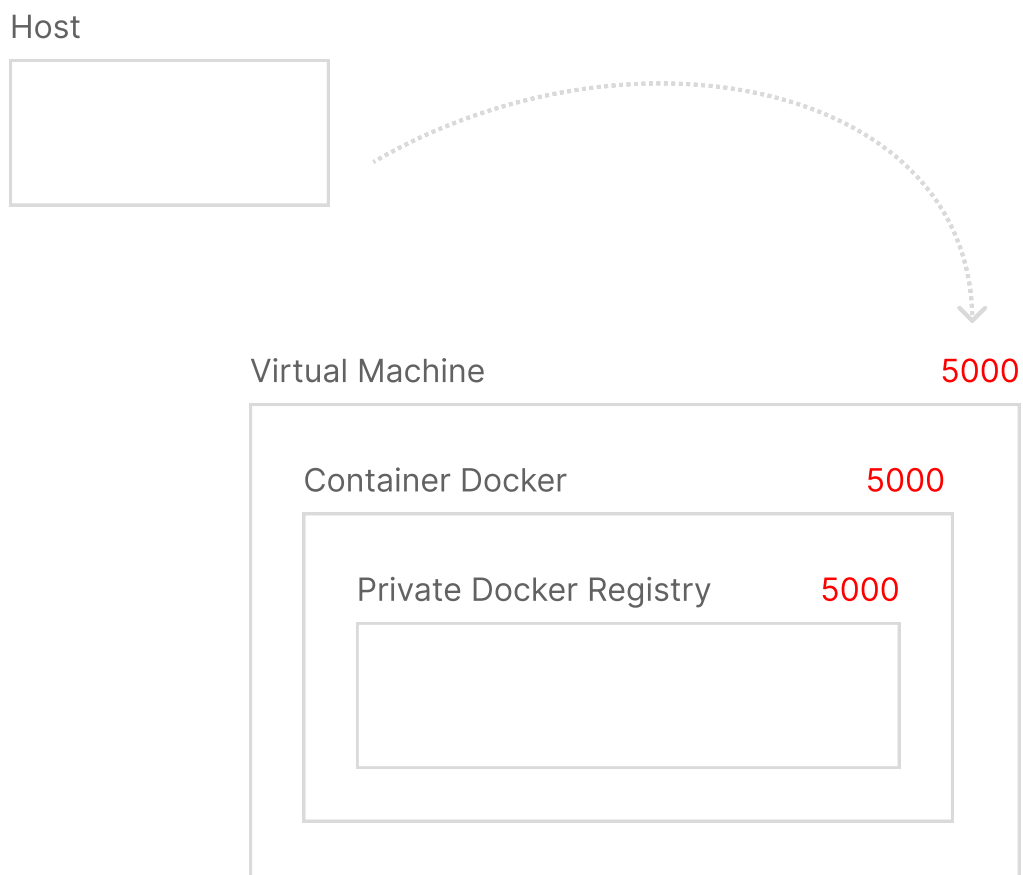


► Registries

Subindo o serviço de Registry Privado

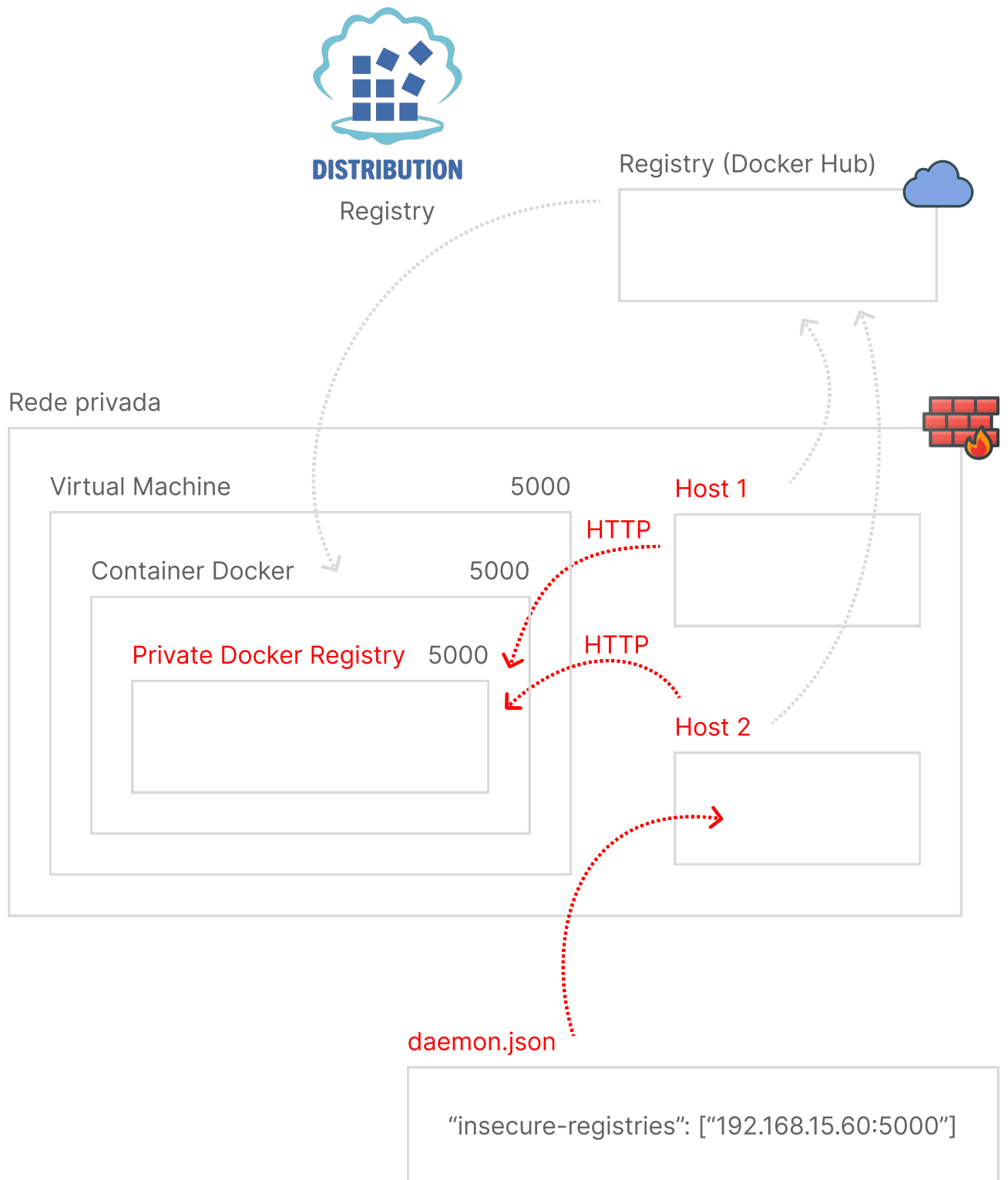


Acessando o IP da VM por meio do Host



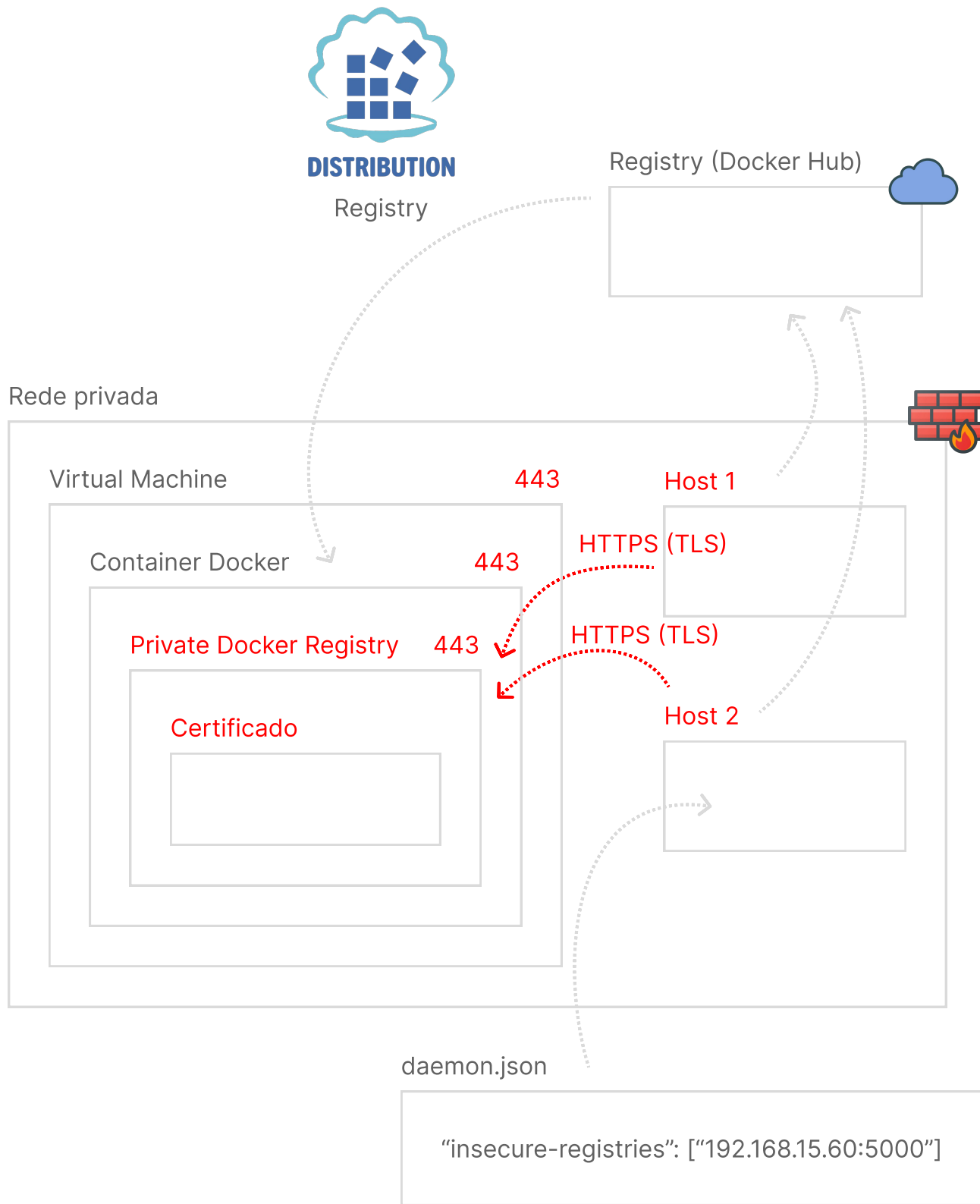
► Registries

Enviando imagens para o Registry de modo inseguro



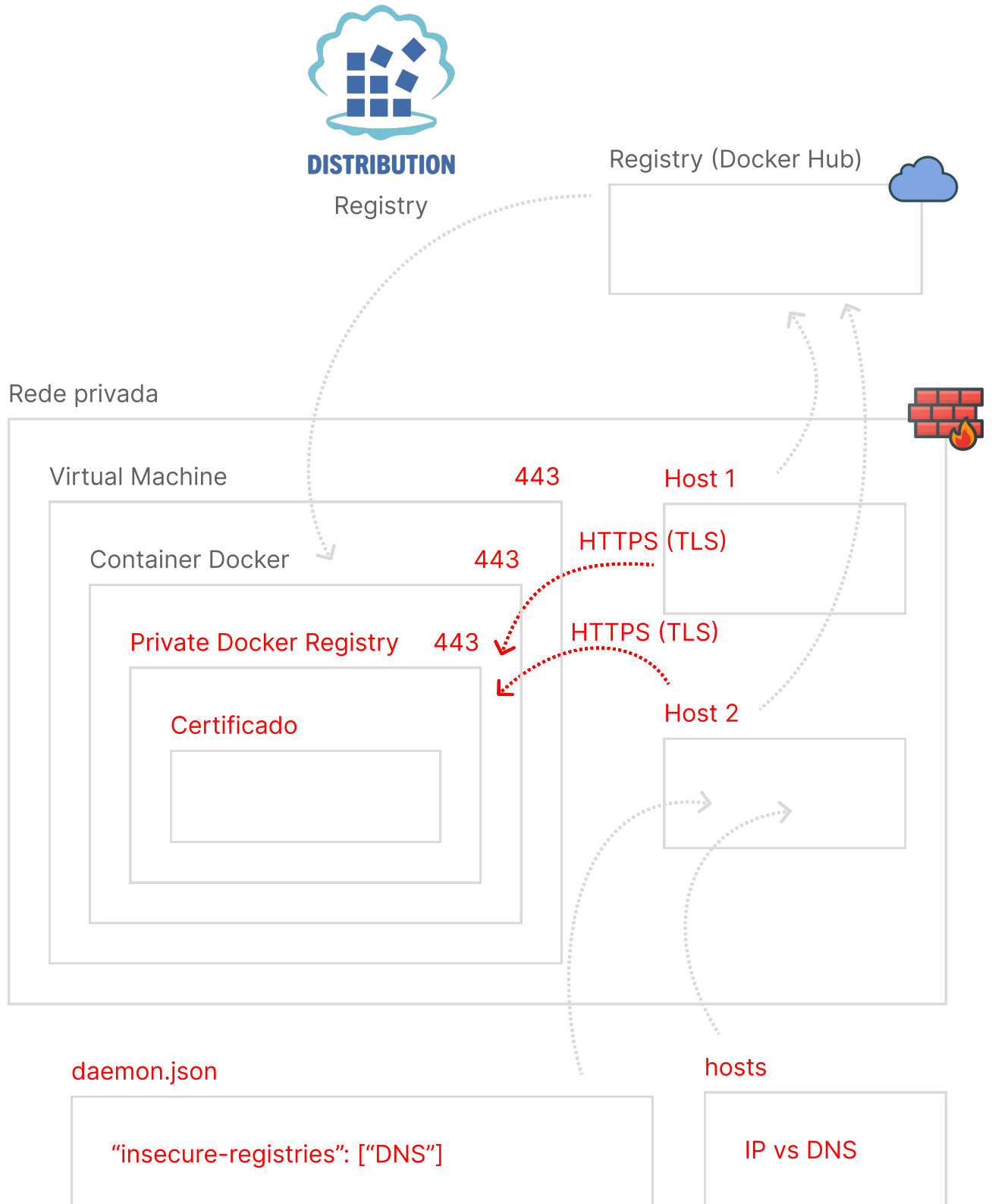
► Registries

Enviando imagens para o Registry de modo seguro

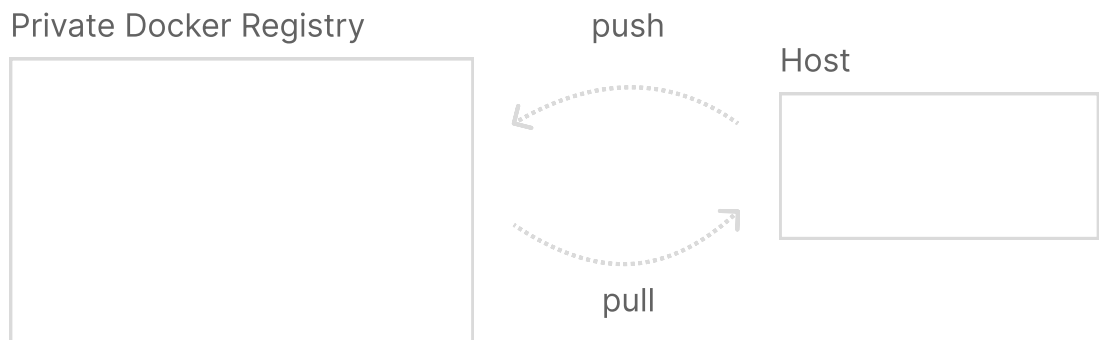


► Registries

Enviando imagens para o Registry de modo seguro parte 2



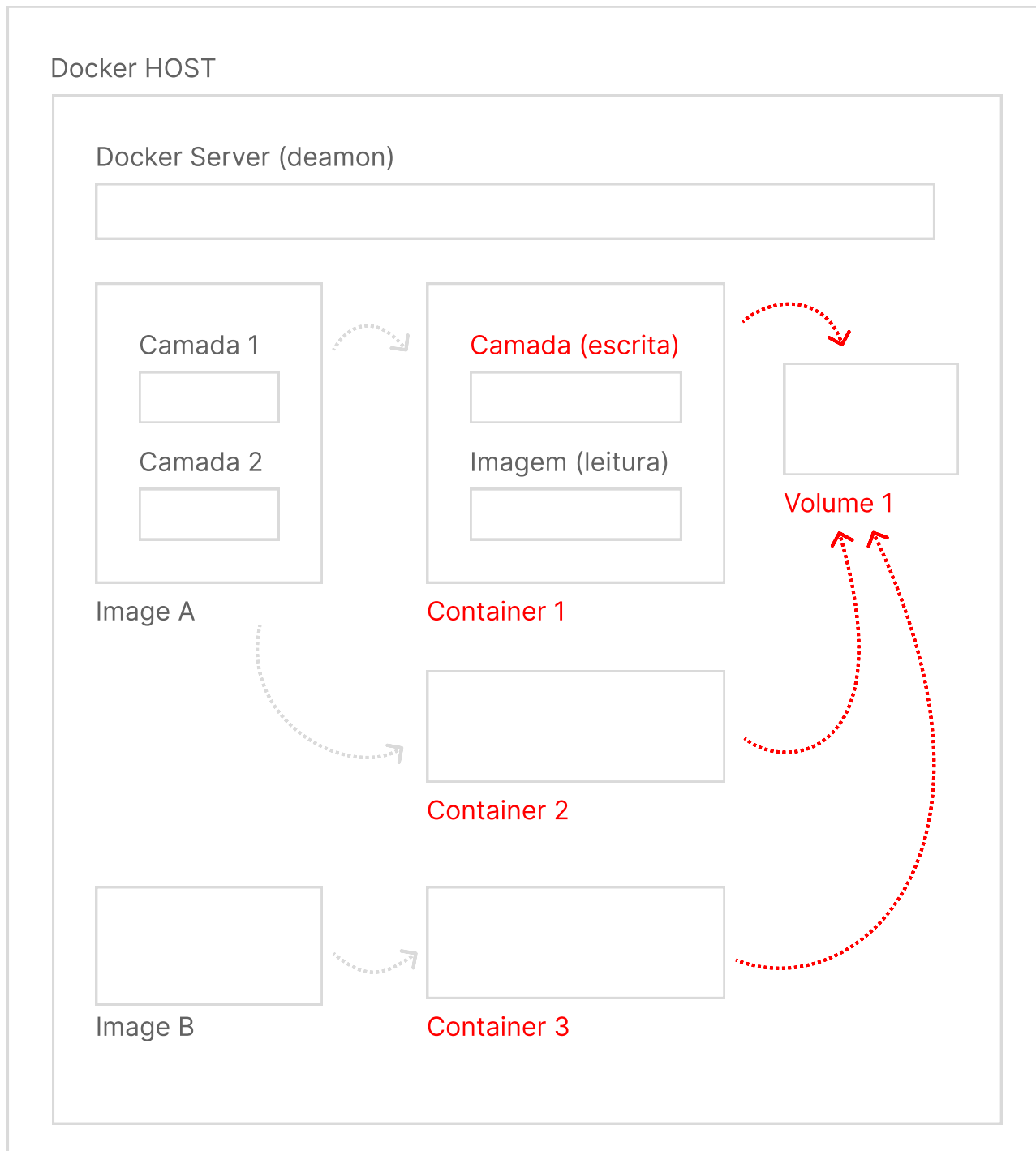
Baixando uma imagem do Registro Privado



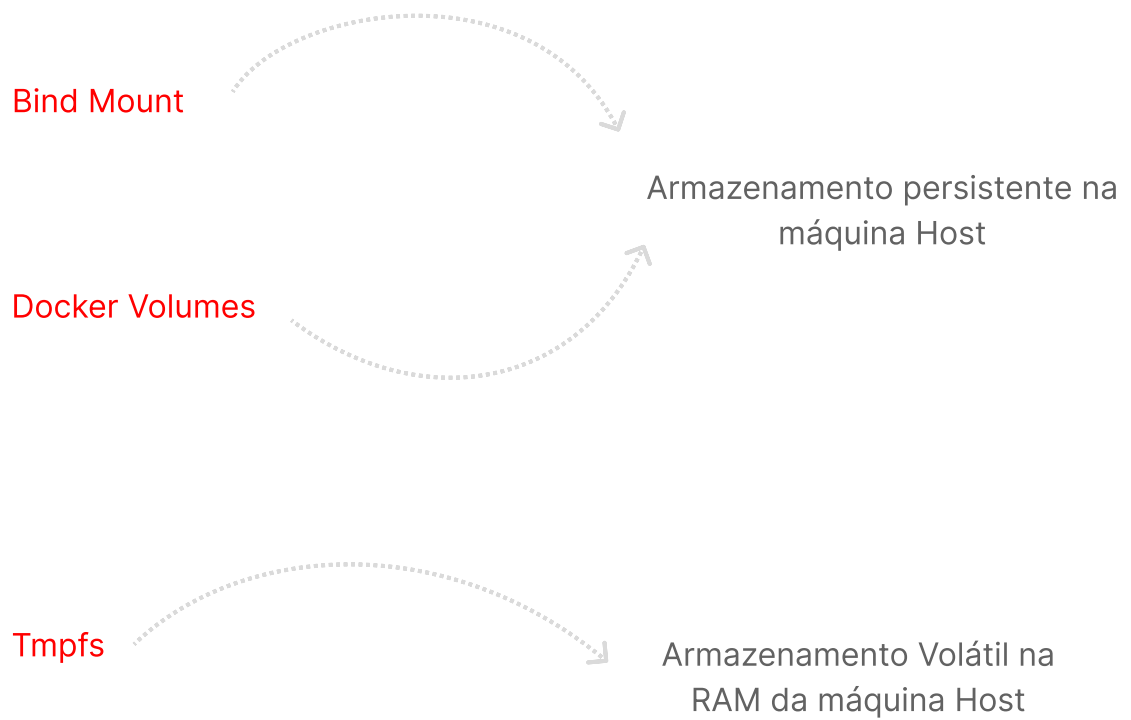
► Volumes

O que são volumes

Host

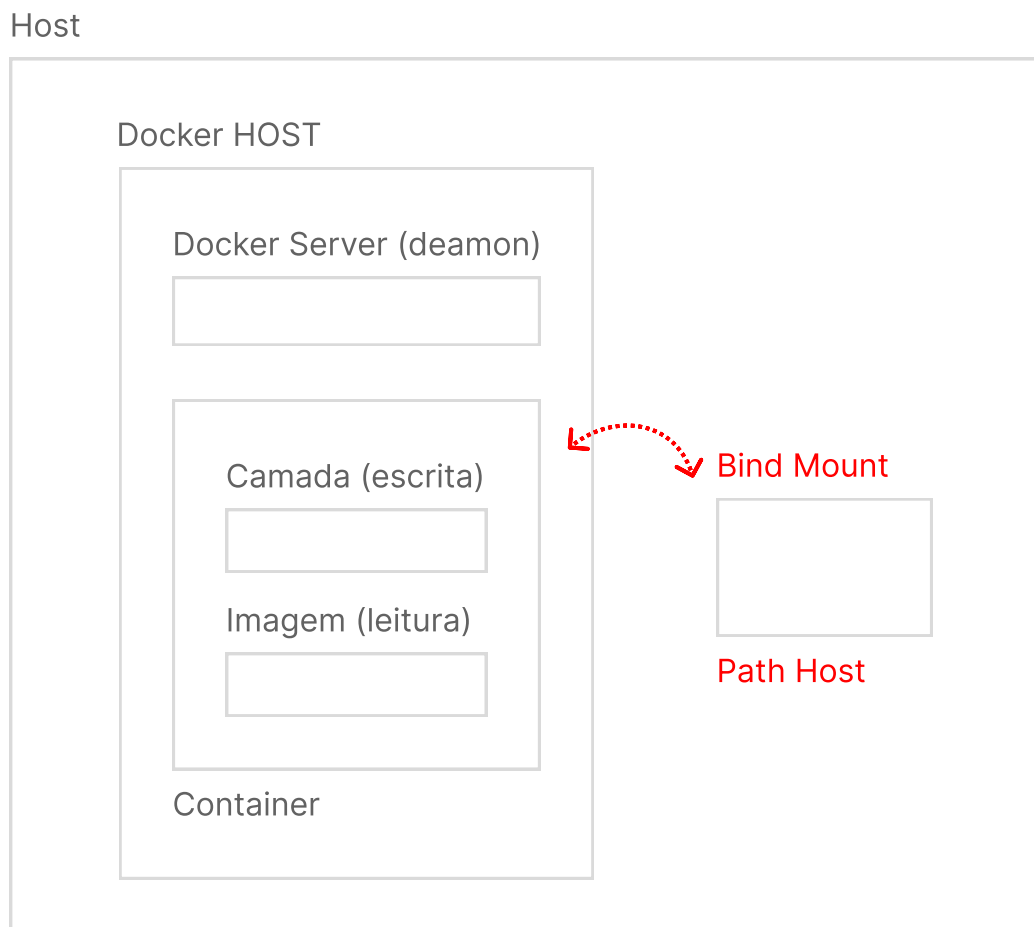


Tipos de montagens



Drivers de Armazenamento

Bind Mount - Casos de Uso

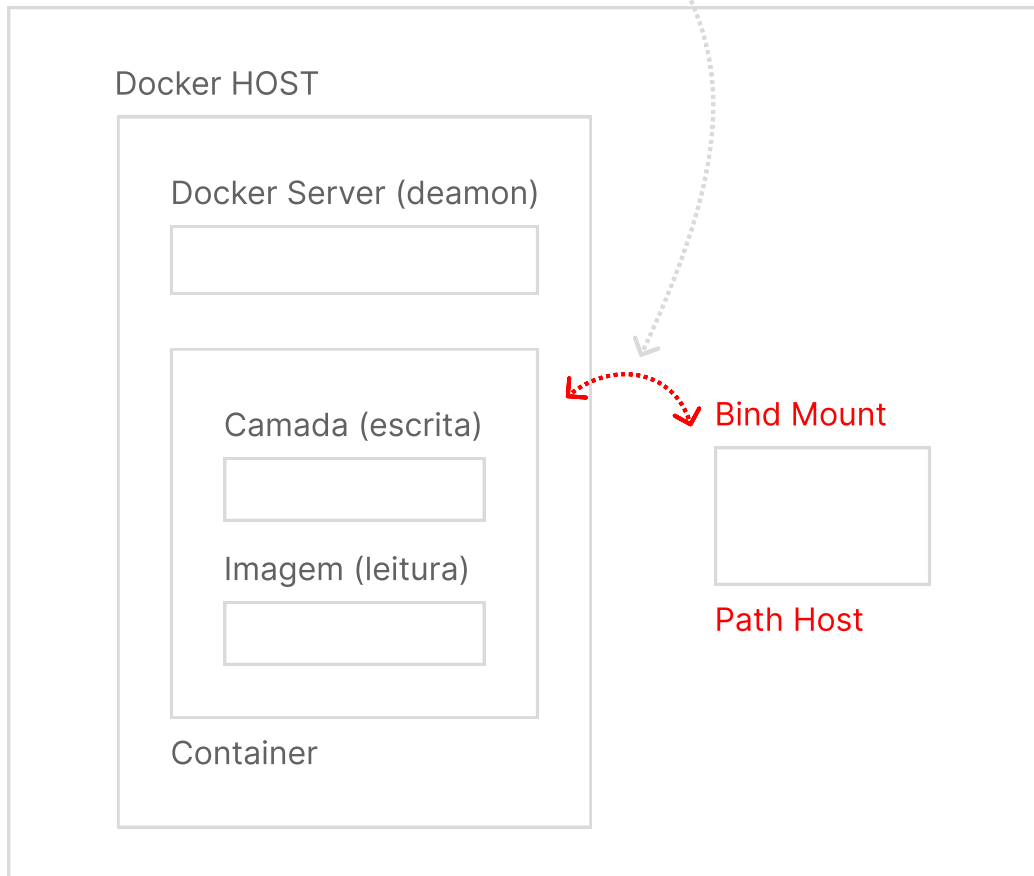


Bind Mount - Solucionando o problema

Docker Client (CLI)

```
docker run -d -p 127.0.0.1:3000:3000 \
--mount type=bind,source=path-host,target=/app getting-started
```

Host



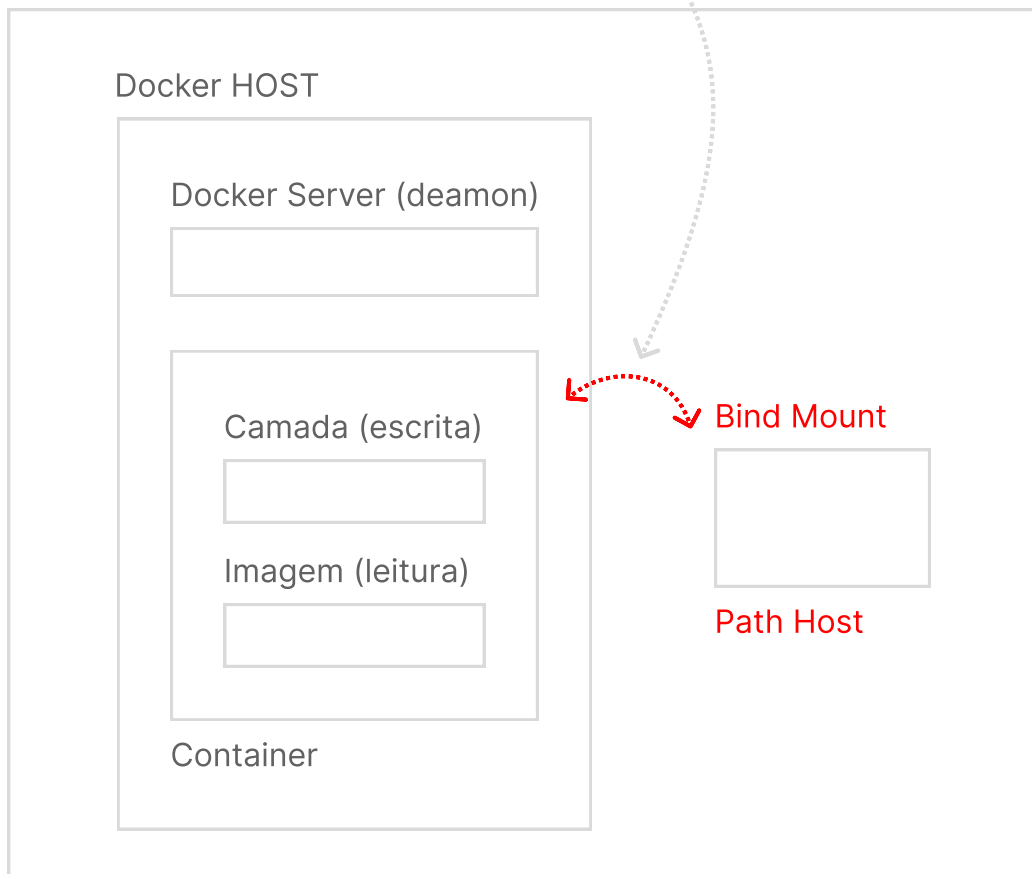
Bind Mount - Sintaxe alternativa

Docker Client (CLI)

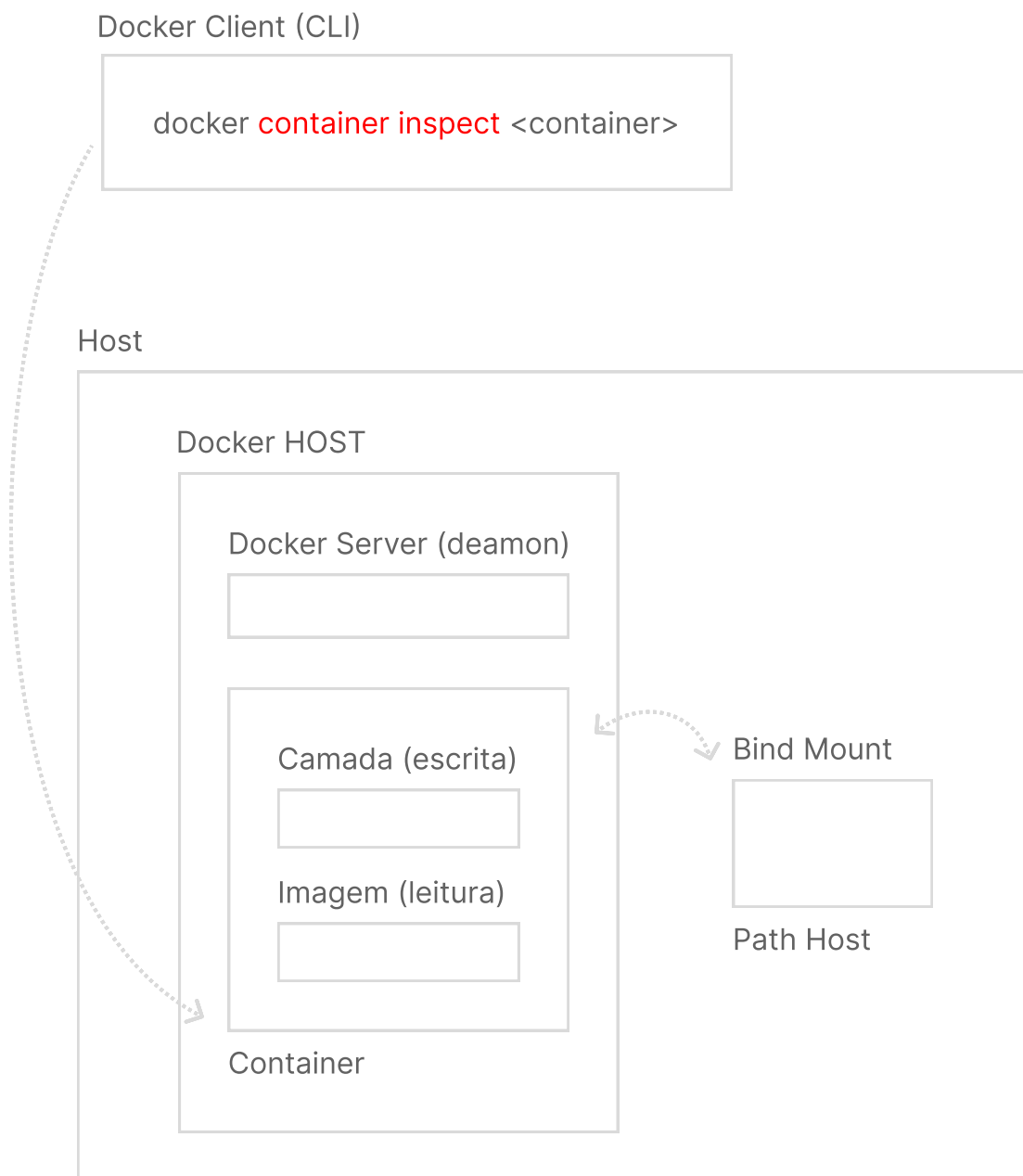
```
docker run -d -p 127.0.0.1:3000:3000 \
--mount type=bind,source=path-host,target=/app getting-started

docker run -d -p 127.0.0.1:3000:3000 \
-v path-host:path-container getting-started
```

Host



Inspecionando o container para verificar os pontos de montagem



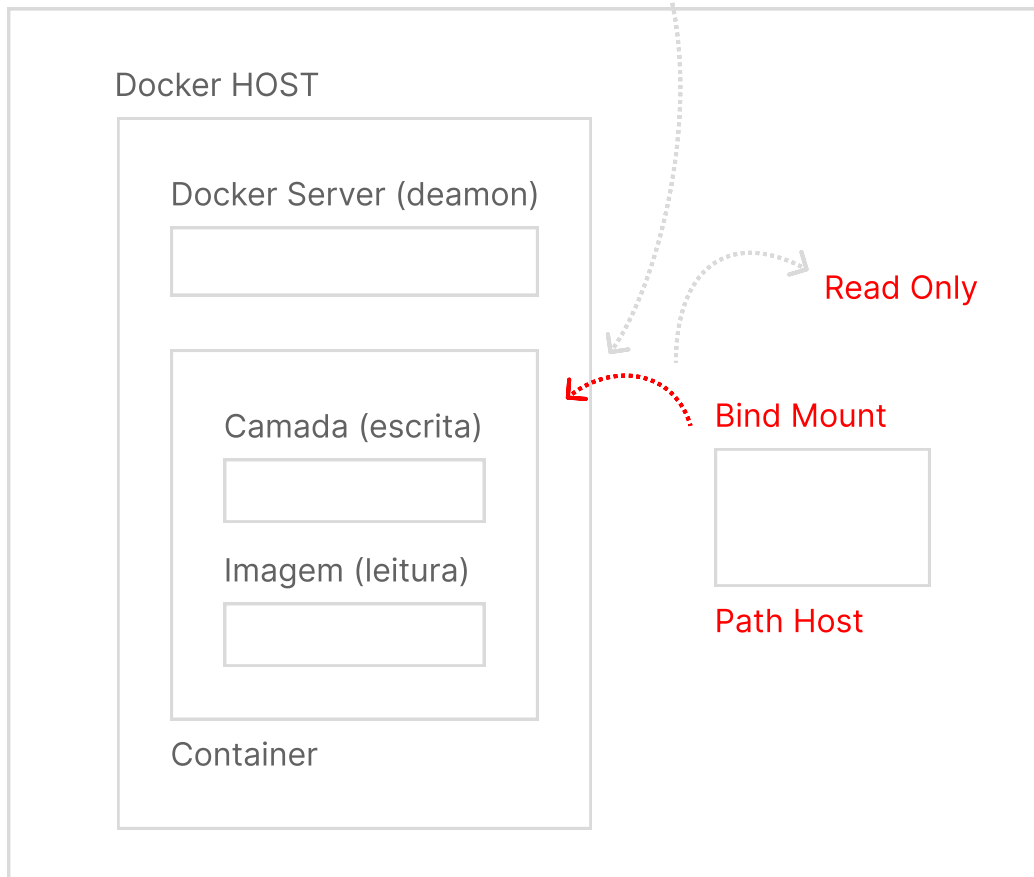
Bind Mount - Somente leitura

Docker Client (CLI)

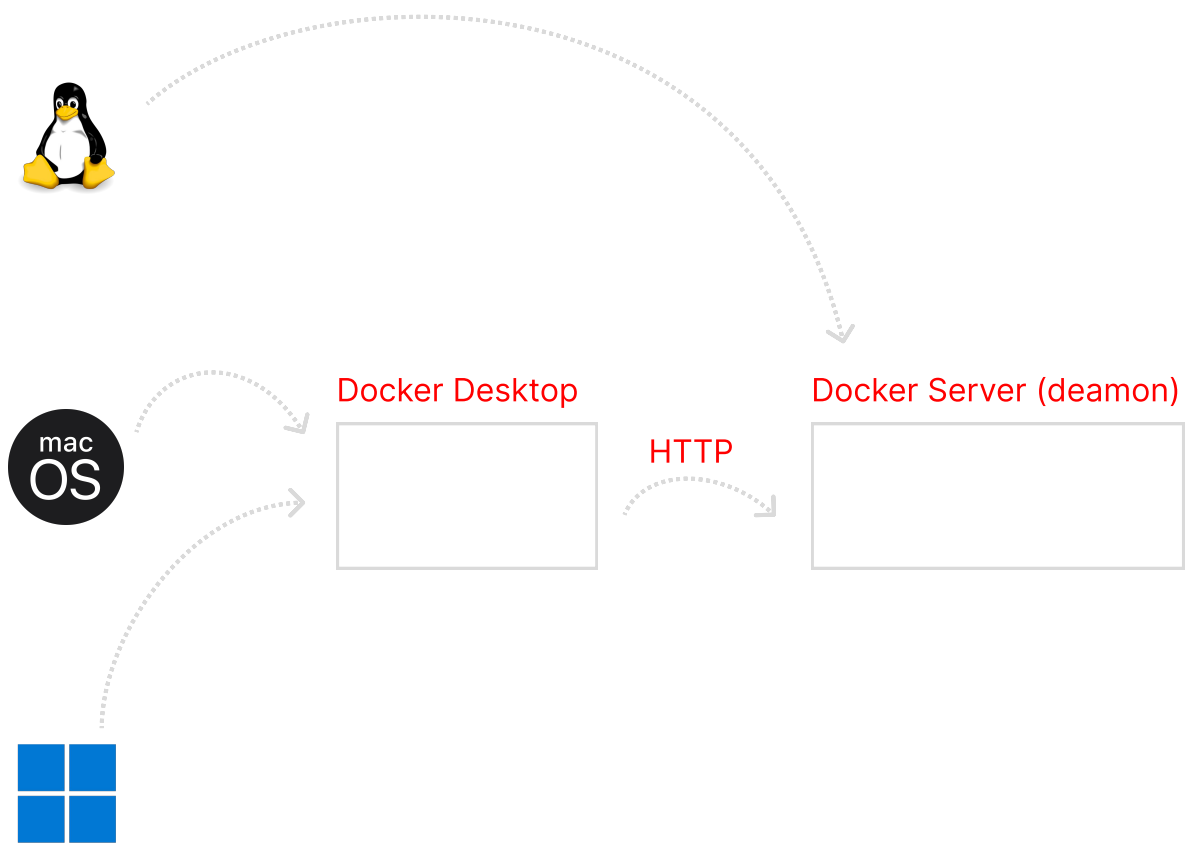
```
docker run -d -p 127.0.0.1:3000:3000 \
--mount type=bind,source=path-host,target=/app,readonly \
getting-started

docker run -d -p 127.0.0.1:3000:3000 \
-v path-host:path-container:ro getting-started
```

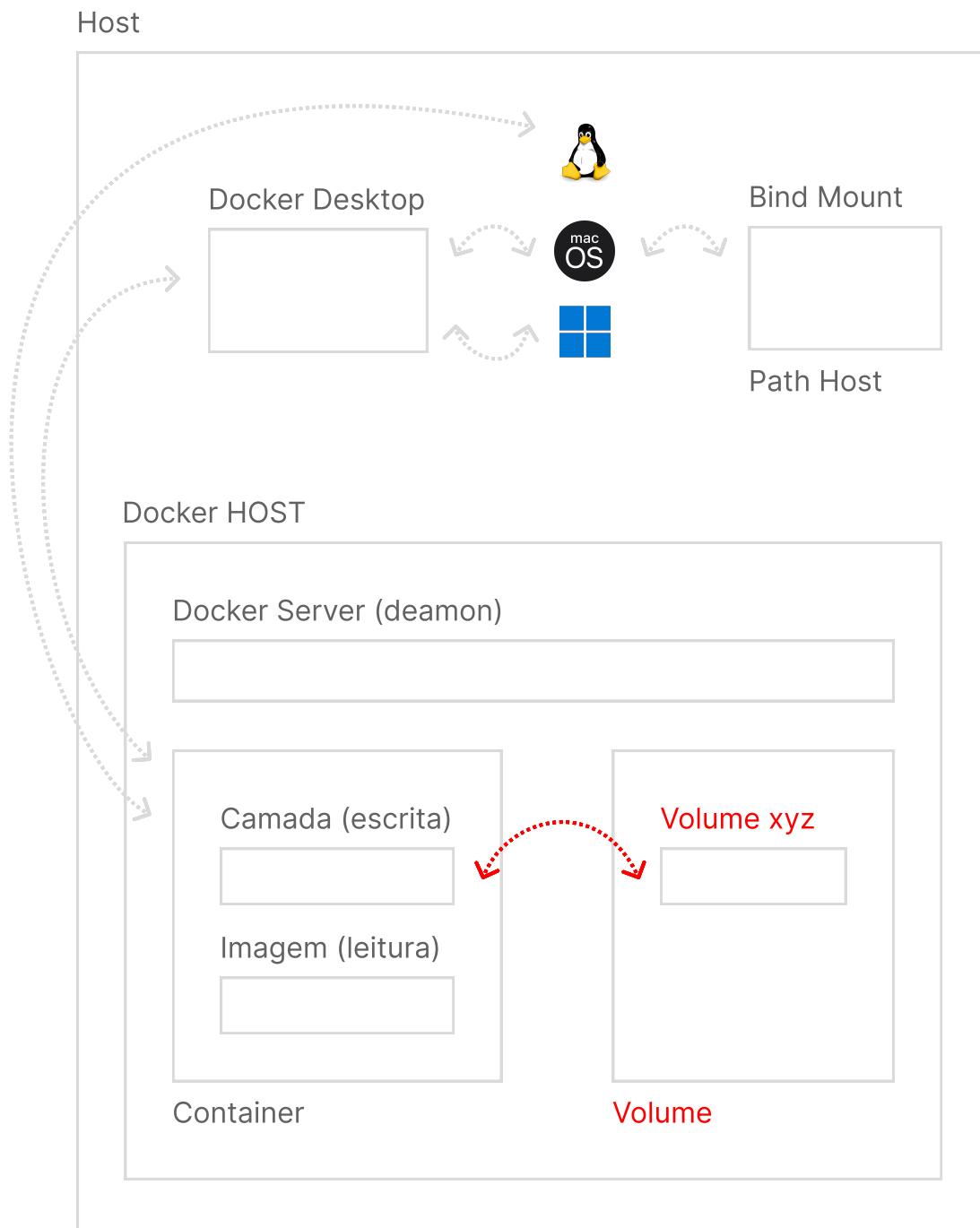
Host



Bind Mount - Considerações sobre o funcionamento e performance



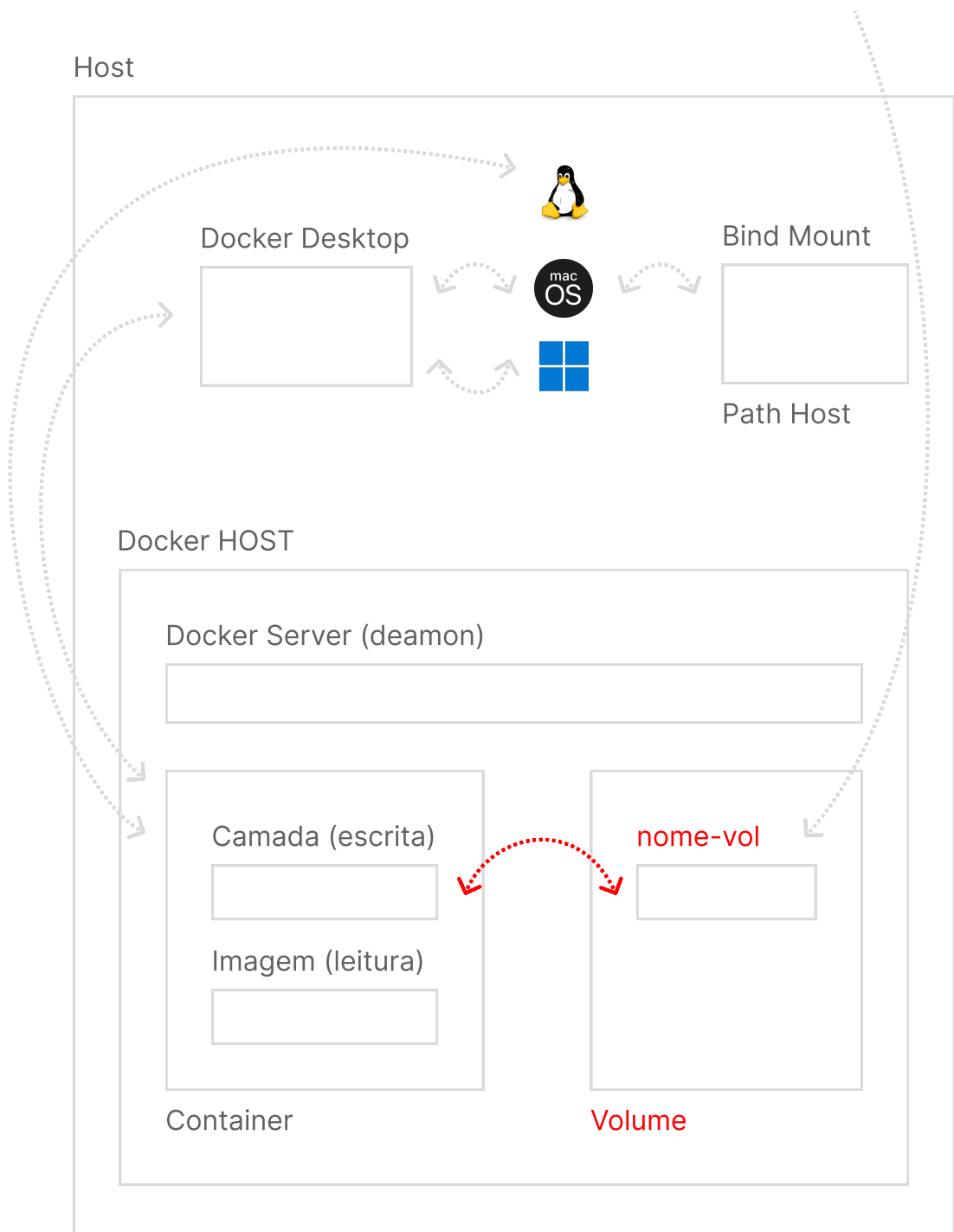
Docker Volume - Casos de Uso



Docker Volume - Solucionando o problema

Docker Client (CLI)

```
docker run -d -p 5432:5432 \
--mount type=volume,source=nome-vol,target=/data postgresql
```



► Volumes

Docker Volume - Localizando, inspecionando e manipulando os volumes

Docker Client (CLI)

```
docker volume ls

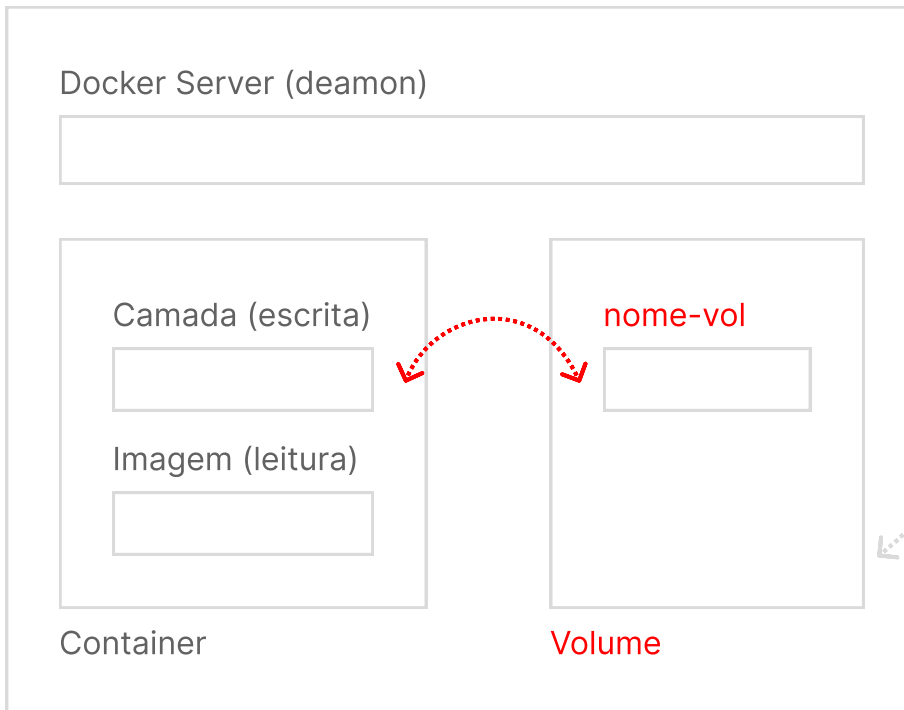
docker volume inspect <nome_volume>

docker inspect <nome_volume>

docker volume create <nome_volume>

docker volume rm <nome_volume>
```

Docker HOST

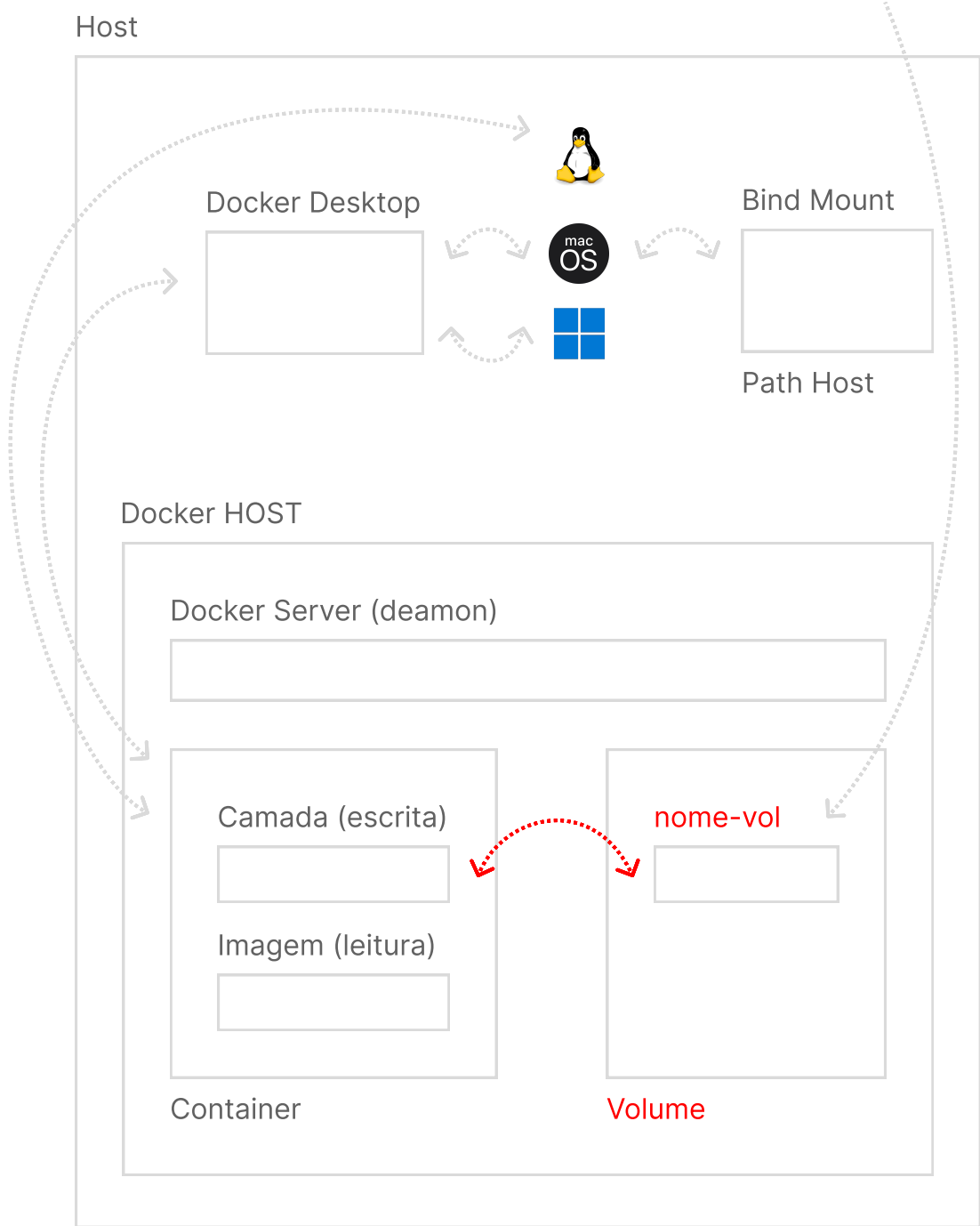


Docker Volume - Sintaxe alternativa

Docker Client (CLI)

```
docker run -d -p 5432:5432 \
--mount type=volume,source=nome-vol,target=/data postgresql

docker run -d -p 5432:5432 -v nome-vol:/data postgresql
```



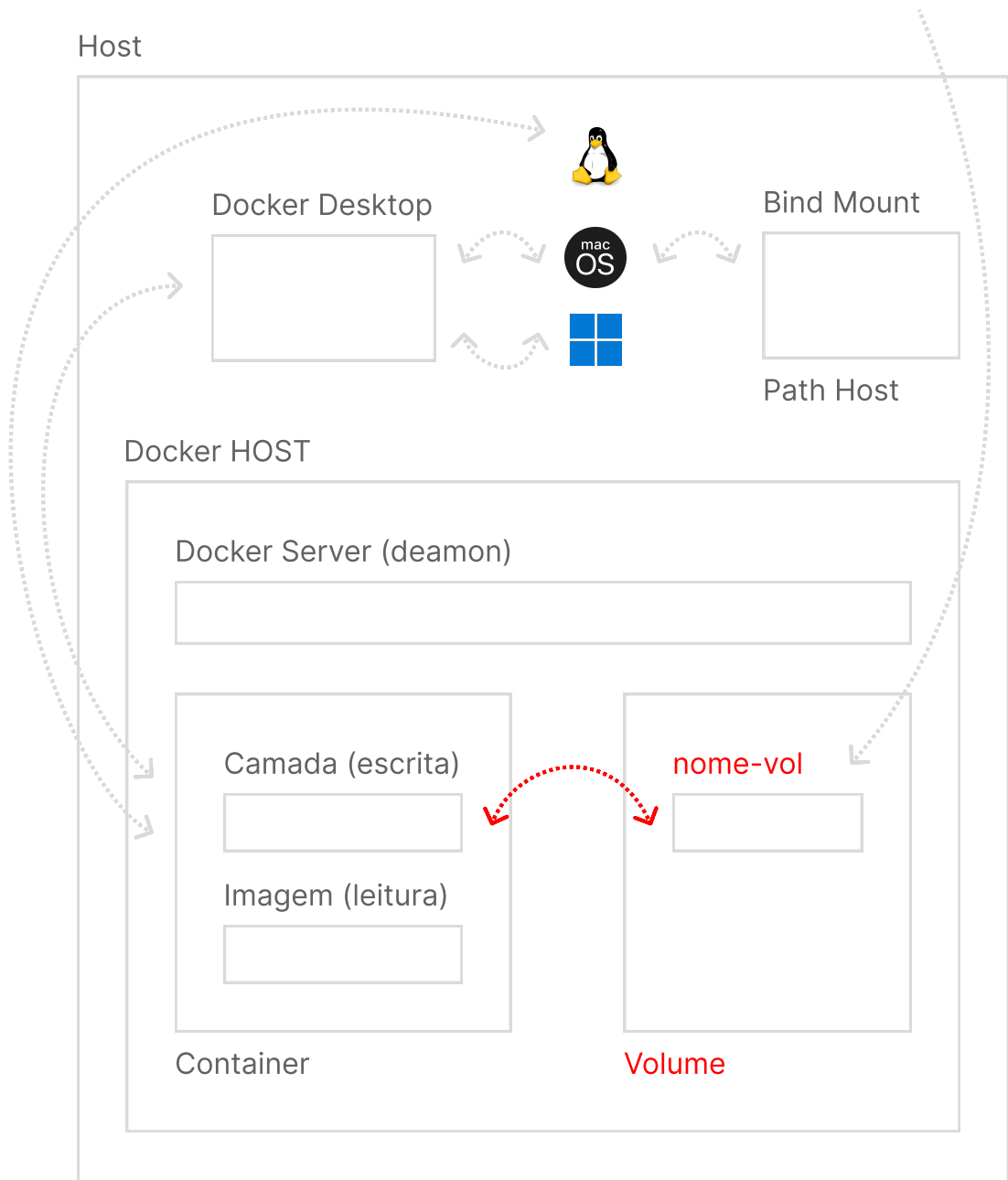
Docker Volume - Sintaxe alternativa parte 2

Docker Client (CLI)

```
docker run -d -p 5432:5432 \
--mount type=volume,source=nome-vol,target=/data postgresql

docker run -d -p 5432:5432 -v nome-vol:/data postgresql

docker run -d -p 5432:5432 -v /data postgresql
```



Docker Volume - Somente leitura

Docker Client (CLI)

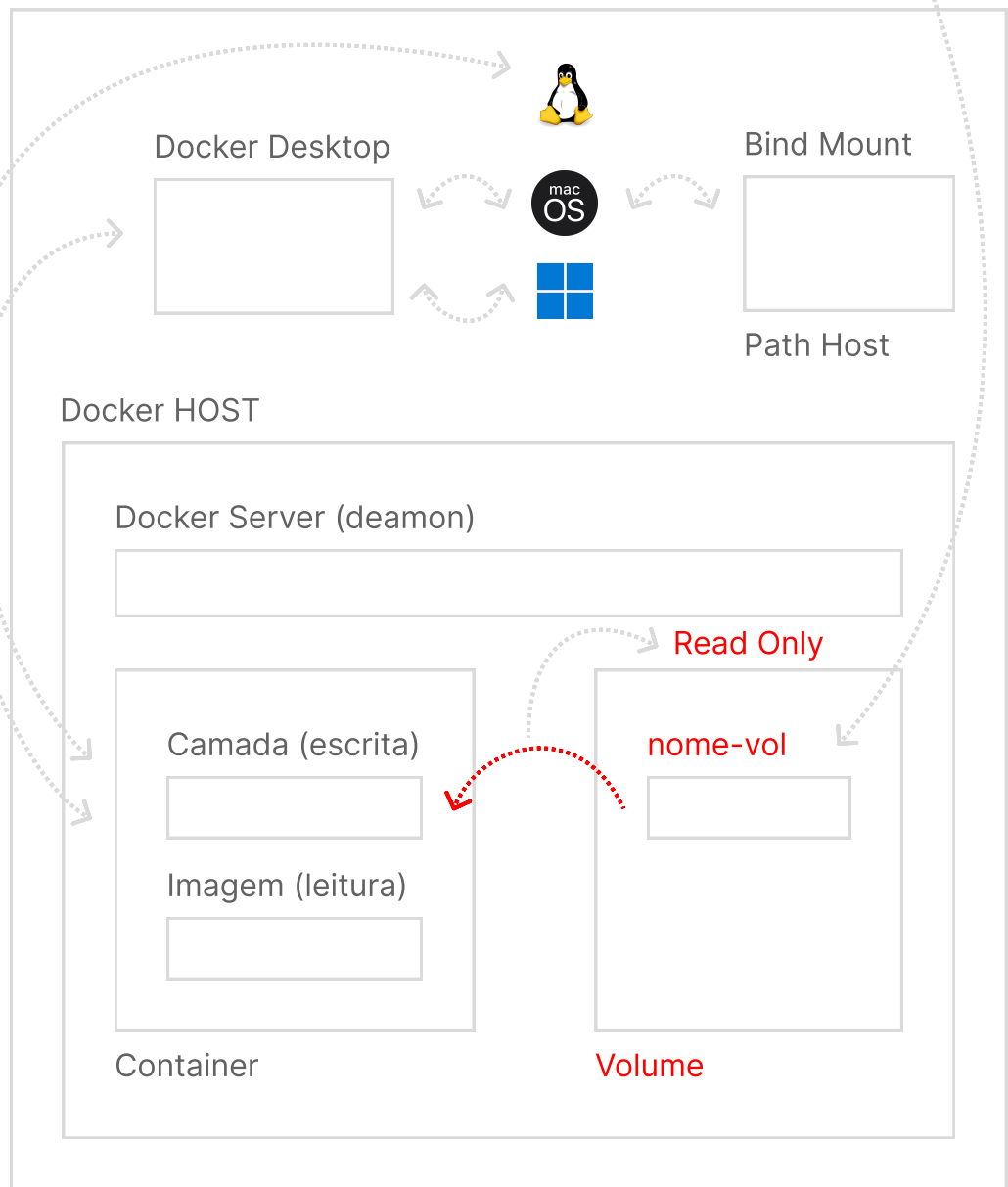
```
docker run -d -p 5432:5432 \
--mount type=volume,source=nome-vol,target=/data,readonly postgresql
```

```
docker run -d -p 5432:5432 -v nome-vol:/data:ro postgresql
```

```
docker run -d -p 5432:5432 -v /data:ro postgresql
```

Erro de sintaxe

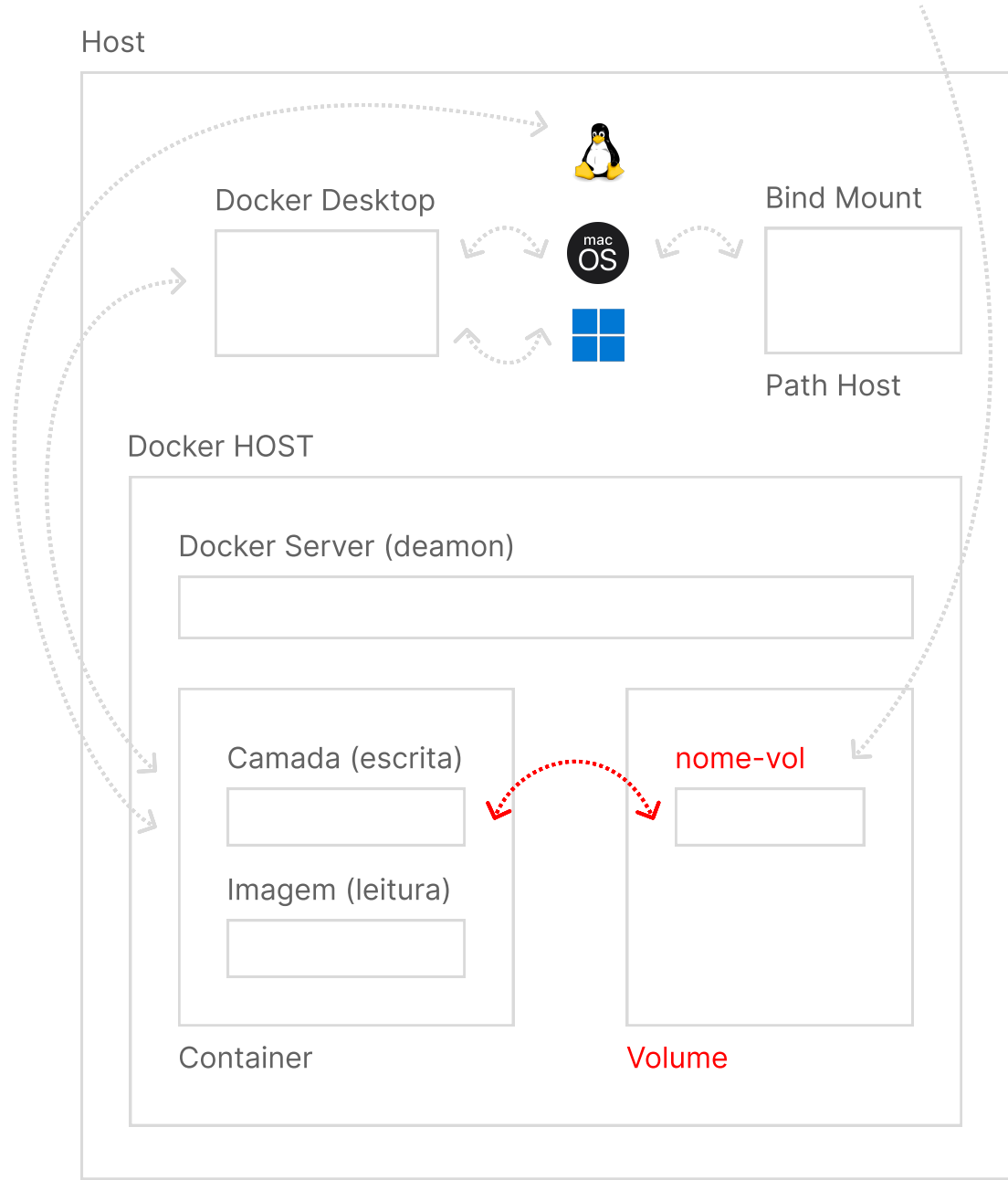
Host



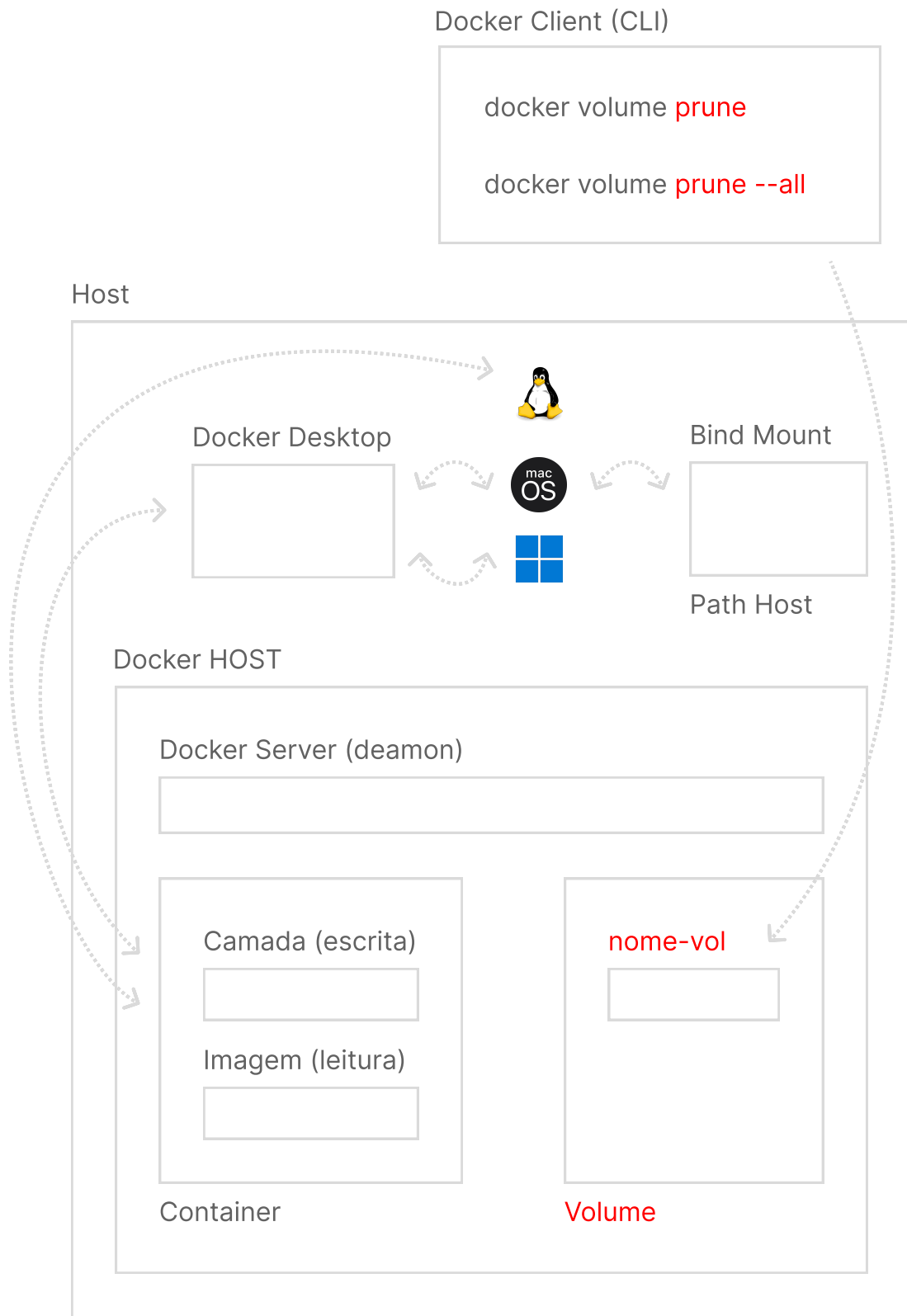
Docker Volume - Definição no Dockerfile

Dockerfile

```
VOLUME ["/var/log"]  
VOLUME /var/log  
VOLUME /var/log /var/db
```

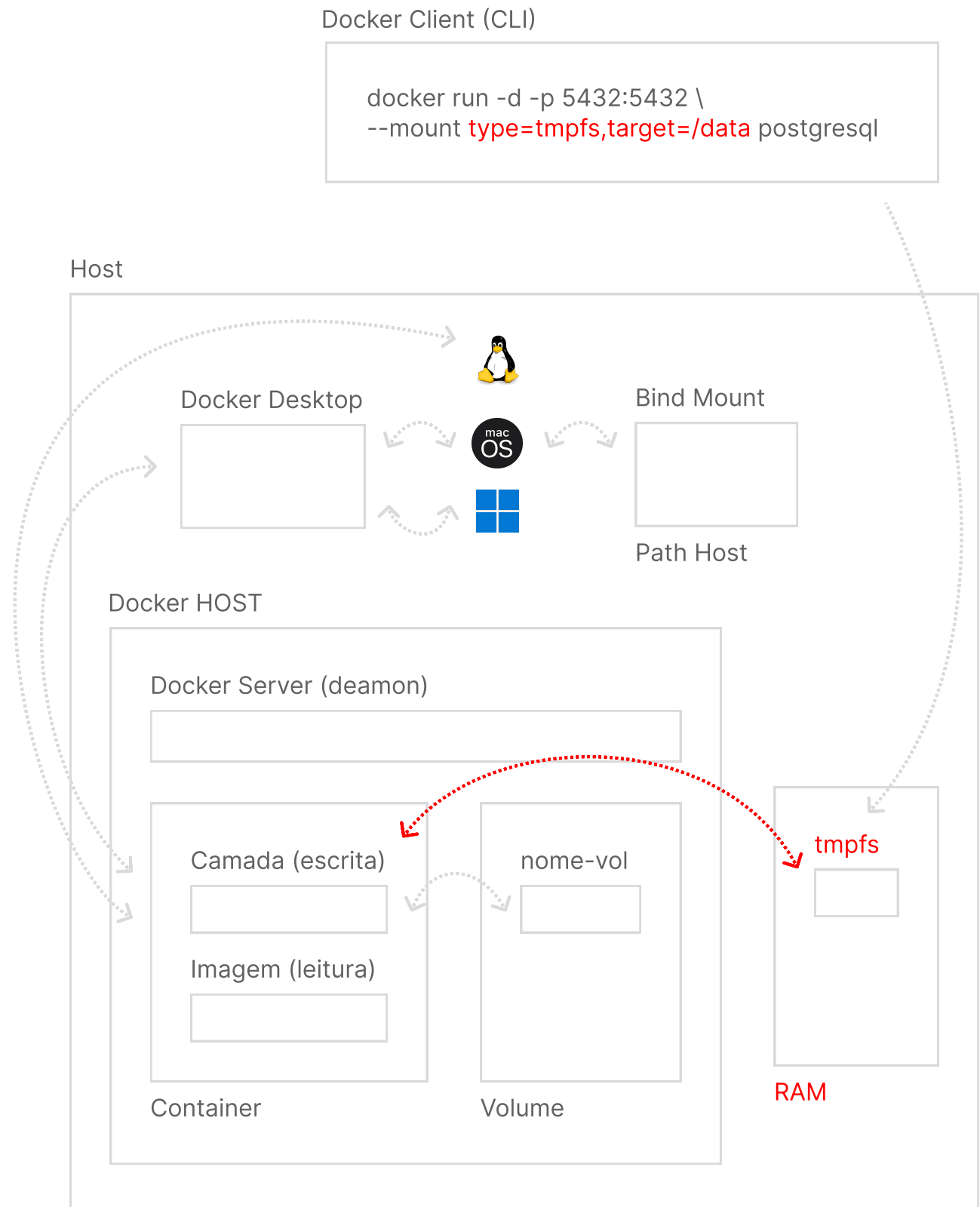


Removendo volumes em massa



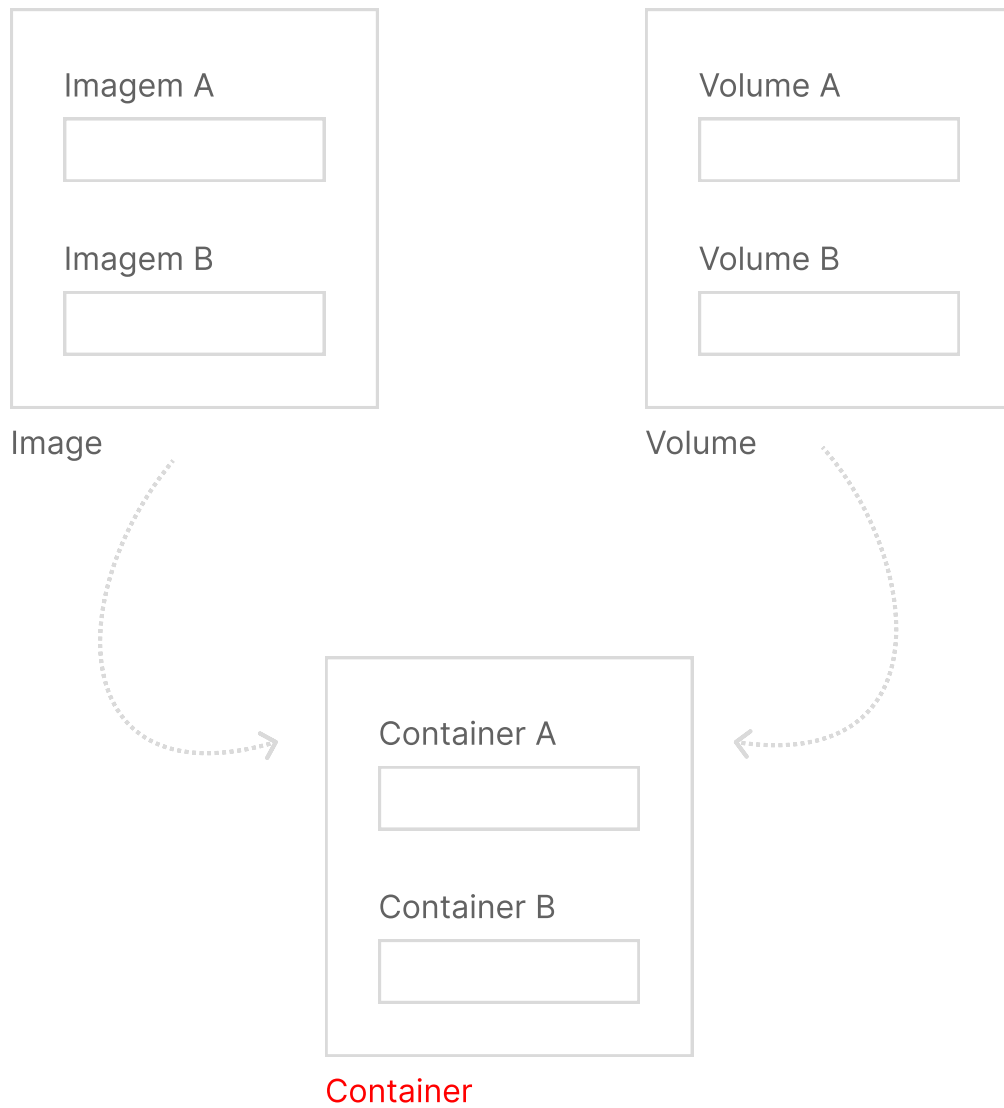
► Volumes

TMPFS - Armazenamento de dados não persistentes



► Containers

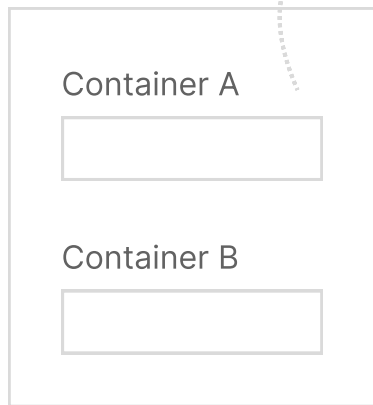
O que são containers



Executando, parando, iniciando e reiniciando containers

Docker Client (CLI)

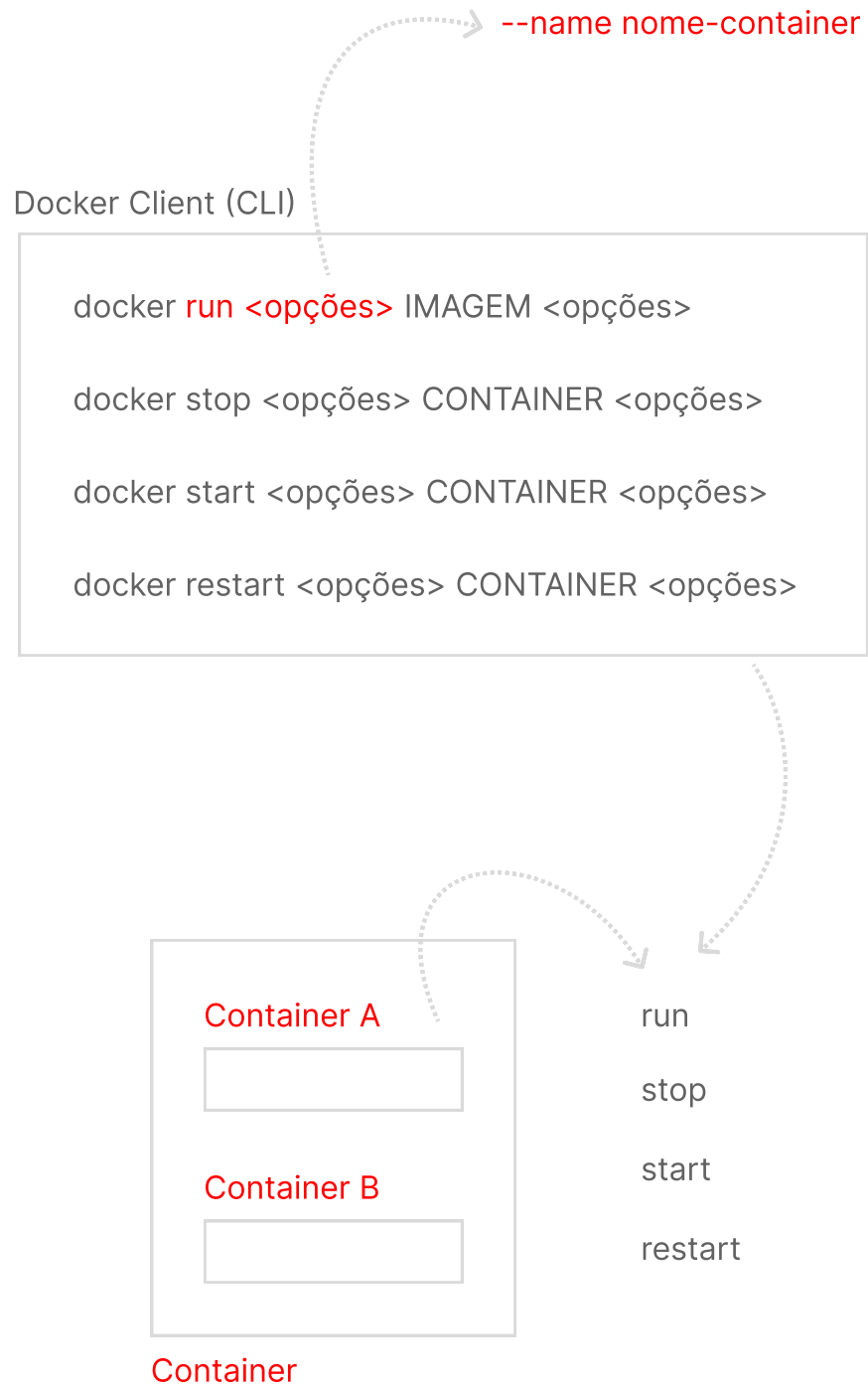
```
docker run <opções> IMAGEM <opções>
docker stop <opções> CONTAINER <opções>
docker start <opções> CONTAINER <opções>
docker restart <opções> CONTAINER <opções>
```



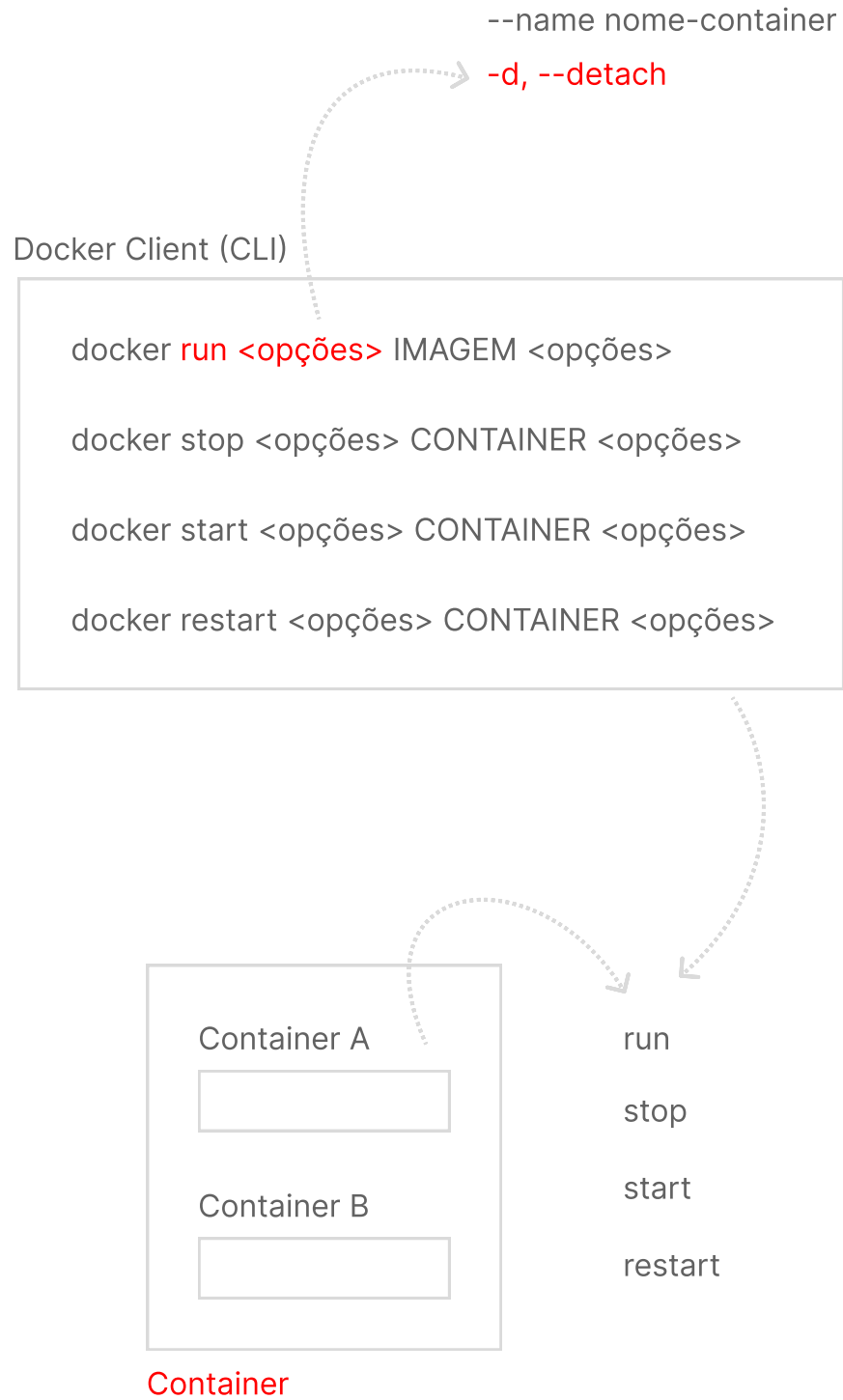
- run
- stop
- start
- restart

Container

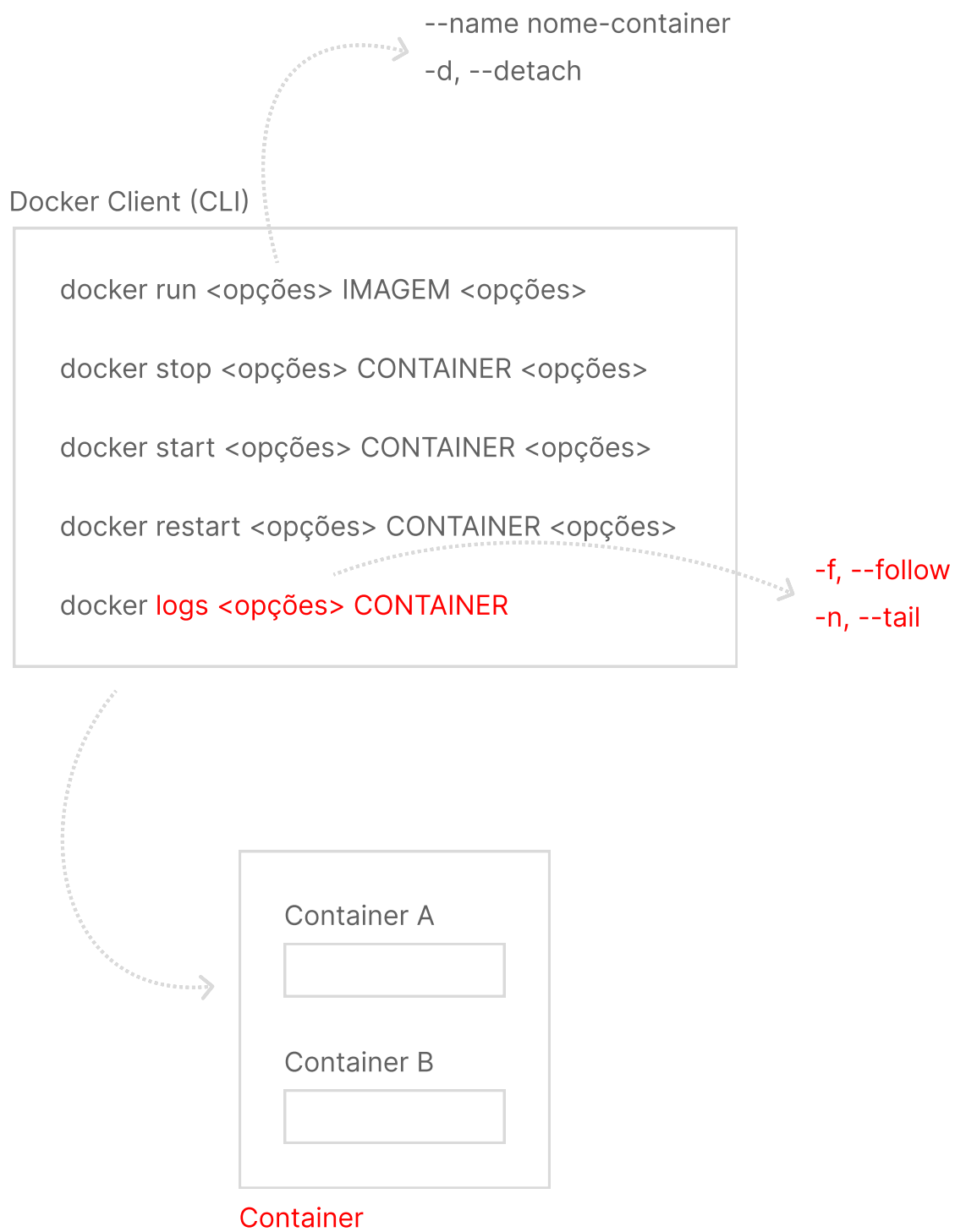
Nomeando o container



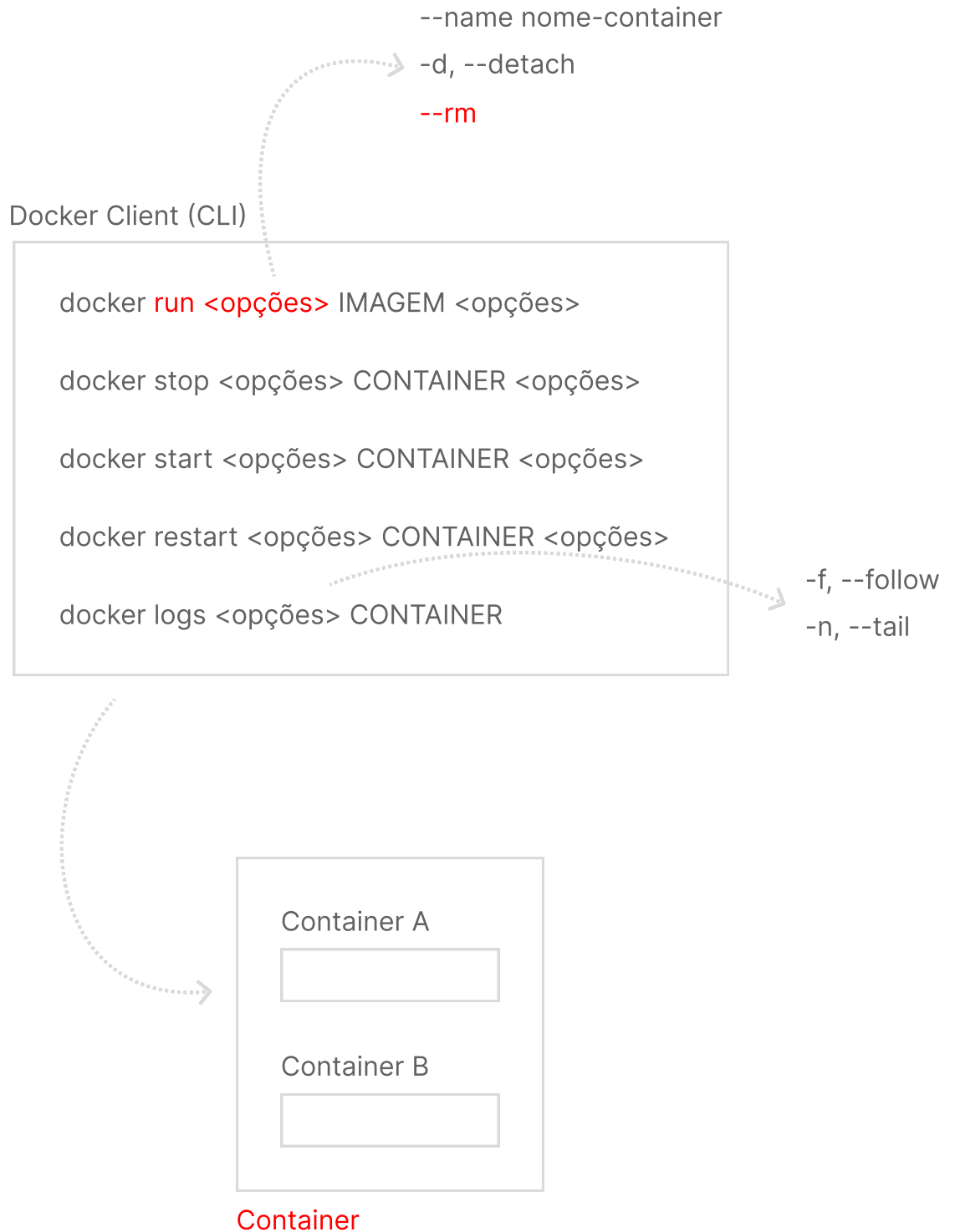
Execução do container em segundo plano



Docker logs

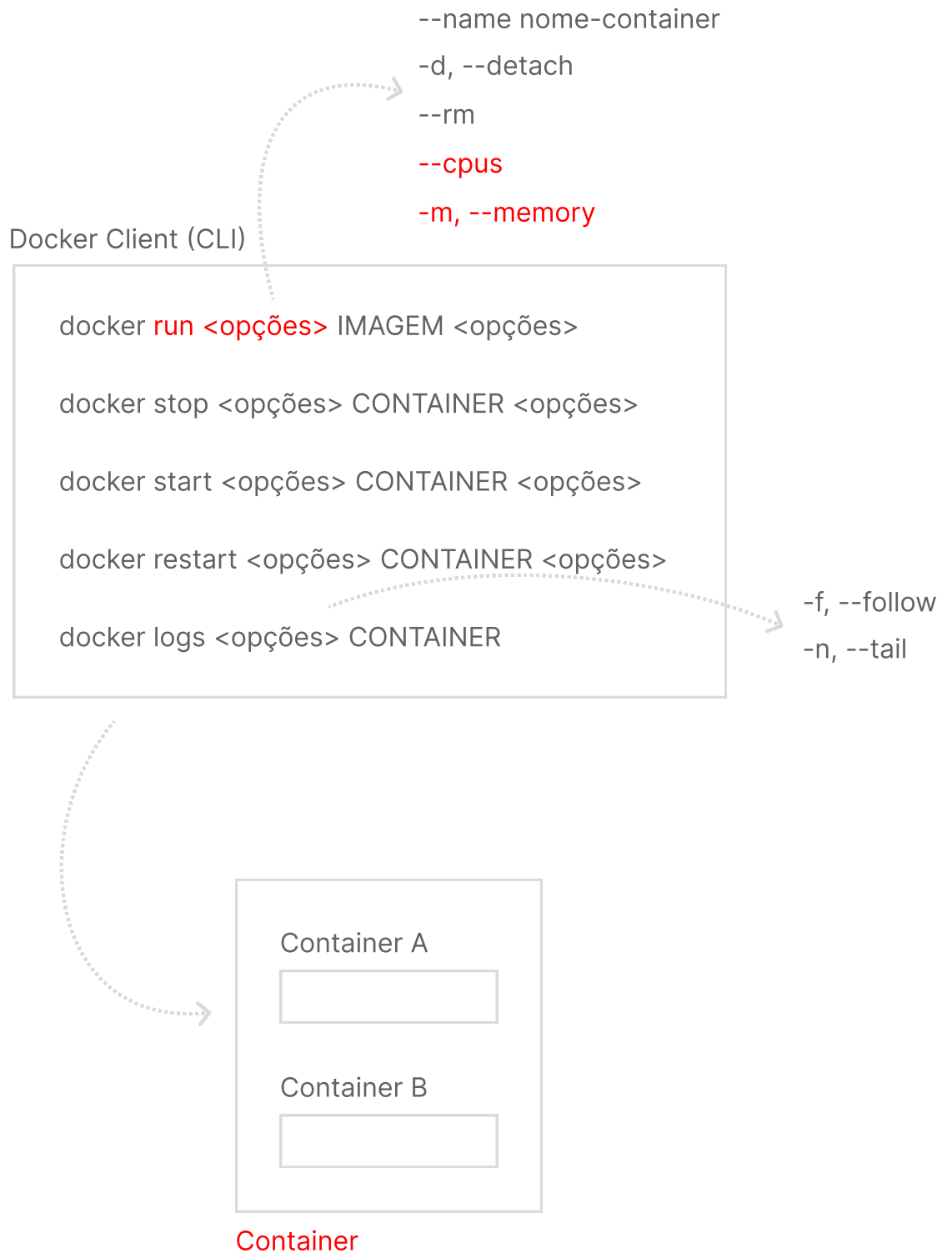


Removendo automaticamente o container após a execução

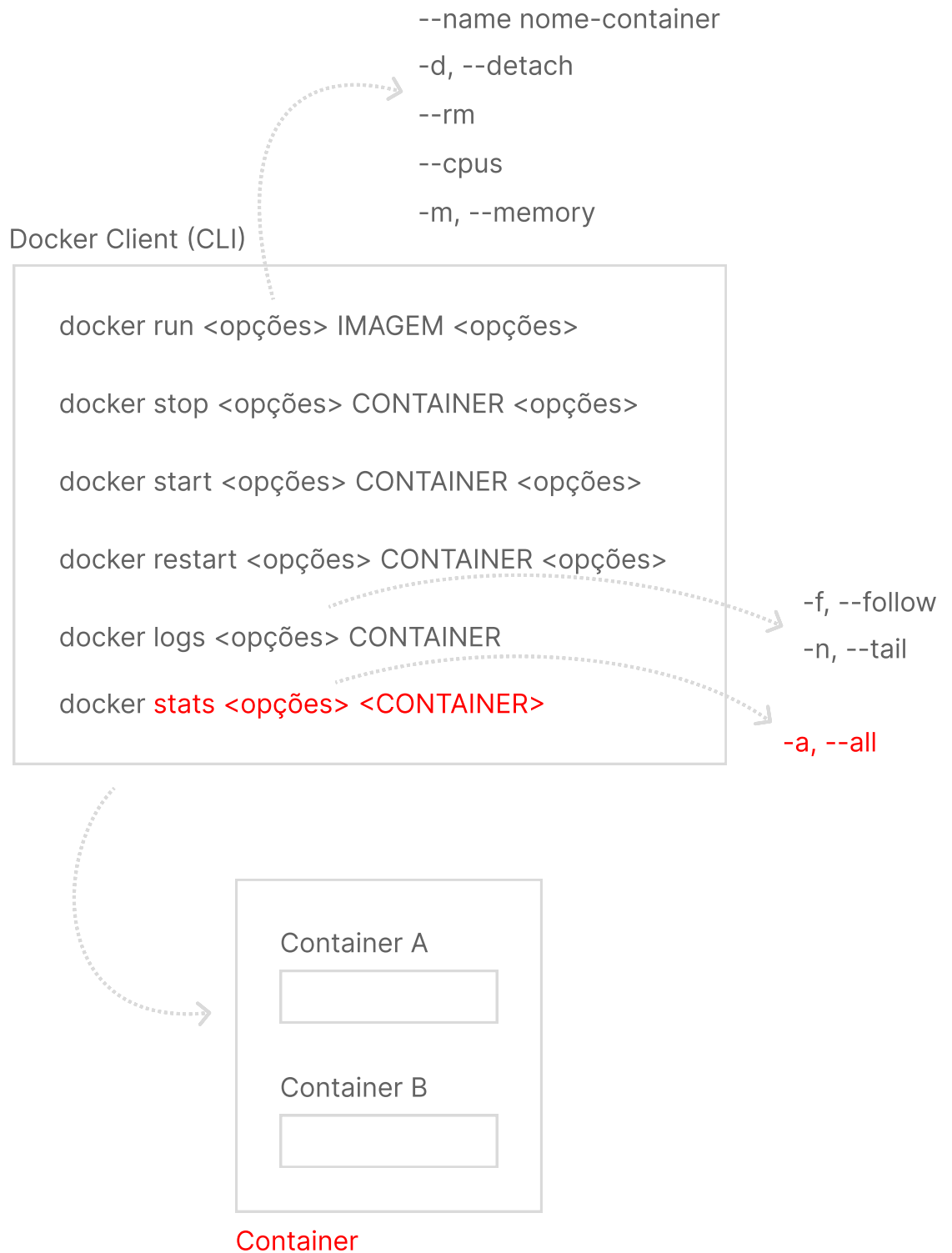


► Containers

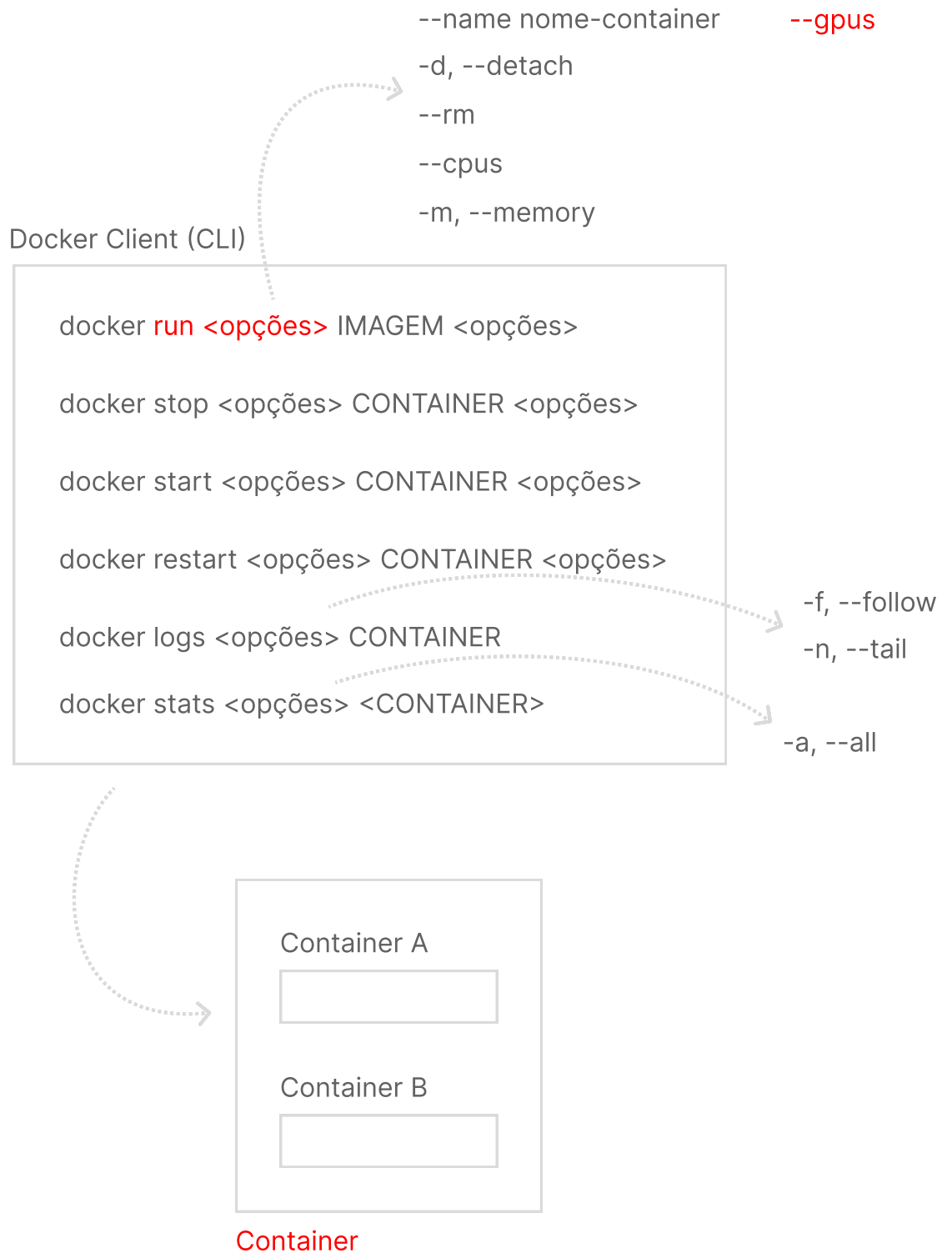
Definindo capacidade computacional (memória e cpu)



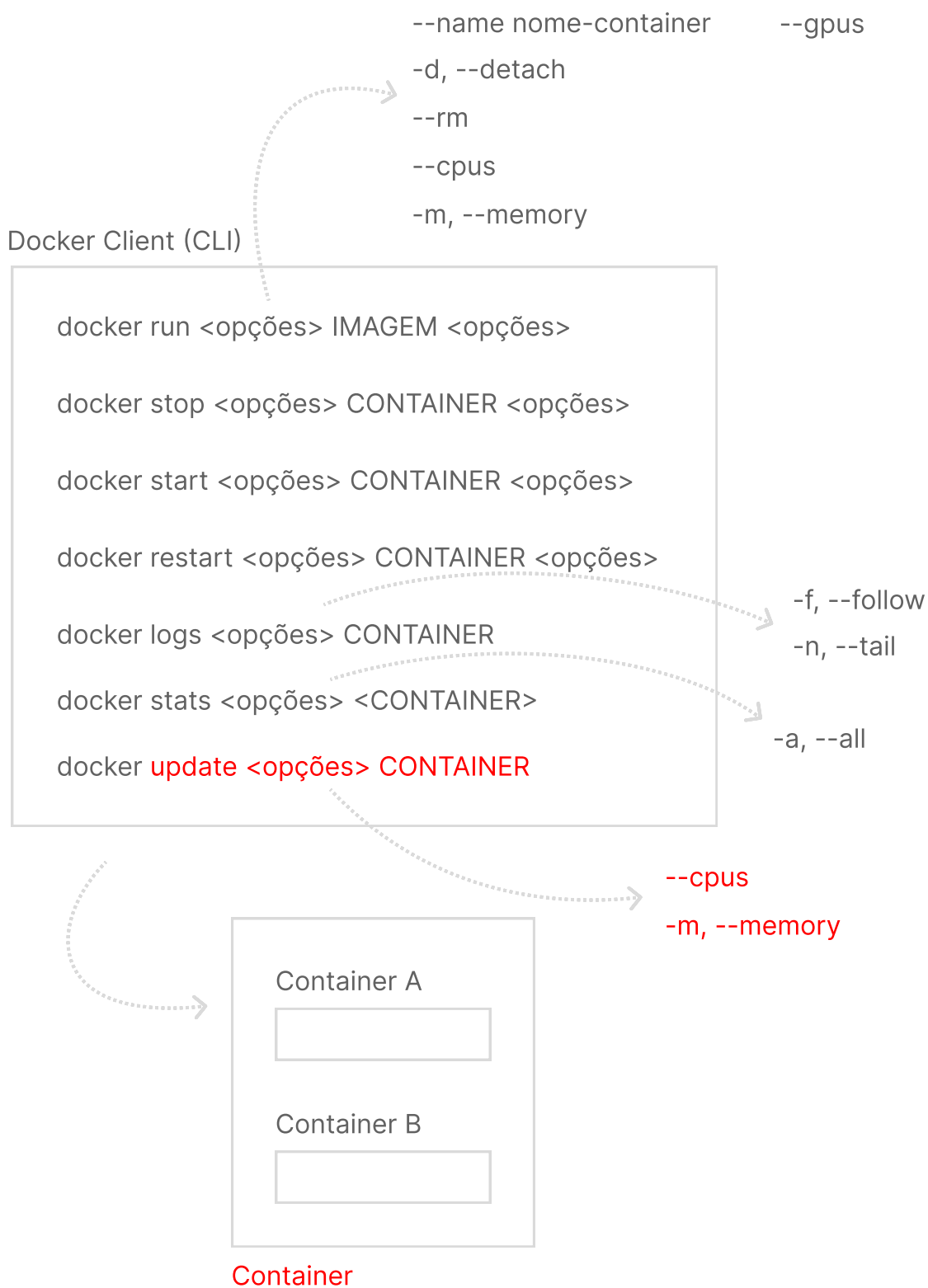
Verificando as estatísticas de um container



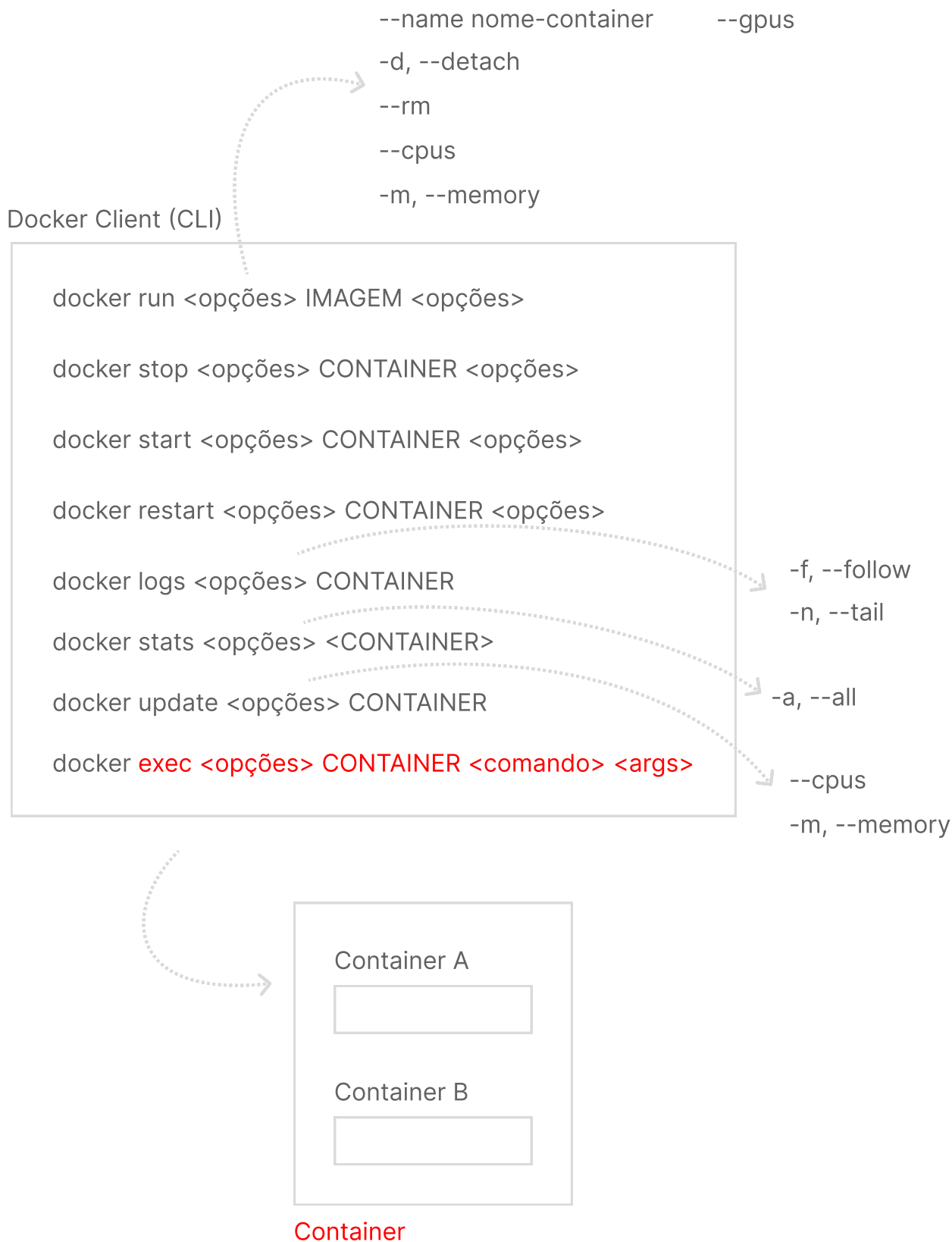
Habilitando o uso a GPUs



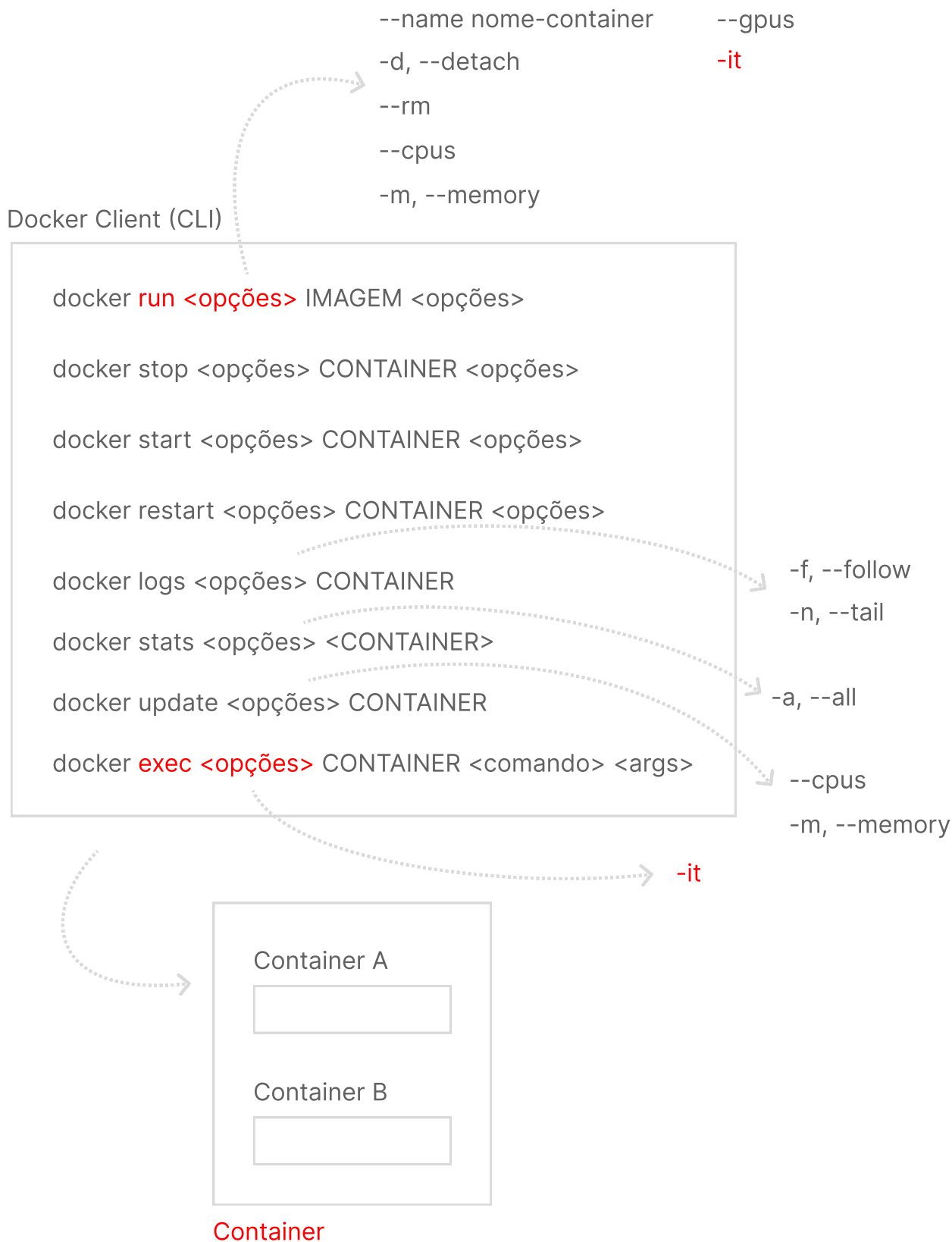
Atualizando a capacidade computacional de um container



Executando comandos em um container

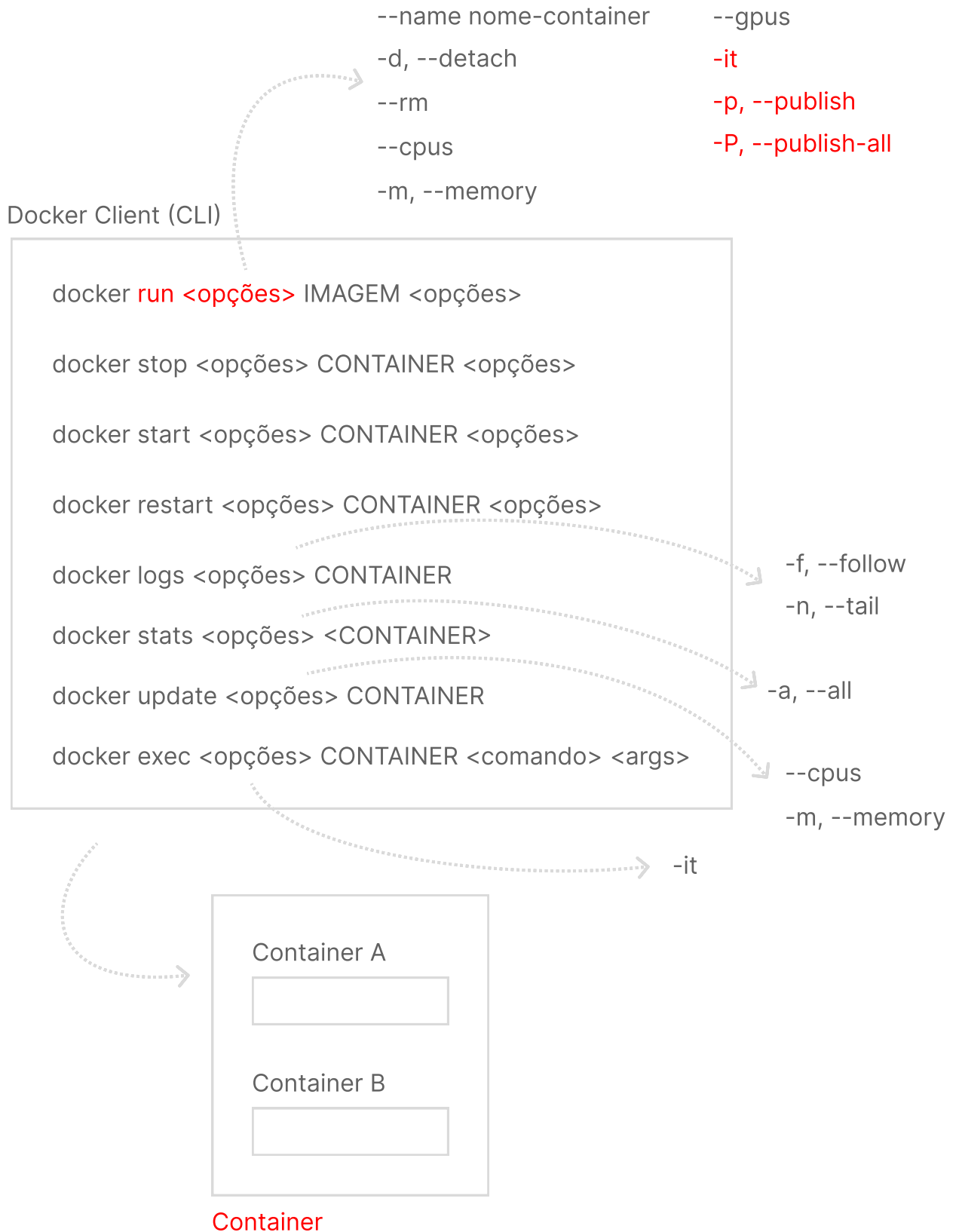


Acessando o terminal interativo



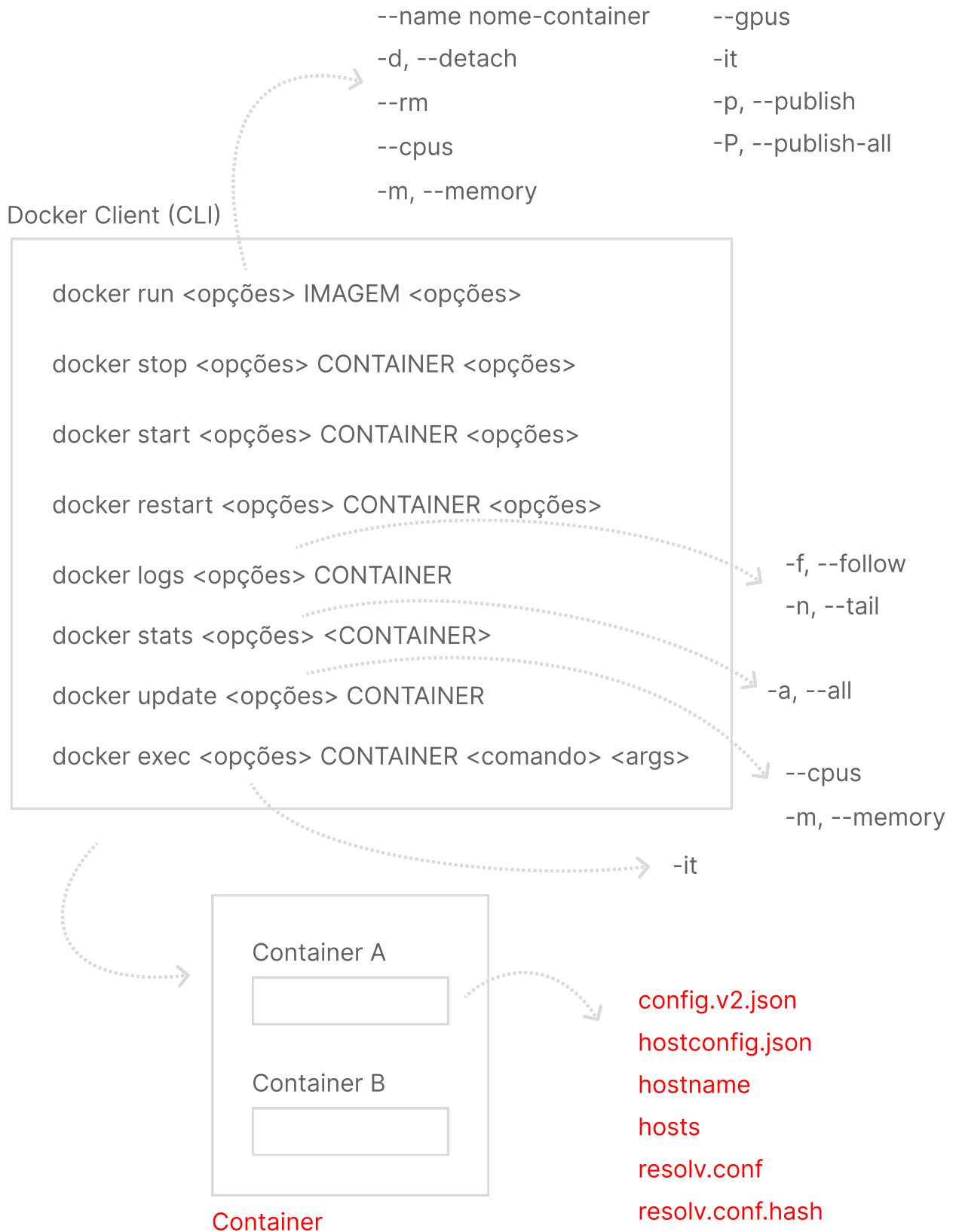
► Containers

Expondo portas

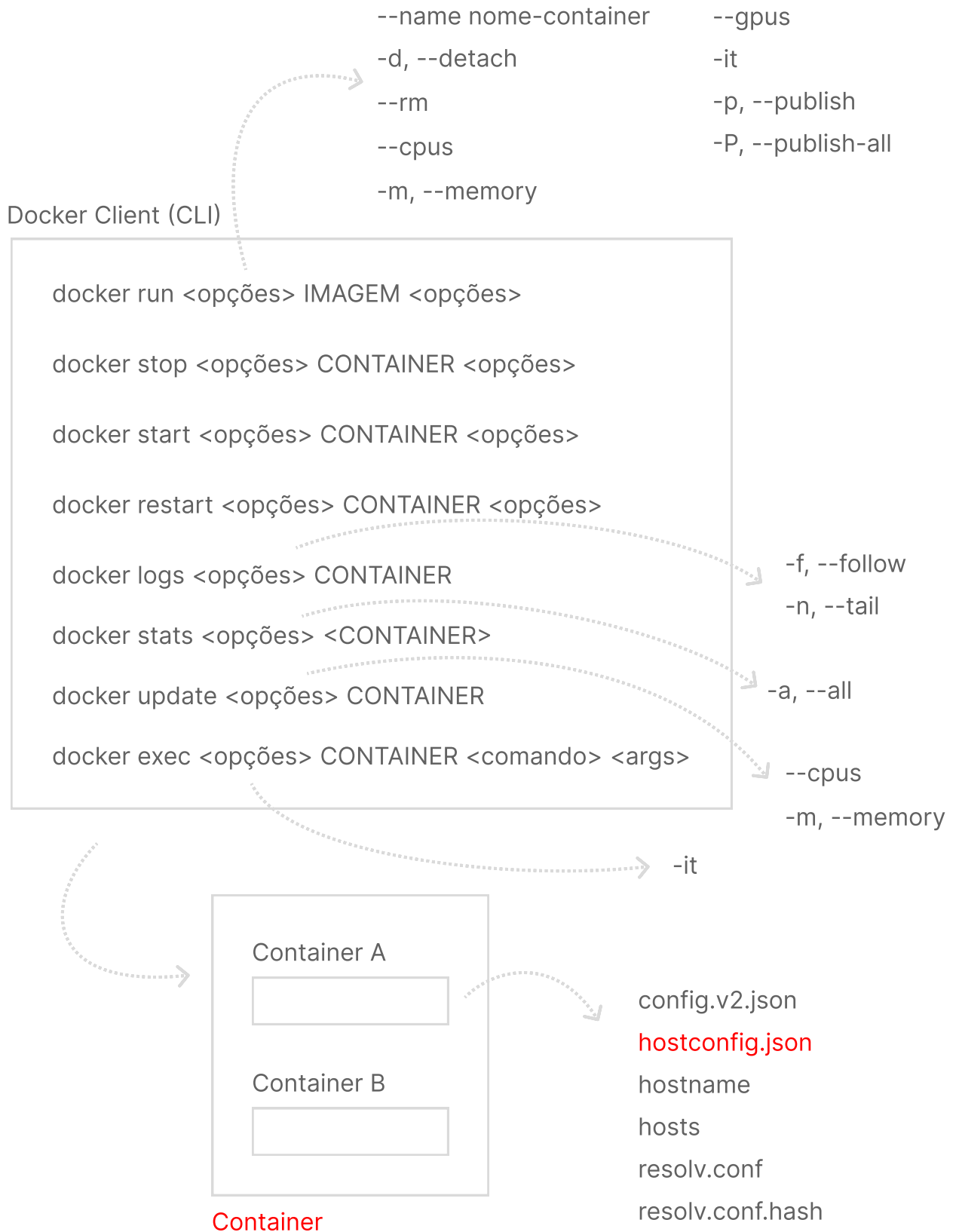


► Containers

Localização dos arquivos de configuração do container

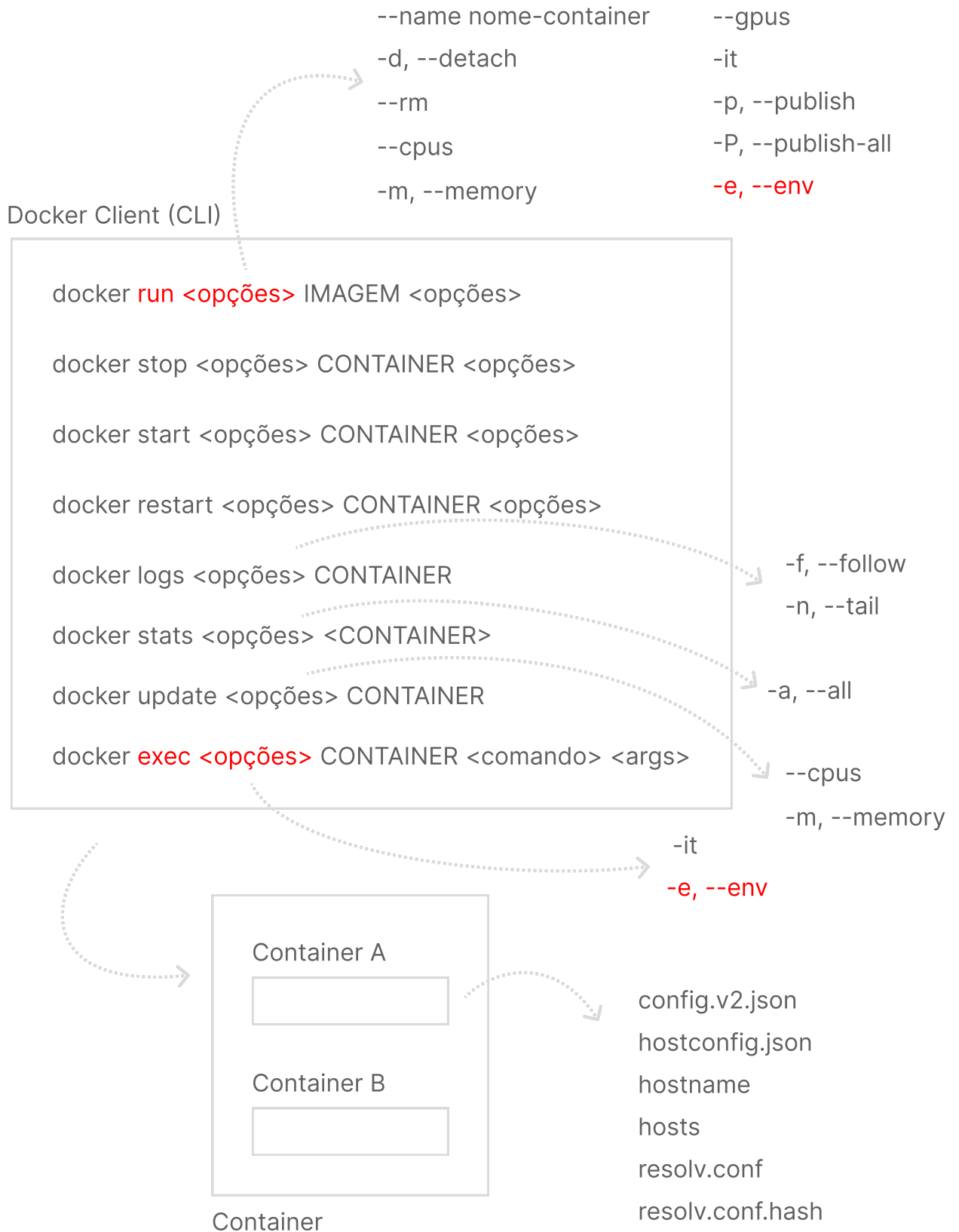


Alterando configurações do container via hostconfig



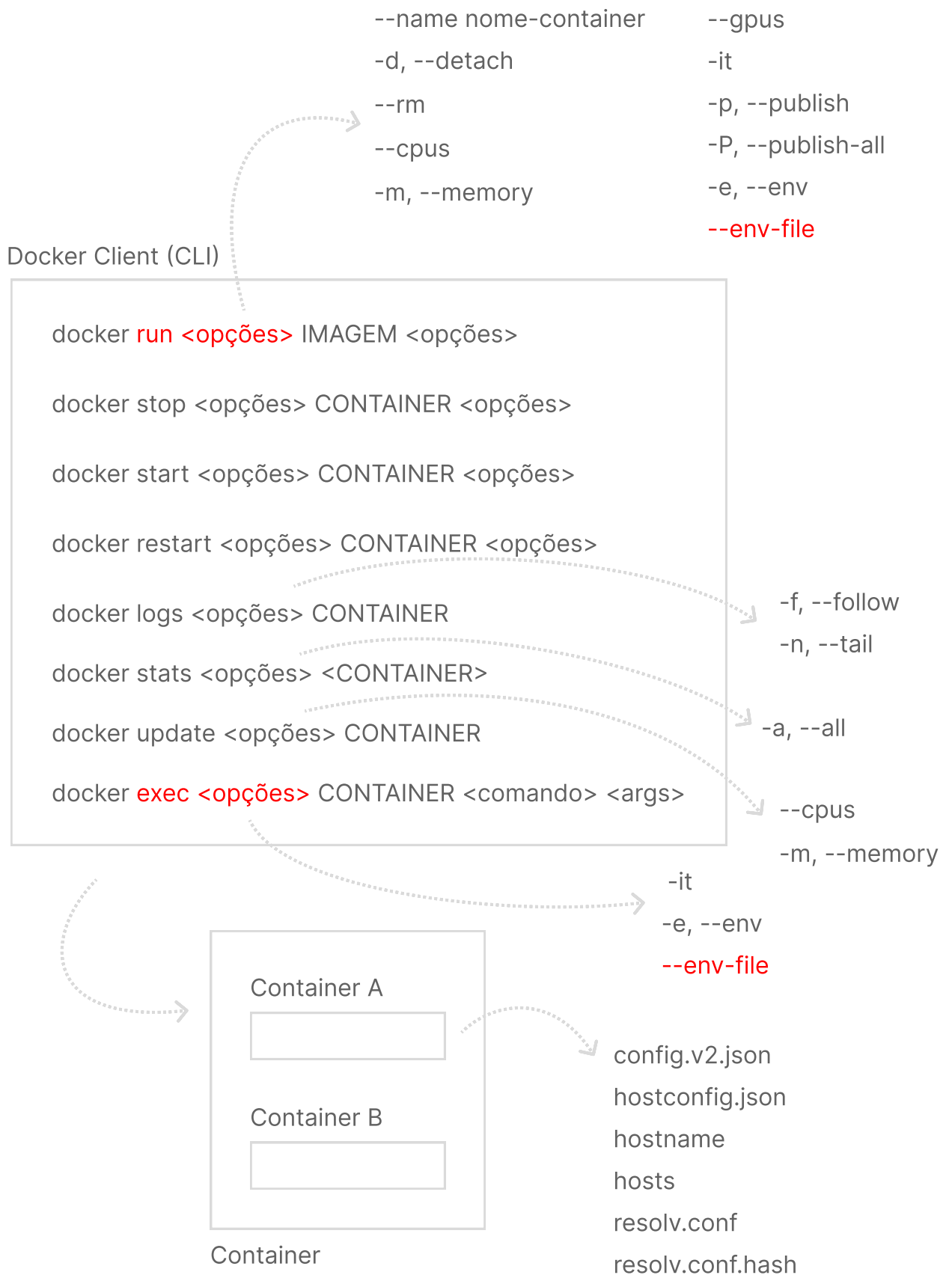
► Containers

Definindo variáveis de ambiente (via linha de comando)



► Containers

Definindo variáveis de ambiente (via arquivo externo)



► Containers

Transferindo arquivos do host para o container

--name nome-container -m, --memory -P, --publish-all
 -d, --detach --gpus -e, --env
 --rm -it --env-file
 --cpus -p, --publish

```
docker run <opções> IMAGEM <opções>

docker stop <opções> CONTAINER <opções>

docker start <opções> CONTAINER <opções>

docker restart <opções> CONTAINER <opções>

docker logs <opções> CONTAINER

docker stats <opções> <CONTAINER>

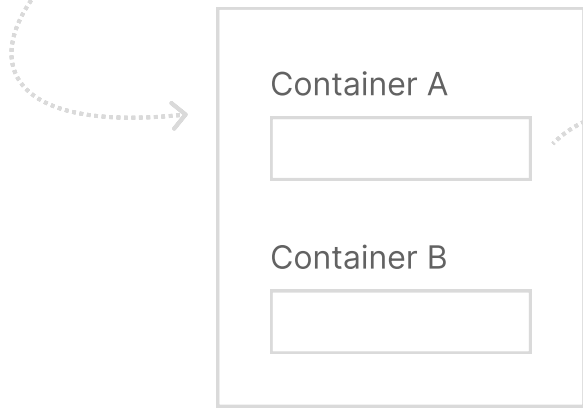
docker update <opções> CONTAINER

docker exec <opções> CONTAINER <comando> <args>

docker cp arq_host CONTAINER:dest_container
```

-f, --follow
 -n, --tail
 -a, --all
 --cpus
 -m, --memory
 -it
 -e, --env
 --env-file

Docker Client (CLI)



Container

config.v2.json
 hostconfig.json
 hostname
 hosts
 resolv.conf
 resolv.conf.hash

► Containers

Transferindo arquivos do container para o host

--name nome-container	-m, --memory	-P, --publish-all
-d, --detach	--gpus	-e, --env
--rm	-it	--env-file
--cpus	-p, --publish	

```

docker run <opções> IMAGEM <opções>

docker stop <opções> CONTAINER <opções>

docker start <opções> CONTAINER <opções>

docker restart <opções> CONTAINER <opções>

docker logs <opções> CONTAINER

docker stats <opções> <CONTAINER>

docker update <opções> CONTAINER

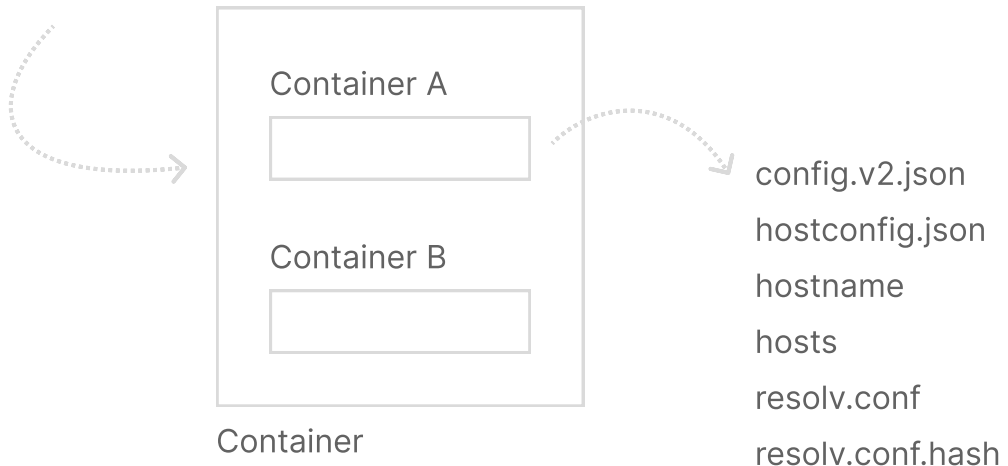
docker exec <opções> CONTAINER <comando> <args>

docker cp arq_host CONTAINER:dest_container

docker cp CONTAINER:arq_container dest_host
    
```

-f, --follow
 -n, --tail
 -a, --all
 --cpus
 -m, --memory
 -it
 -e, --env
 --env-file

Docker Client (CLI)



► Containers

Definindo o diretório de trabalho

- name nome-container -m, --memory -P, --publish-all
- d, --detach --gpu -e, --env
- rm -it --env-file
- cpus -p, --publish **-w, --workdir**

```

docker run <opções> IMAGEM <opções>

docker stop <opções> CONTAINER <opções>

docker start <opções> CONTAINER <opções>

docker restart <opções> CONTAINER <opções>

docker logs <opções> CONTAINER

docker stats <opções> <CONTAINER>

docker update <opções> CONTAINER

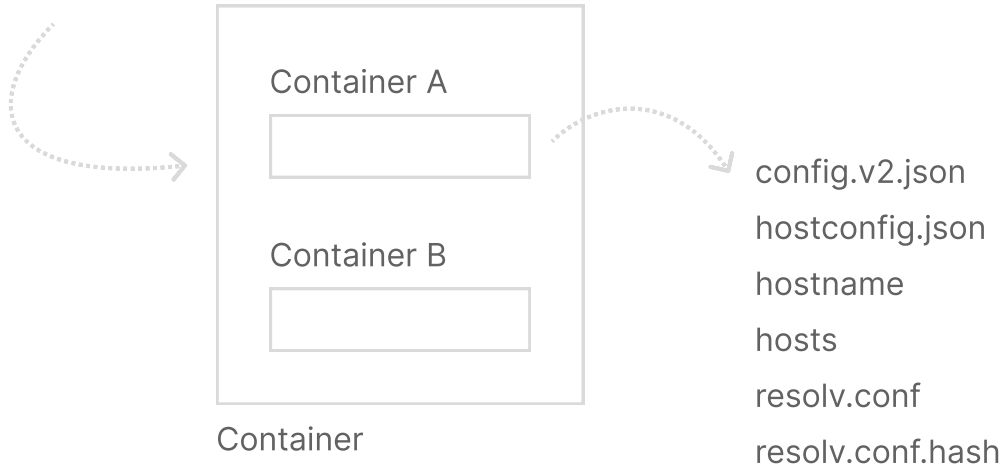
docker exec <opções> CONTAINER <comando> <args>

docker cp arq_host CONTAINER:dest_container

docker cp CONTAINER:arq_container dest_host
    
```

- f, --follow
- n, --tail
- a, --all
- cpus
- m, --memory
- it
- e, --env
- env-file

Docker Client (CLI)



► Containers

Mapeando volumes parte 1

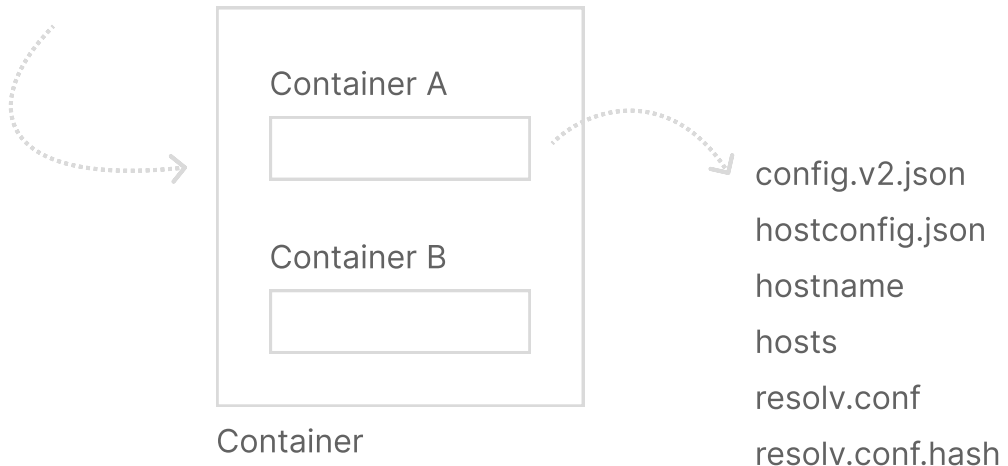
--name nome-container	-m, --memory	-P, --publish-all	-v
-d, --detach	--gpus	-e, --env	
--rm	-it	--env-file	
--cpus	-p, --publish	-w, --workdir	

```

docker run <opções> IMAGEM <opções>
docker stop <opções> CONTAINER <opções>
docker start <opções> CONTAINER <opções>
docker restart <opções> CONTAINER <opções>
docker logs <opções> CONTAINER
docker stats <opções> <CONTAINER>
docker update <opções> CONTAINER
docker exec <opções> CONTAINER <comando> <args>
docker cp arq_host CONTAINER:dest_container
docker cp CONTAINER:arq_container dest_host
    
```

-f, --follow
 -n, --tail
 -a, --all
 --cpus
 -m, --memory
 -it
 -e, --env
 --env-file

Docker Client (CLI)



► Containers

Mapeando volumes parte 2

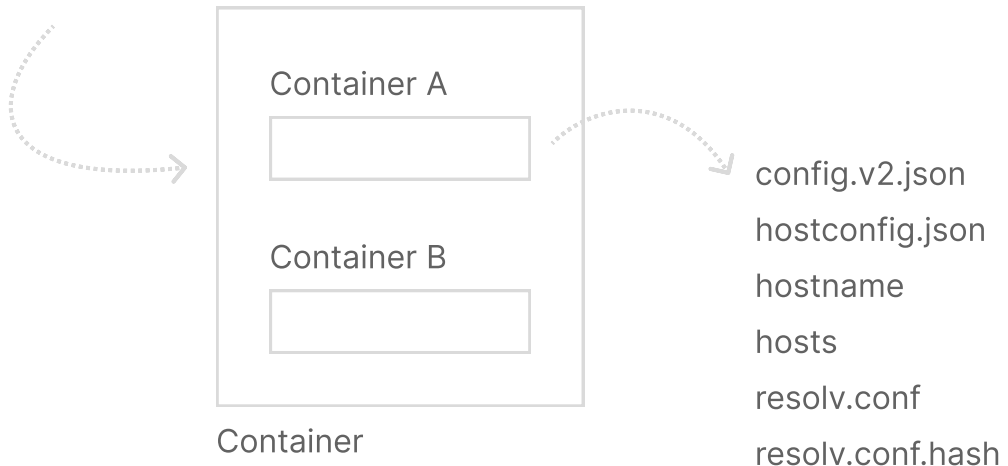
--name nome-container	-m, --memory	-P, --publish-all	-v
-d, --detach	--gpus	-e, --env	--mount
--rm	-it	--env-file	
--cpus	-p, --publish	-w, --workdir	

```

docker run <opções> IMAGEM <opções>
docker stop <opções> CONTAINER <opções>
docker start <opções> CONTAINER <opções>
docker restart <opções> CONTAINER <opções>
docker logs <opções> CONTAINER
docker stats <opções> <CONTAINER>
docker update <opções> CONTAINER
docker exec <opções> CONTAINER <comando> <args>
docker cp arq_host CONTAINER:dest_container
docker cp CONTAINER:arq_container dest_host
    
```

-f, --follow
 -n, --tail
 -a, --all
 --cpus
 -m, --memory
 -it
 -e, --env
 --env-file

Docker Client (CLI)



► Containers

Inspecione diferenças do file system de um container em relação a imagem

--name nome-container	-m, --memory	-P, --publish-all	-v
-d, --detach	--gpus	-e, --env	--mount
--rm	-it	--env-file	
--cpus	-p, --publish	-w, --workdir	

```

docker run <opções> IMAGEM <opções>

docker stop <opções> CONTAINER <opções>

docker start <opções> CONTAINER <opções>

docker restart <opções> CONTAINER <opções>

docker logs <opções> CONTAINER

docker stats <opções> <CONTAINER>

docker update <opções> CONTAINER

docker exec <opções> CONTAINER <comando> <args>

docker cp arq_host CONTAINER:dest_container

docker cp CONTAINER:arq_container dest_host

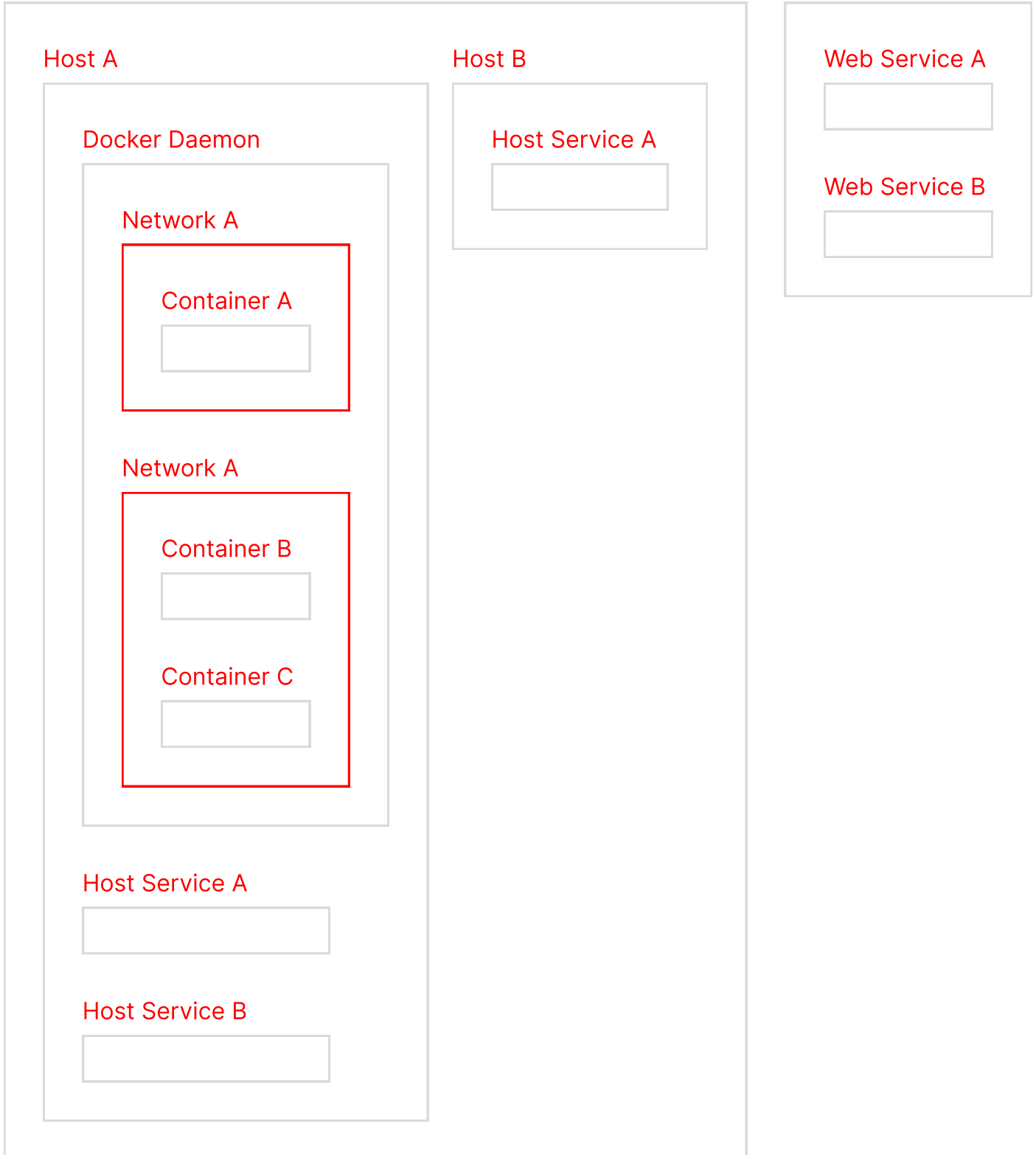
docker diff CONTAINER
    
```

-f, --follow
 -n, --tail
 -a, --all
 --cpus
 -m, --memory
 -it
 -e, --env
 --env-file

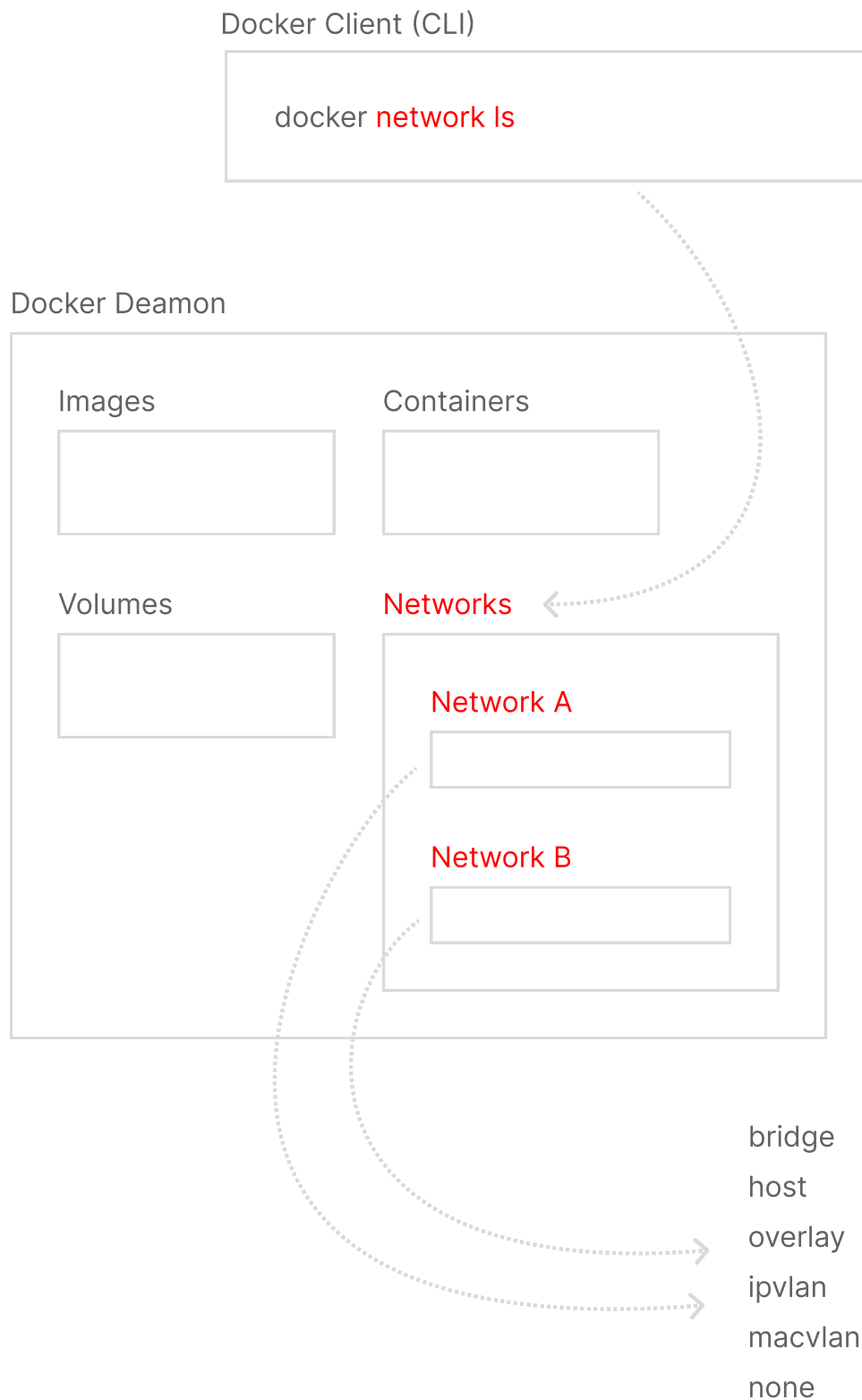
O que são networks

LAN (Rede Local)

WWW

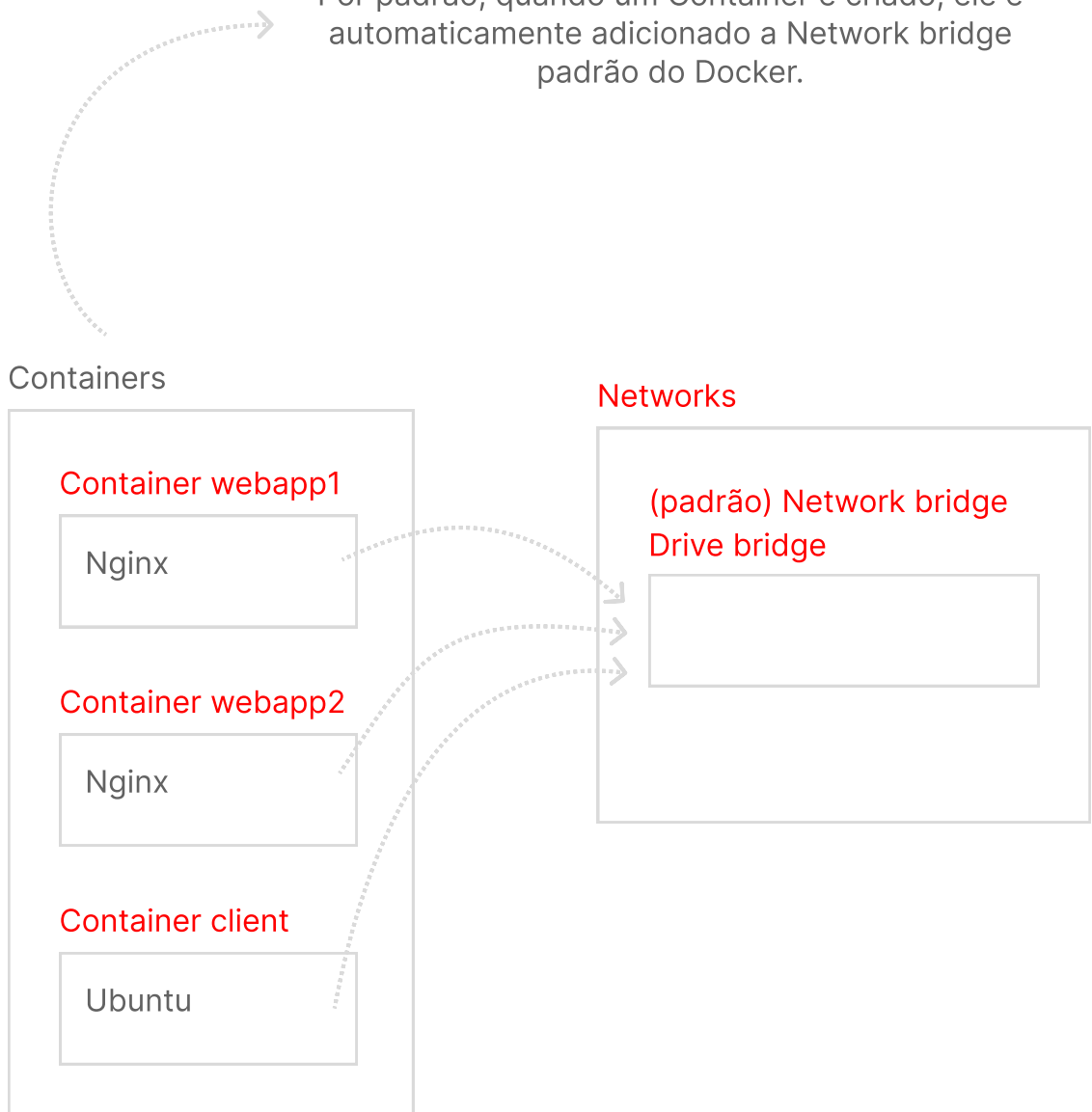


Networks e tipos de drivers

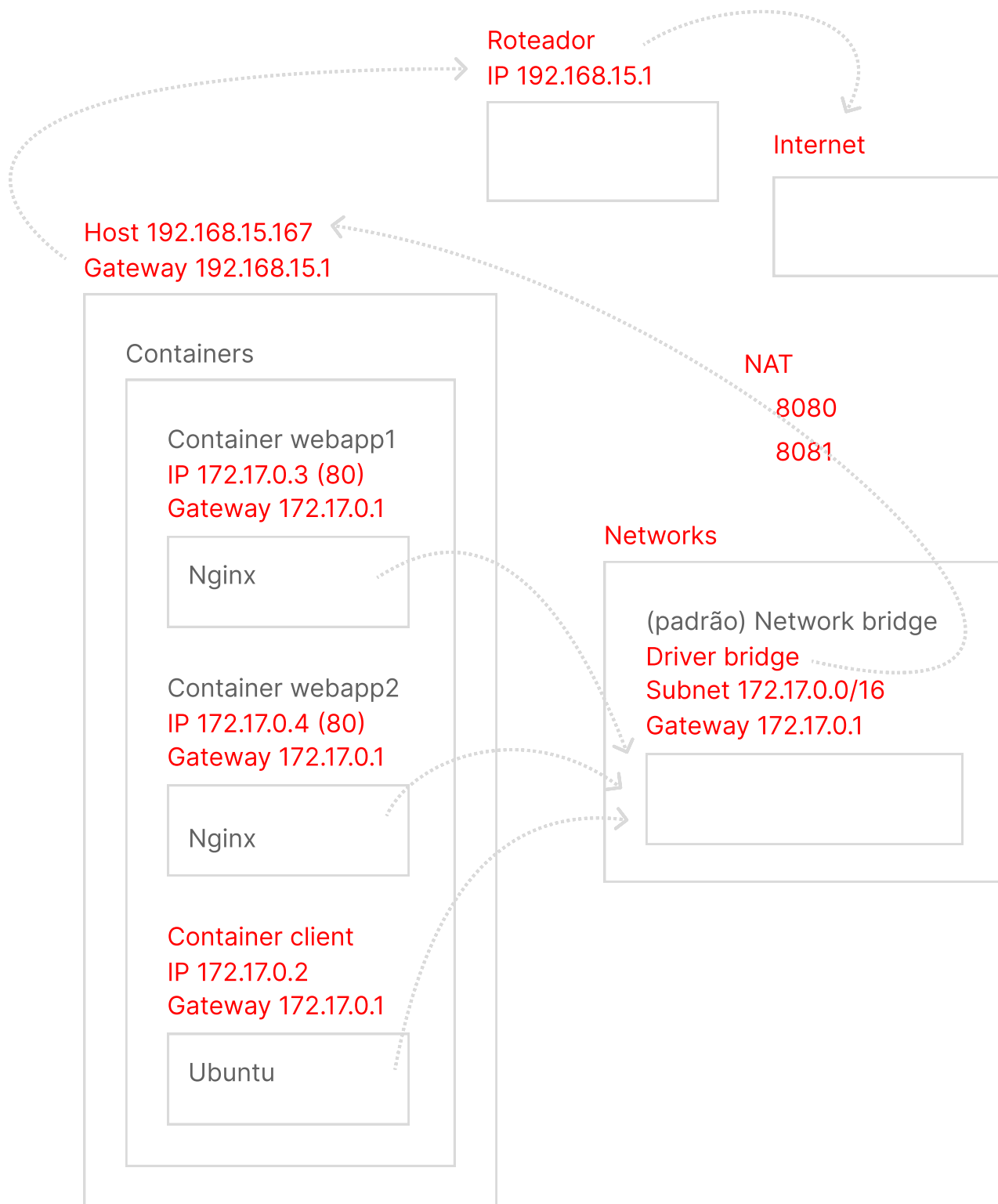


Analisando o comportamento padrão de rede do Docker (bridge) parte 1

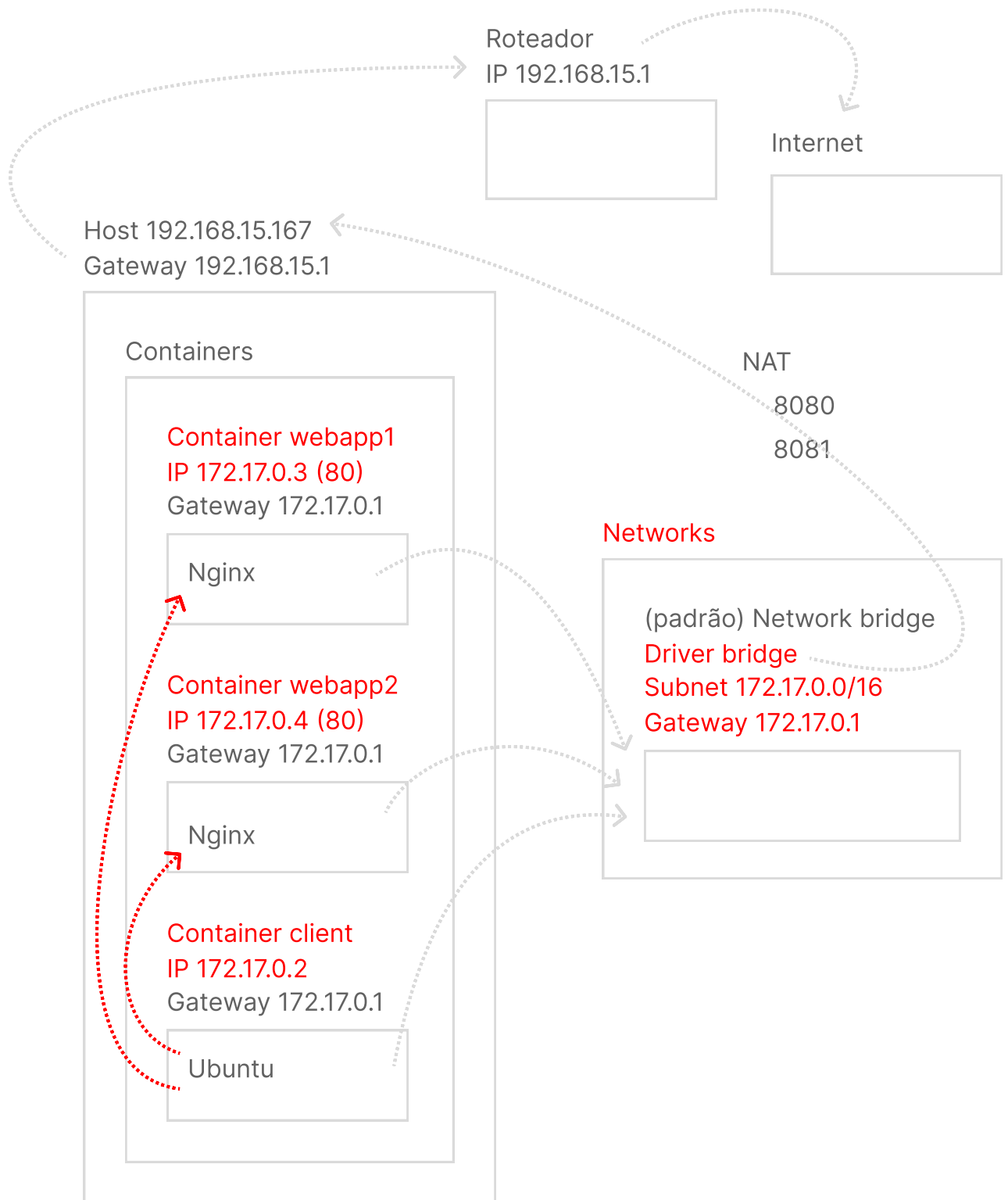
Por padrão, quando um Container é criado, ele é automaticamente adicionado a Network bridge padrão do Docker.



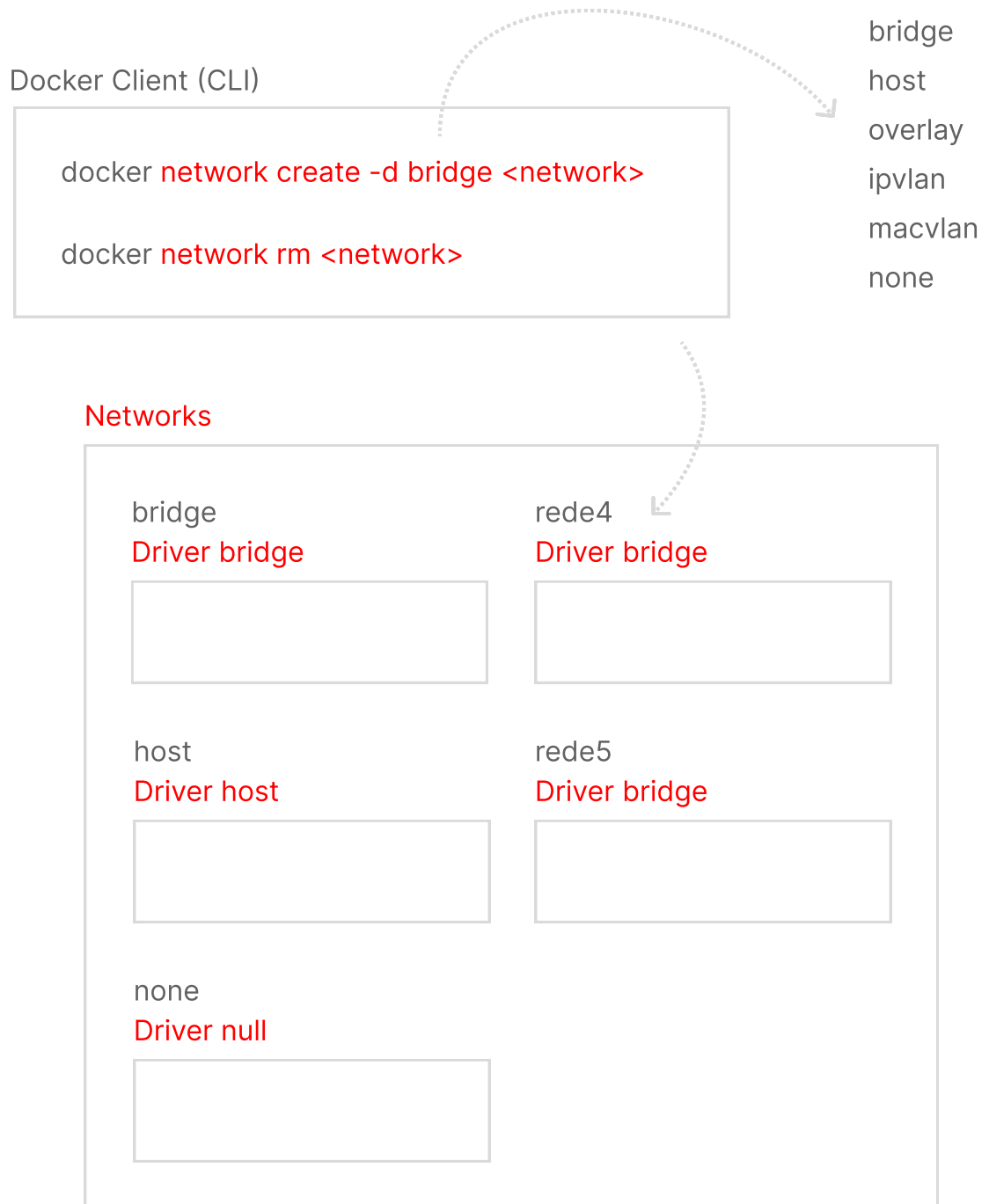
Analisando o comportamento padrão de rede do Docker (bridge) parte 2



Analisando o comportamento padrão de rede do Docker (bridge) parte 3



Criando e removendo networks

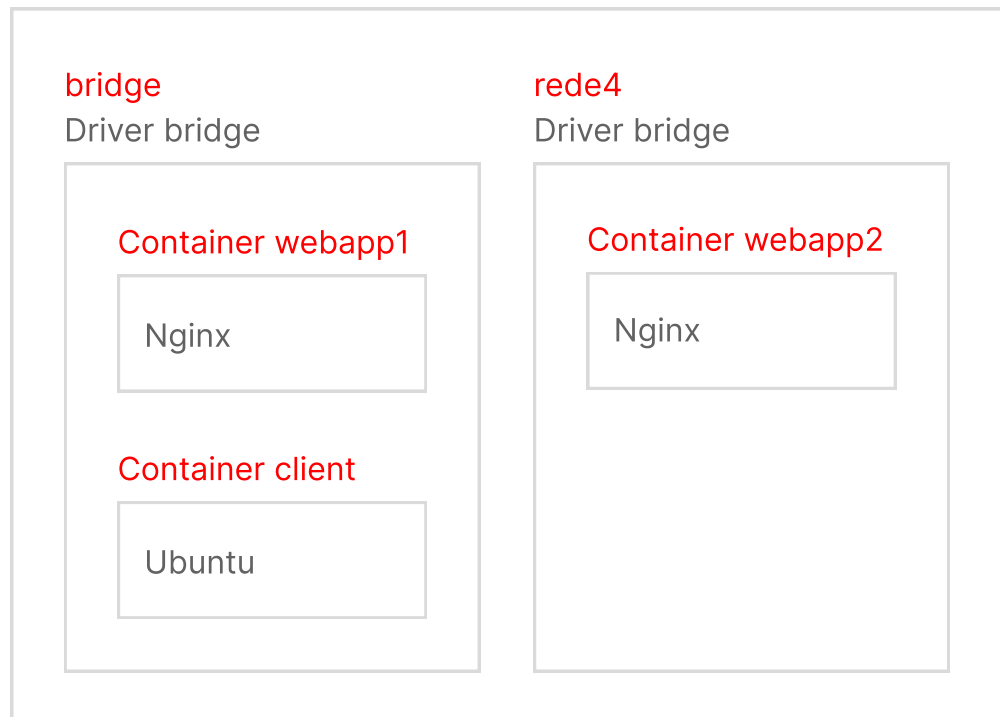


Desconectando e conectando um container a uma network

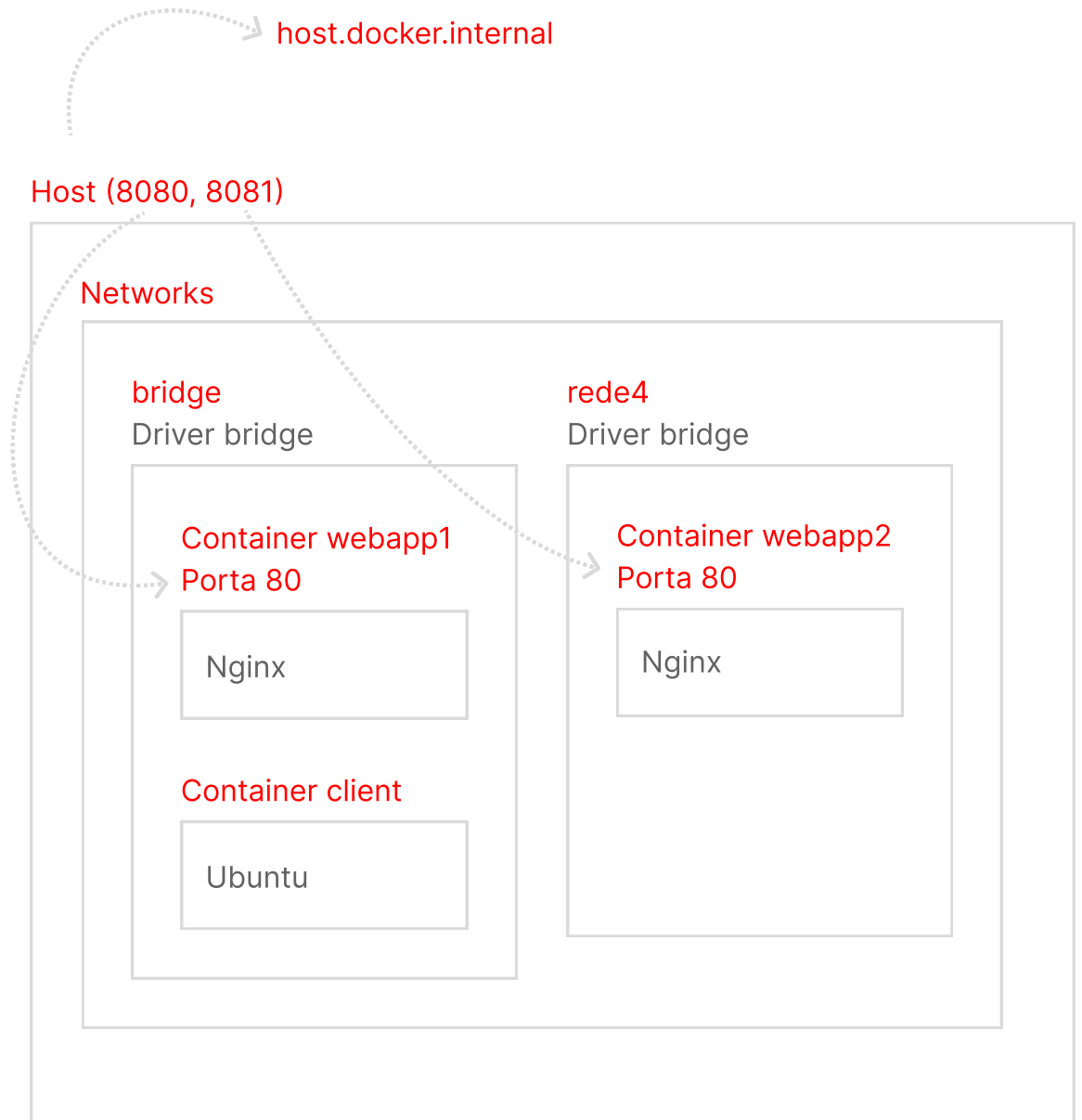
Docker Client (CLI)

```
docker network disconnect <network> <container>  
docker network connect <network> <container>
```

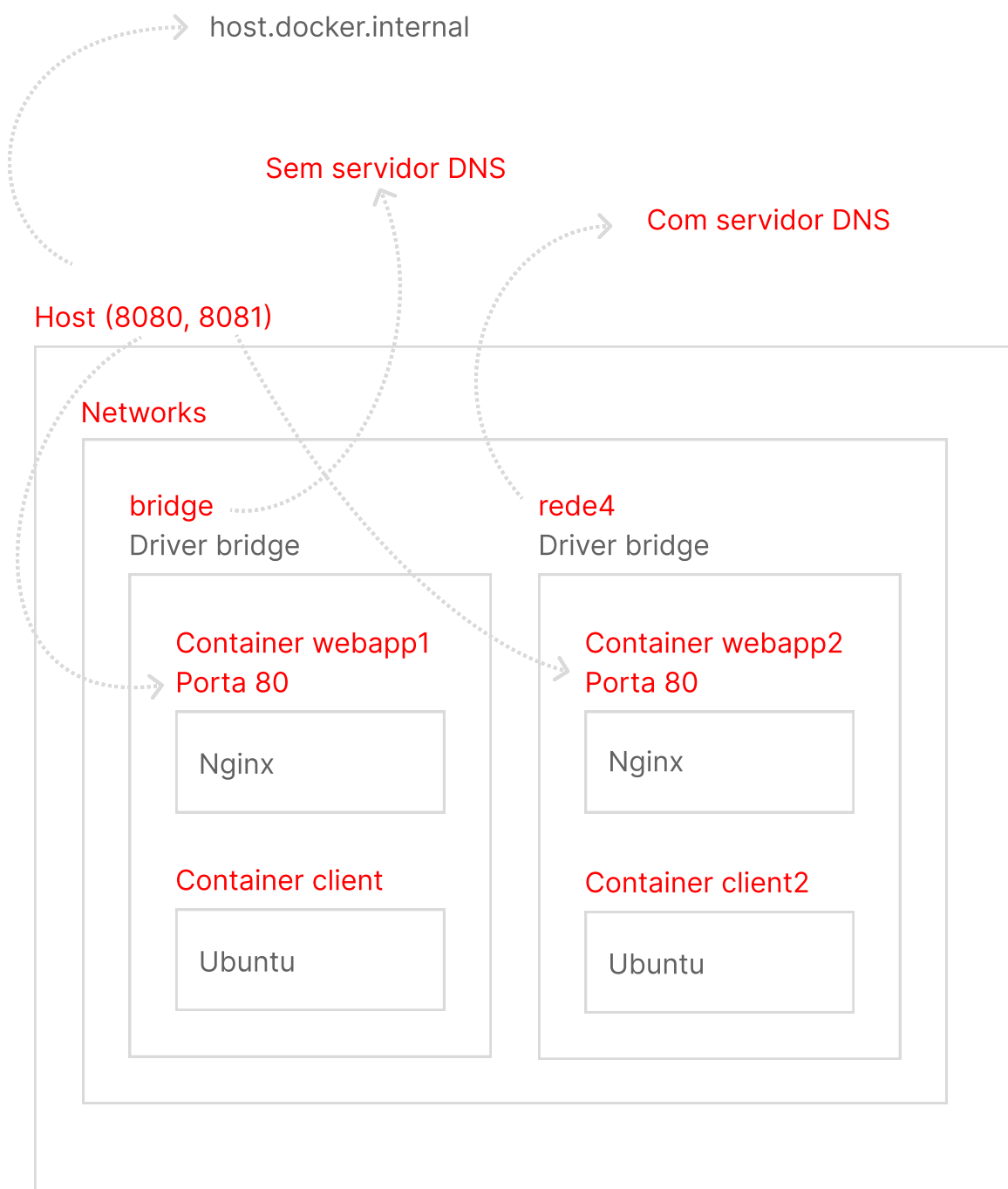
Networks



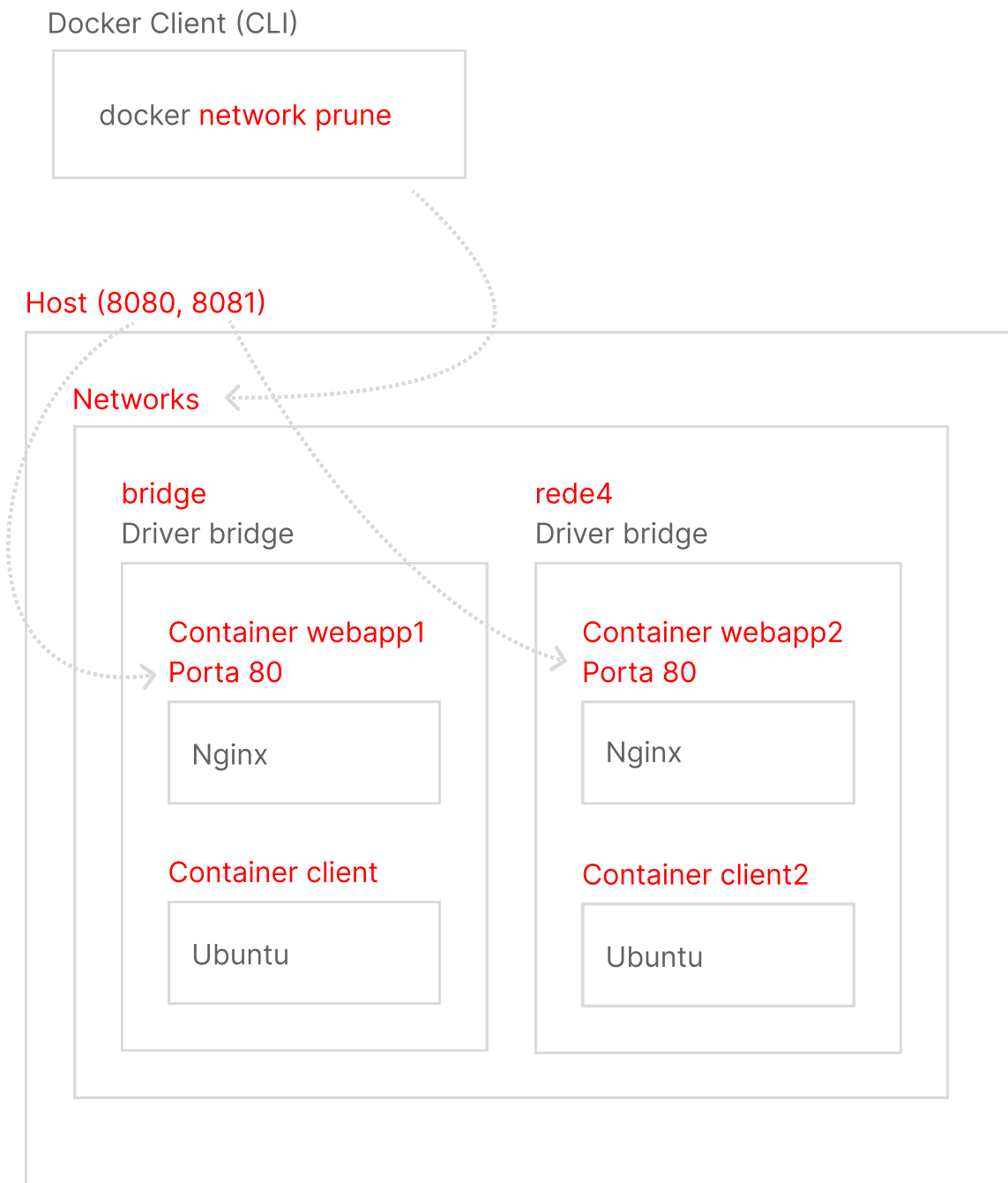
Comunicação entre containers de redes bridges diferentes via host



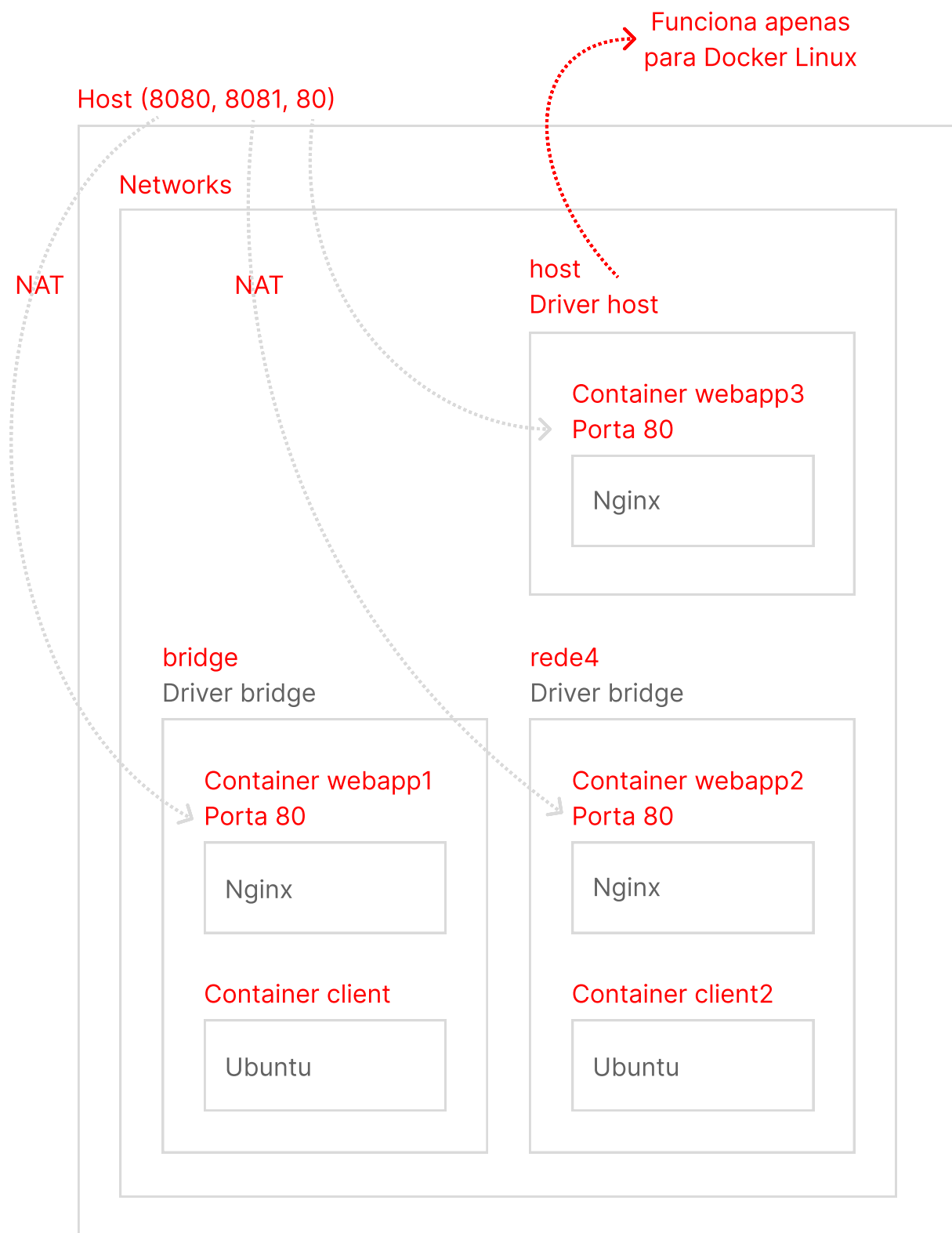
Analisando o comportamento padrão de rede do Docker (bridge) parte 4



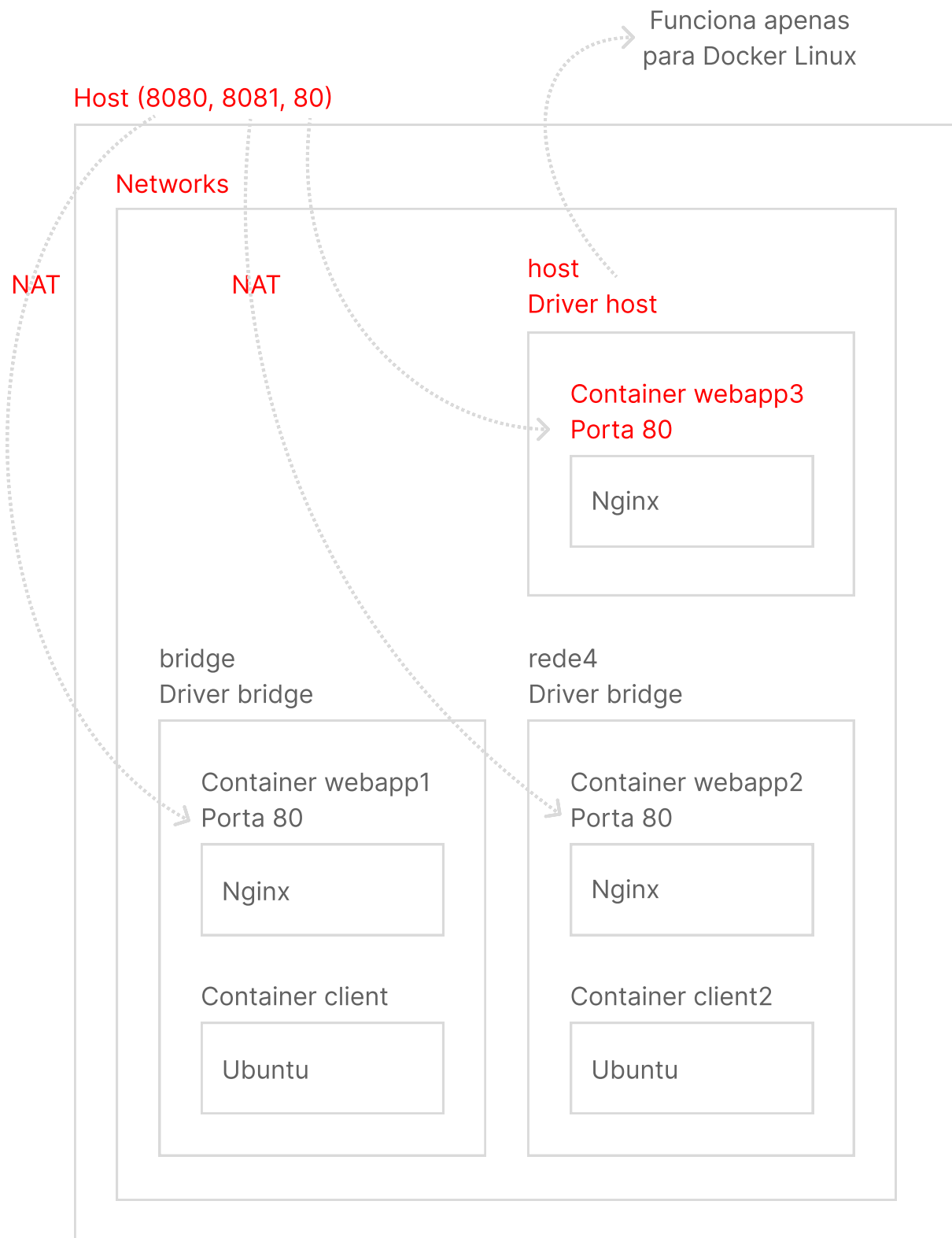
Removendo networks não utilizadas



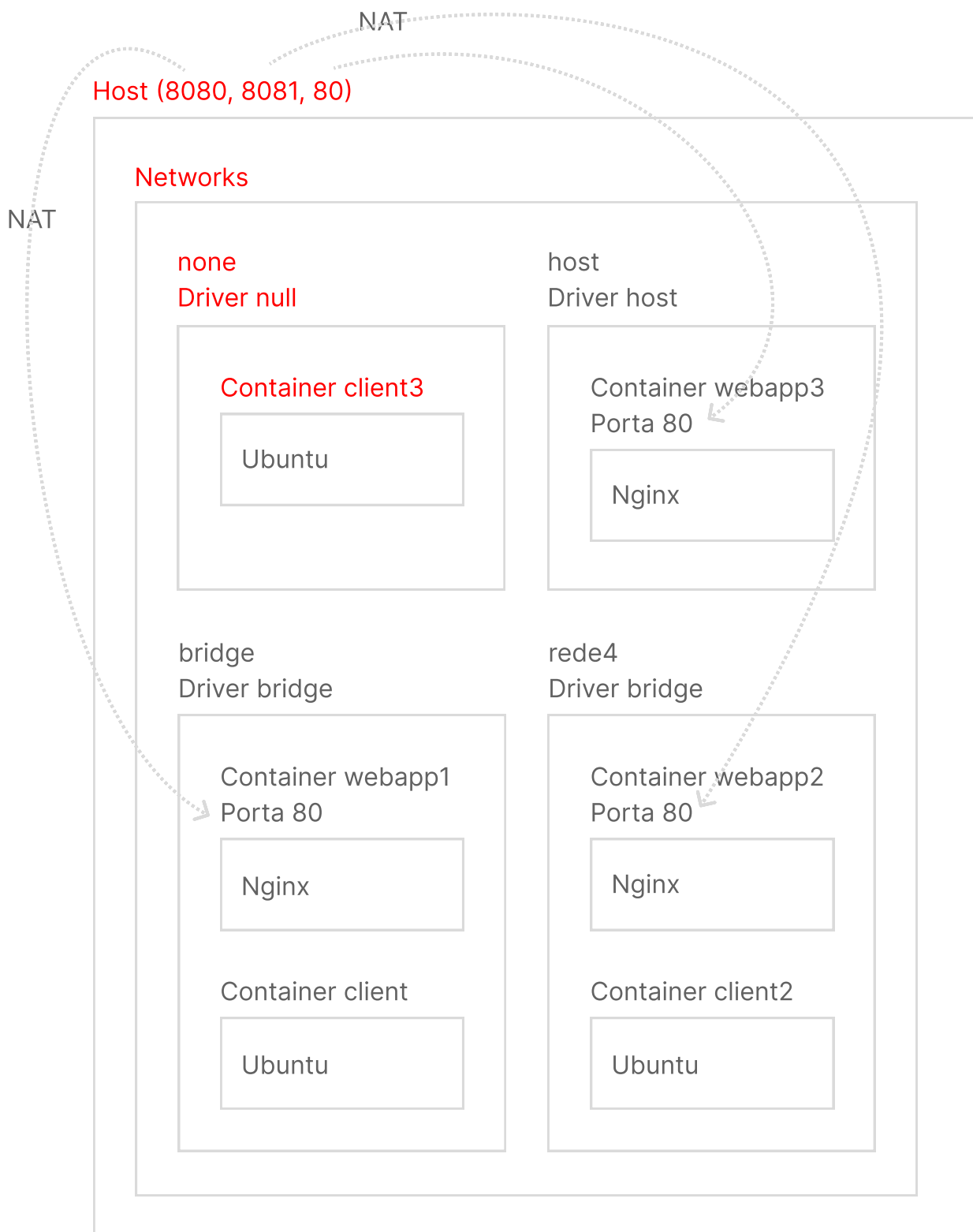
Comportamento do driver host



Vantagens na utilização de uma network com driver host

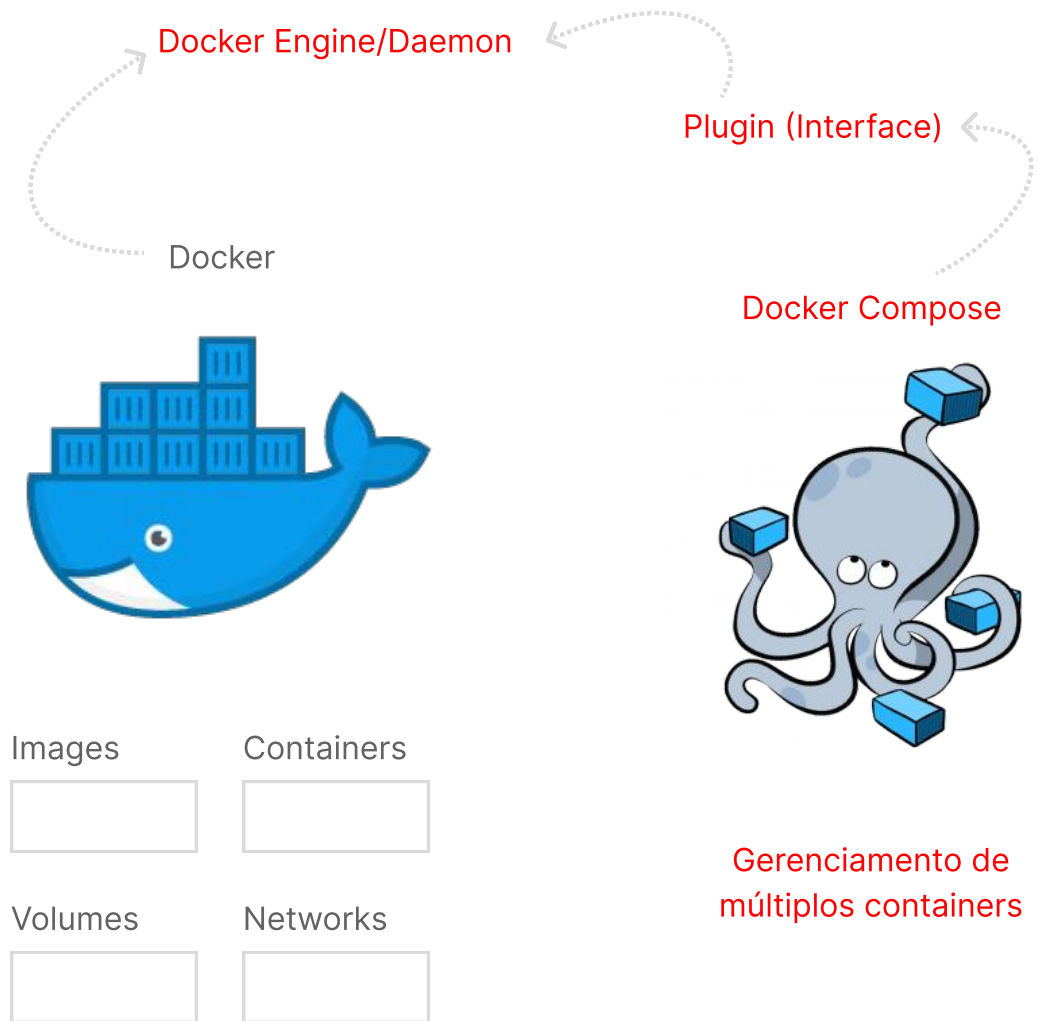


Comportamento do driver null



► Docker Compose

O que é o Docker Compose?



► Docker Compose

Configurando o ambiente com Nginx, Next, Nest, PostgreSQL, Mongo, Redis e Kafka

Servidor HTTP



Nginx, Apache

Front-end



Next, React, Vue, Angular

Back-end



Nest, PHP, GoLang

Banco de Dados Relacional



PostgreSQL, MySQL, MariaDB, SQL Server, Oracle

Banco de Dados Não Relacional (NoSQL)



MongoDB, DynamoDB, Couchbase, ElasticSearch

Cache de Memória



Redis, Memcached

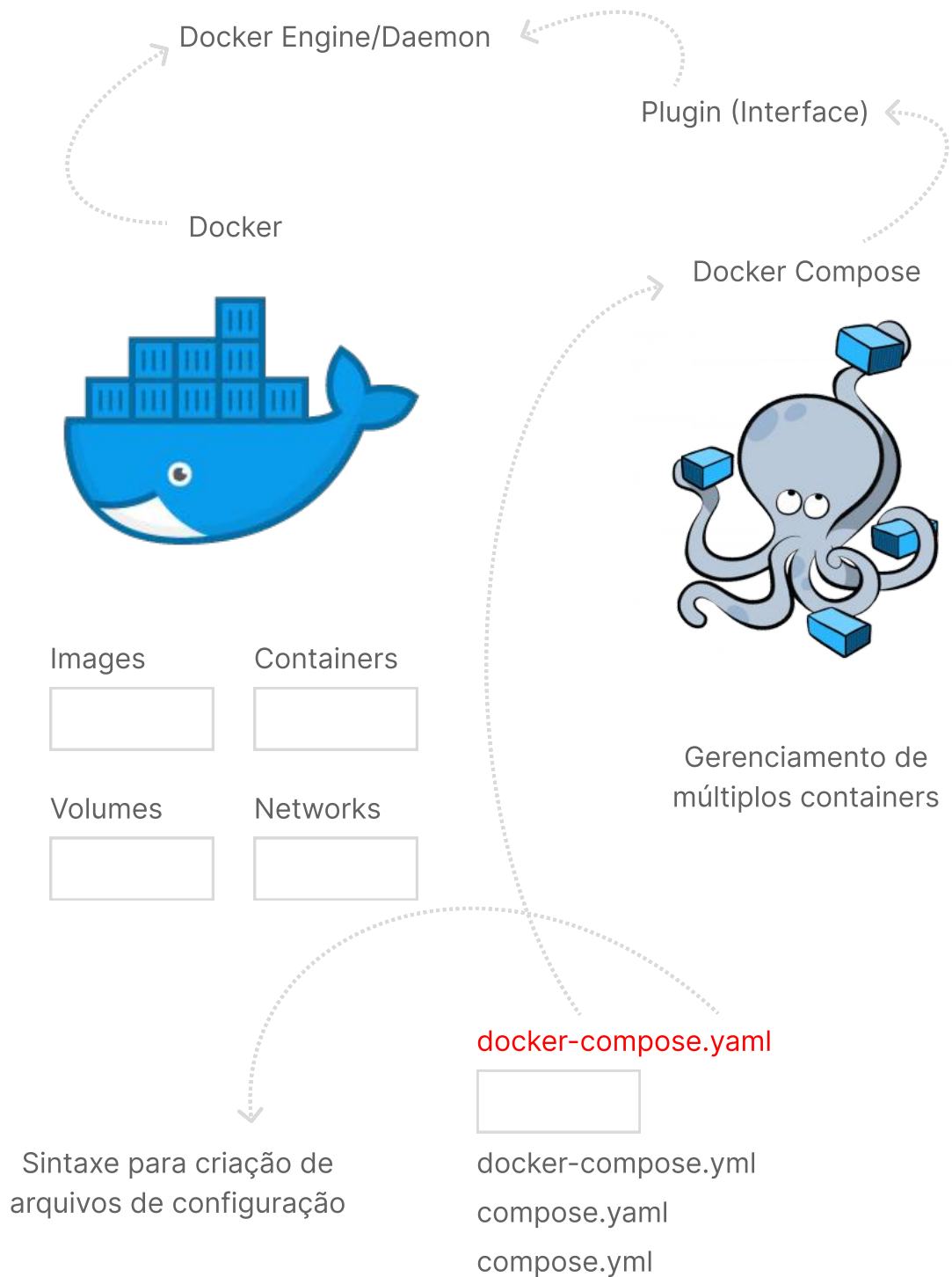
Mensageria



Kafka, RabbitMQ

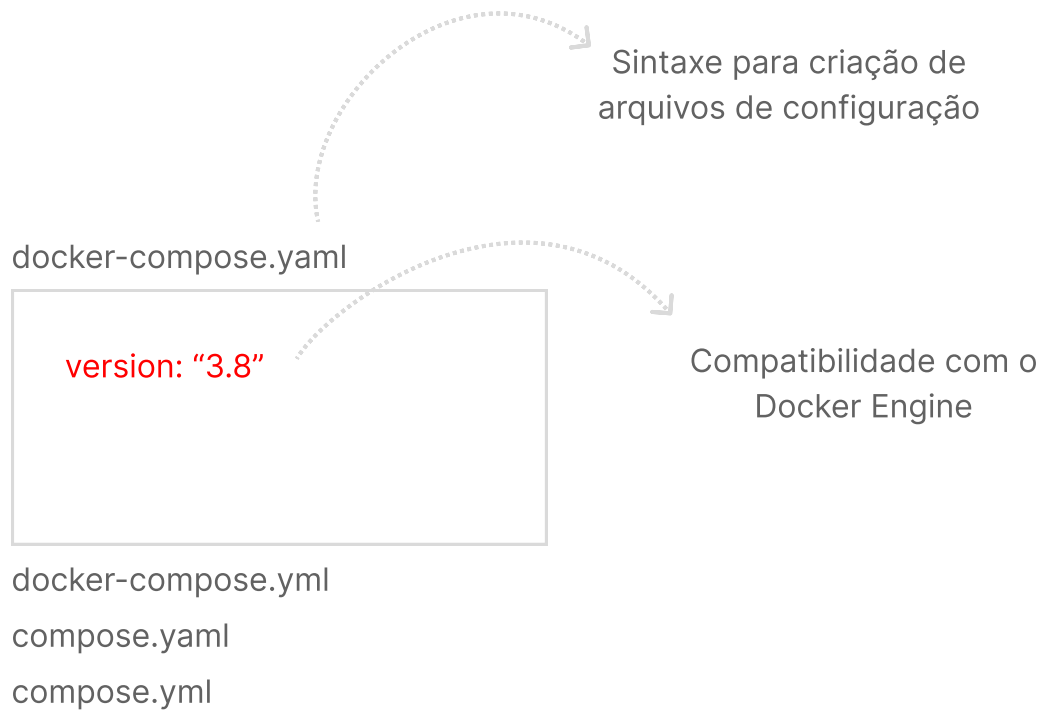
► Docker Compose

Conhecendo o arquivo docker-compose e a sintaxe YAML

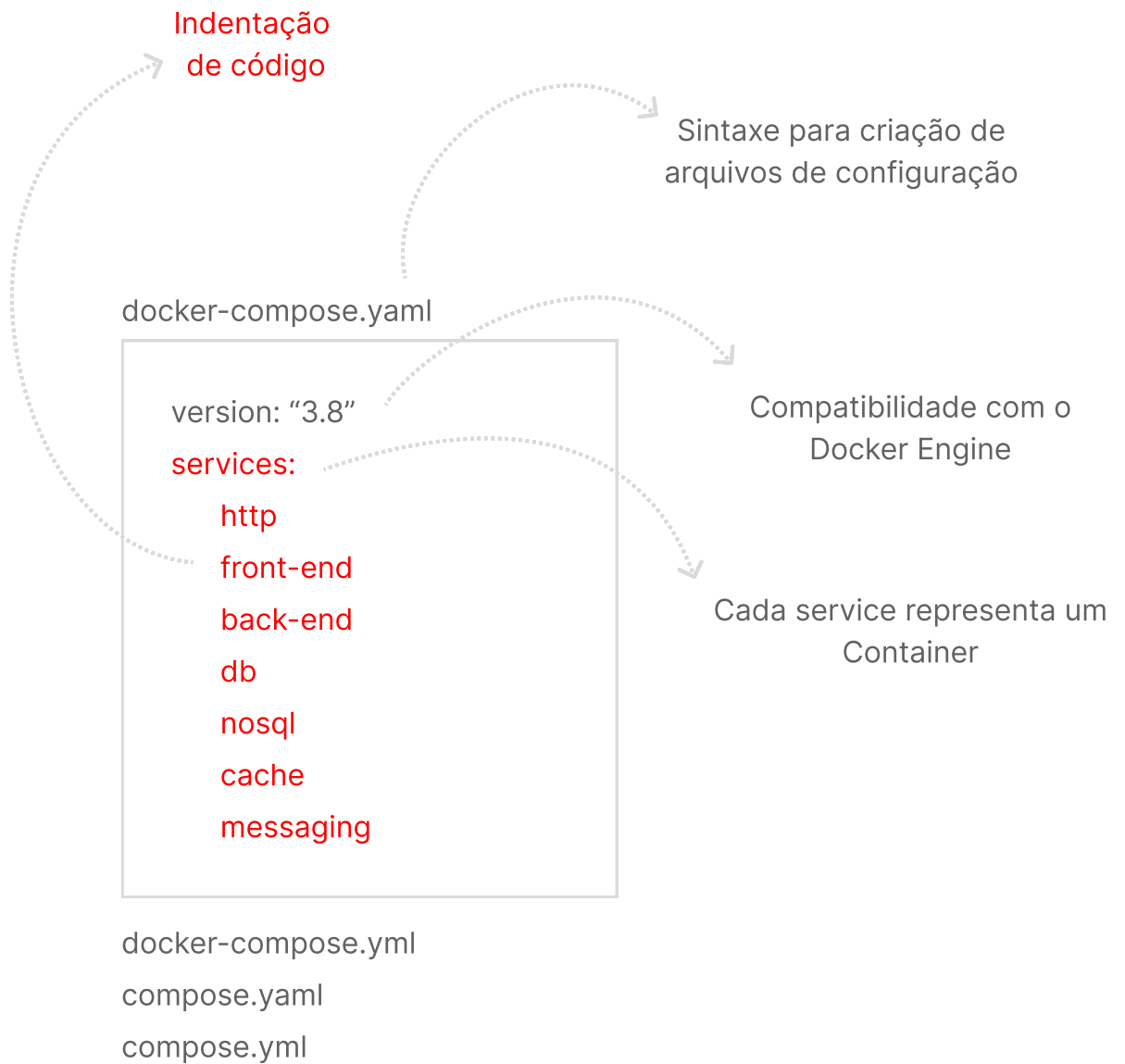


► Docker Compose

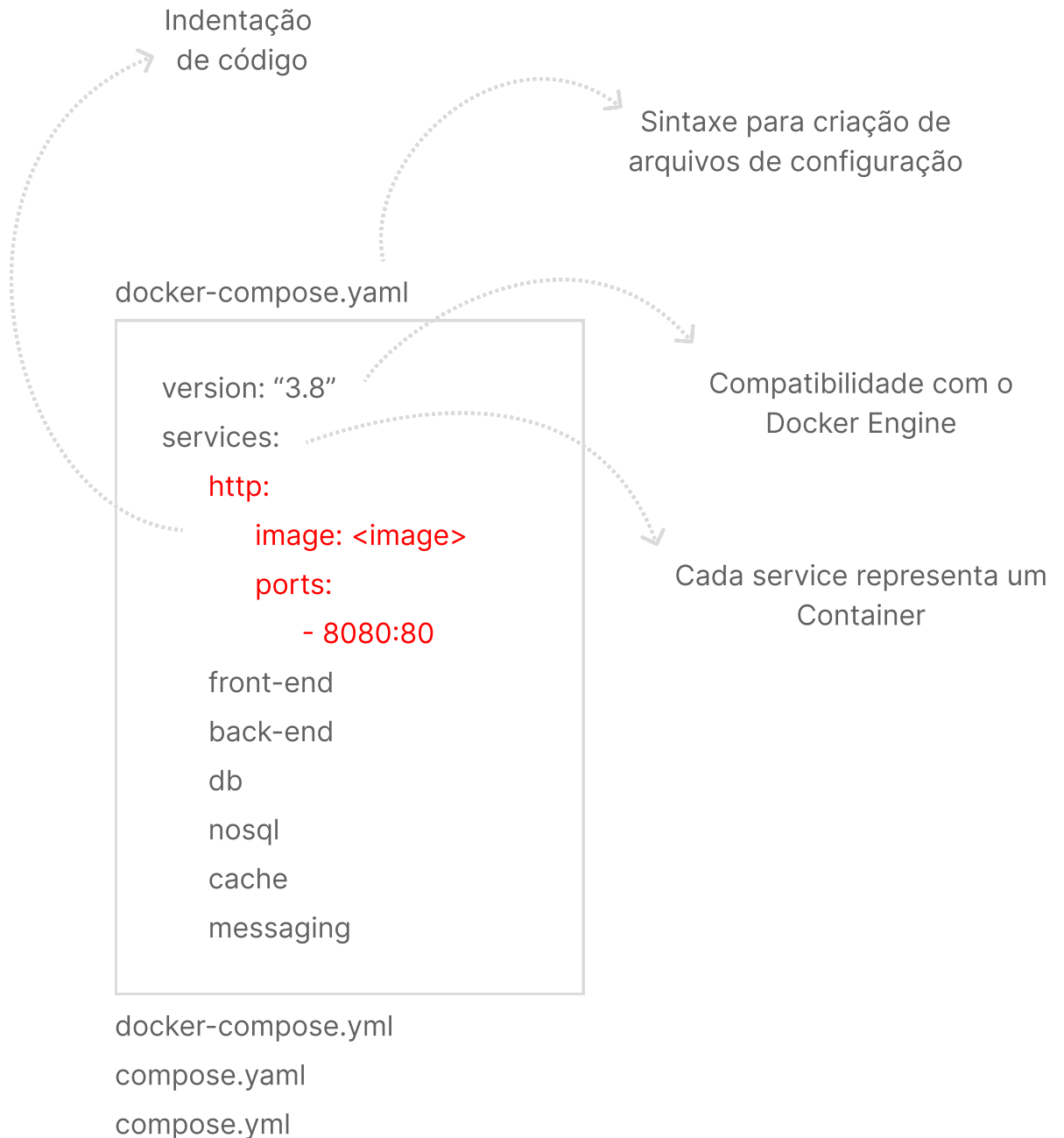
Definindo a versão da sintaxe utilizada no arquivo



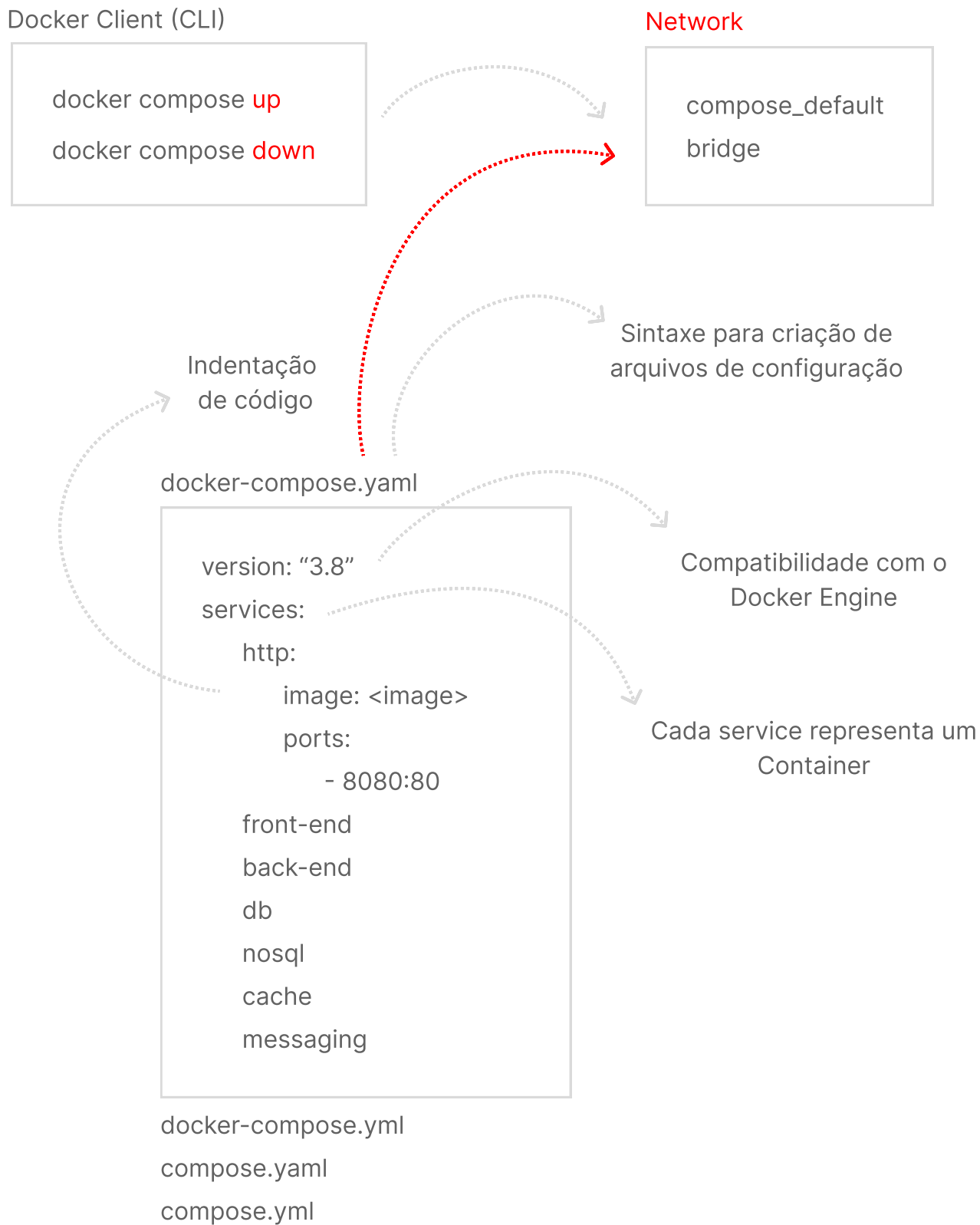
Definindo os serviços



Serviço HTTP (Nginx) - Definindo as instruções image e ports

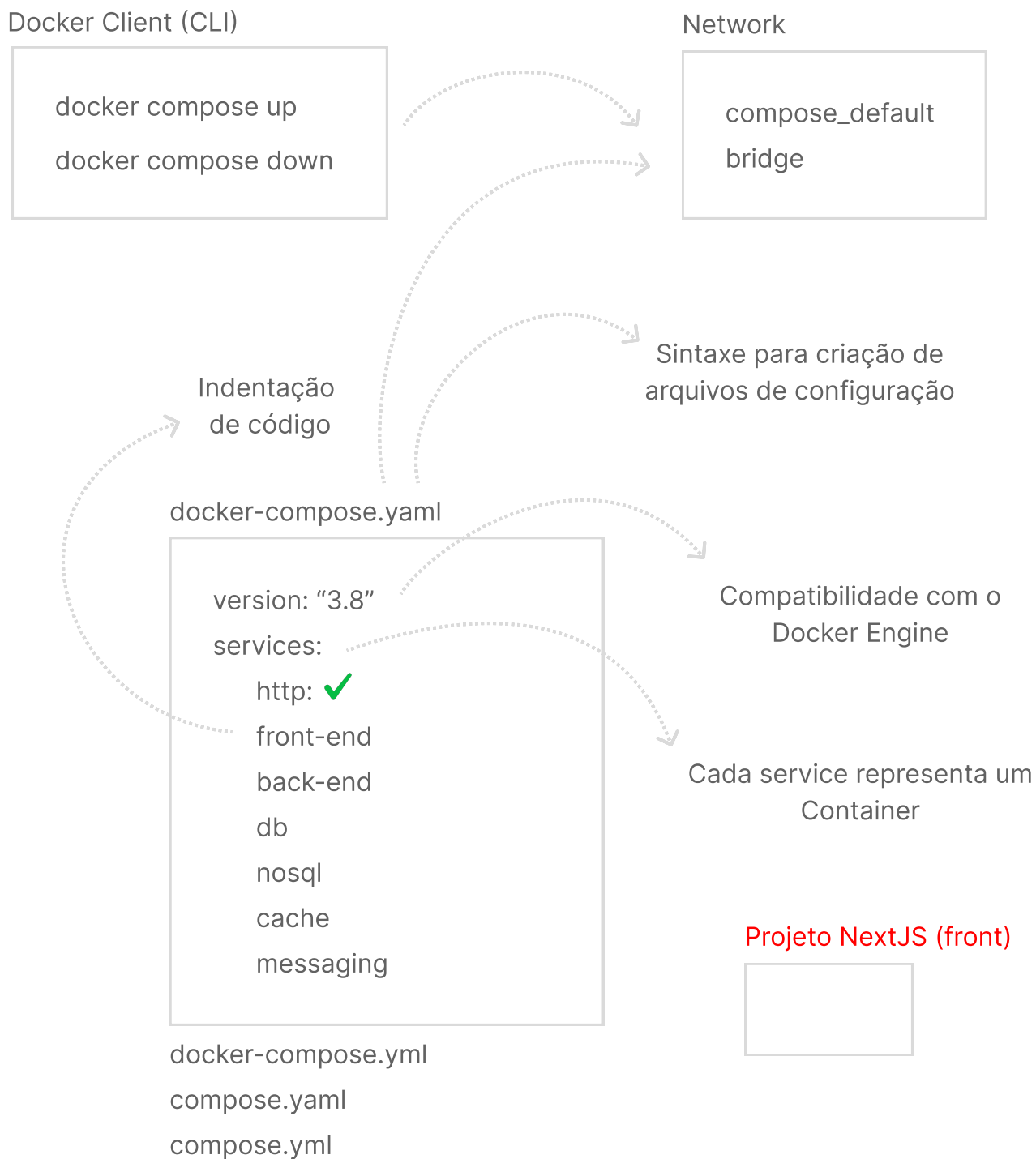


A rede padrão criada pelo Docker Compose



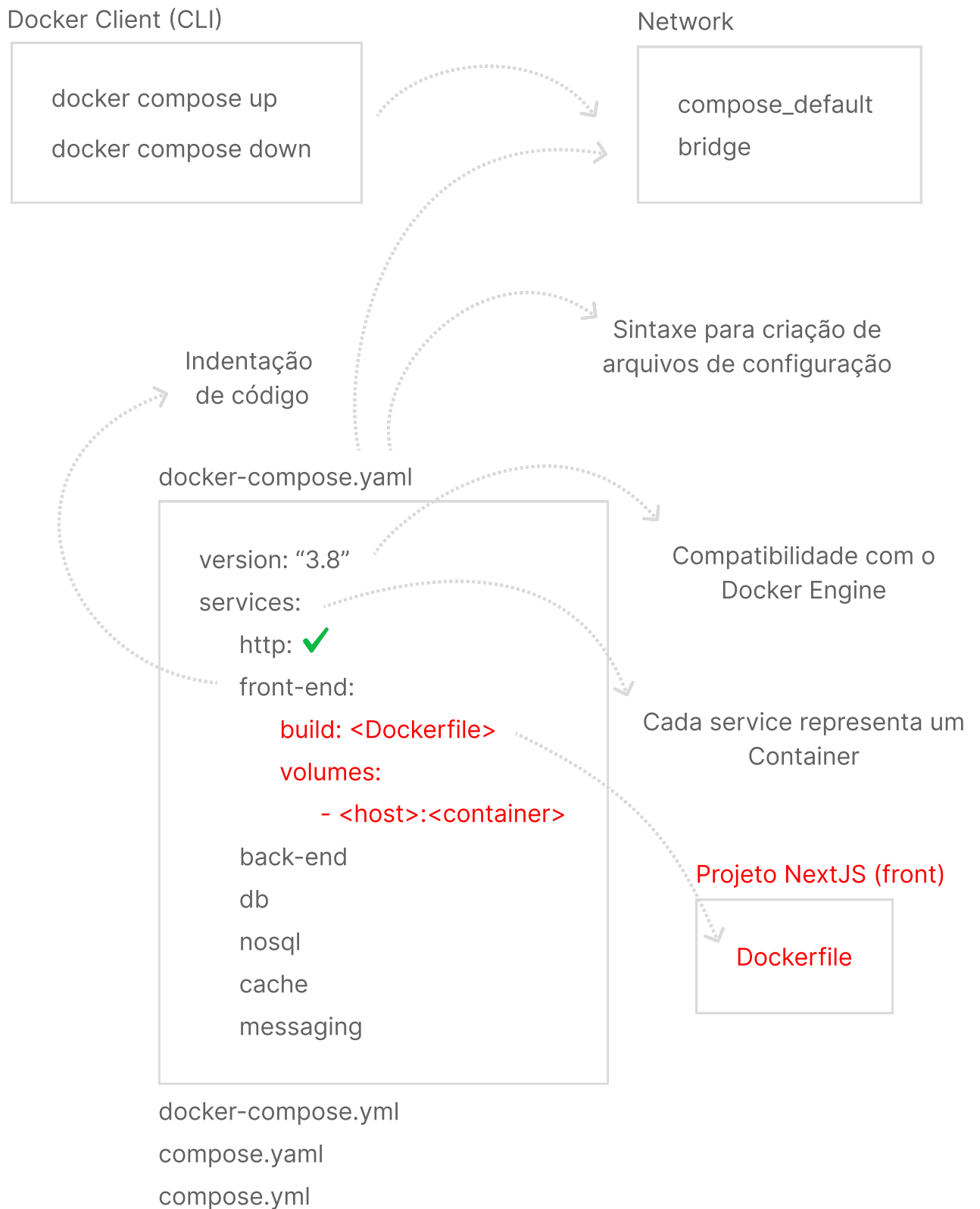
► Docker Compose

Serviço Front-end (NextJS) parte 1 - Organizando o projeto



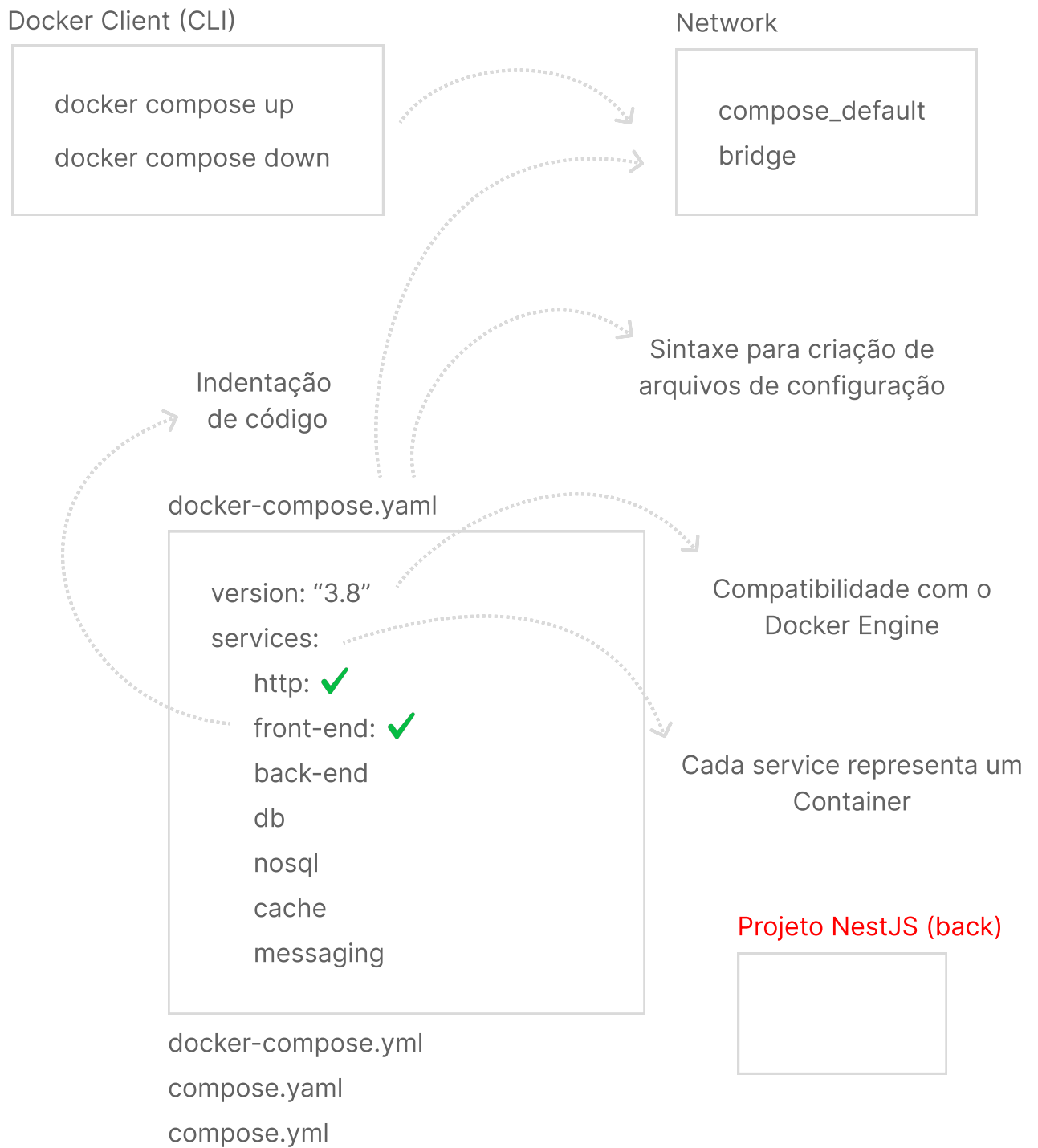
► Docker Compose

Serviço Front-end (NextJS) parte 2 - Definindo as instruções build e volumes



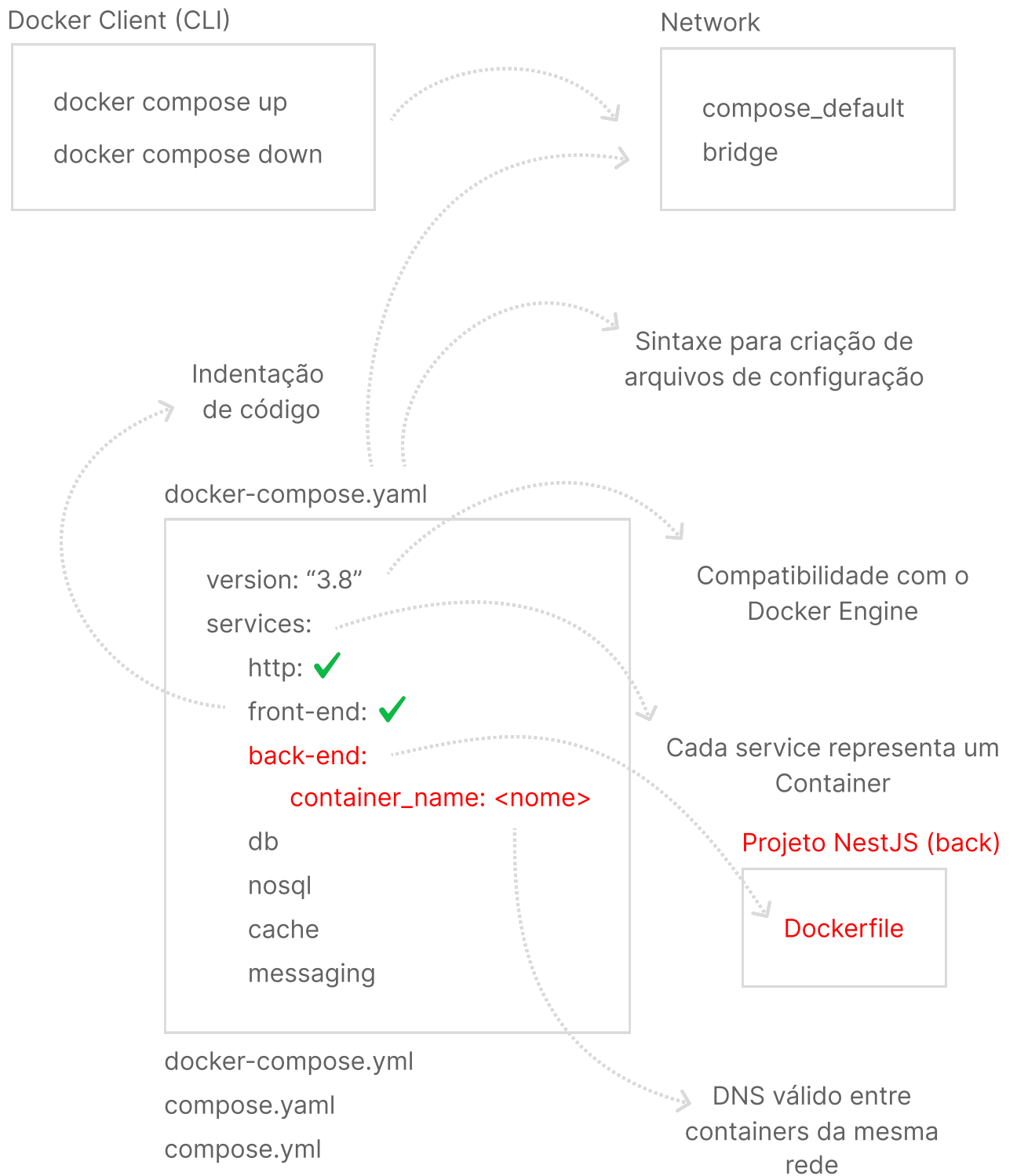
► Docker Compose

Serviço Back-end (NestJS) parte 1 - Organizando o projeto



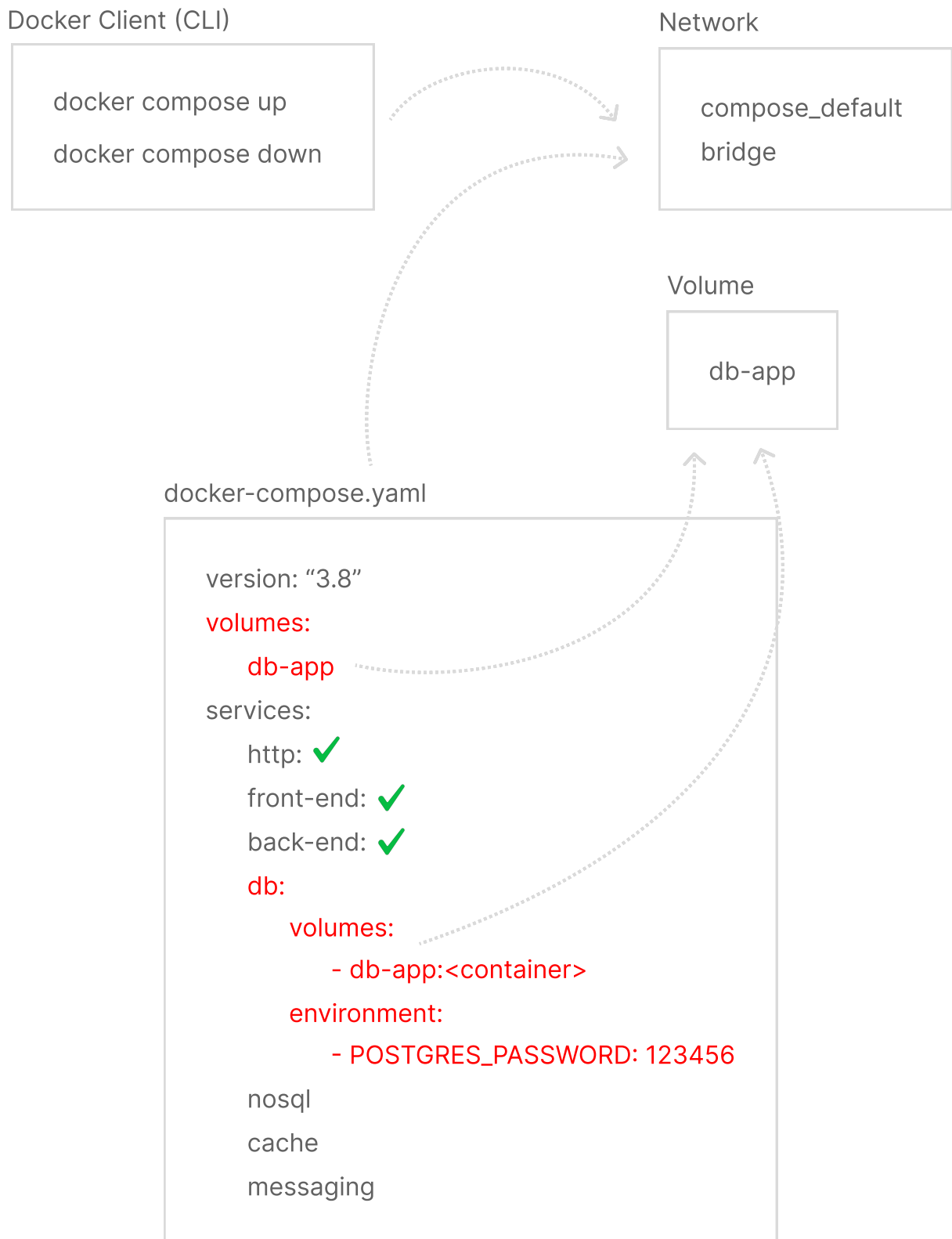
► Docker Compose

Serviço Back-end (NestJS) parte 2 - Definindo a instrução container_name



► Docker Compose

Serviço DB (PostgreSQL) - Definindo variáveis de ambiente e volumes



► Docker Compose

Reiniciando os serviços

Docker Client (CLI)

```
docker compose up
docker compose down
```

Network

```
compose_default
bridge
```

Volume

```
db-app
```

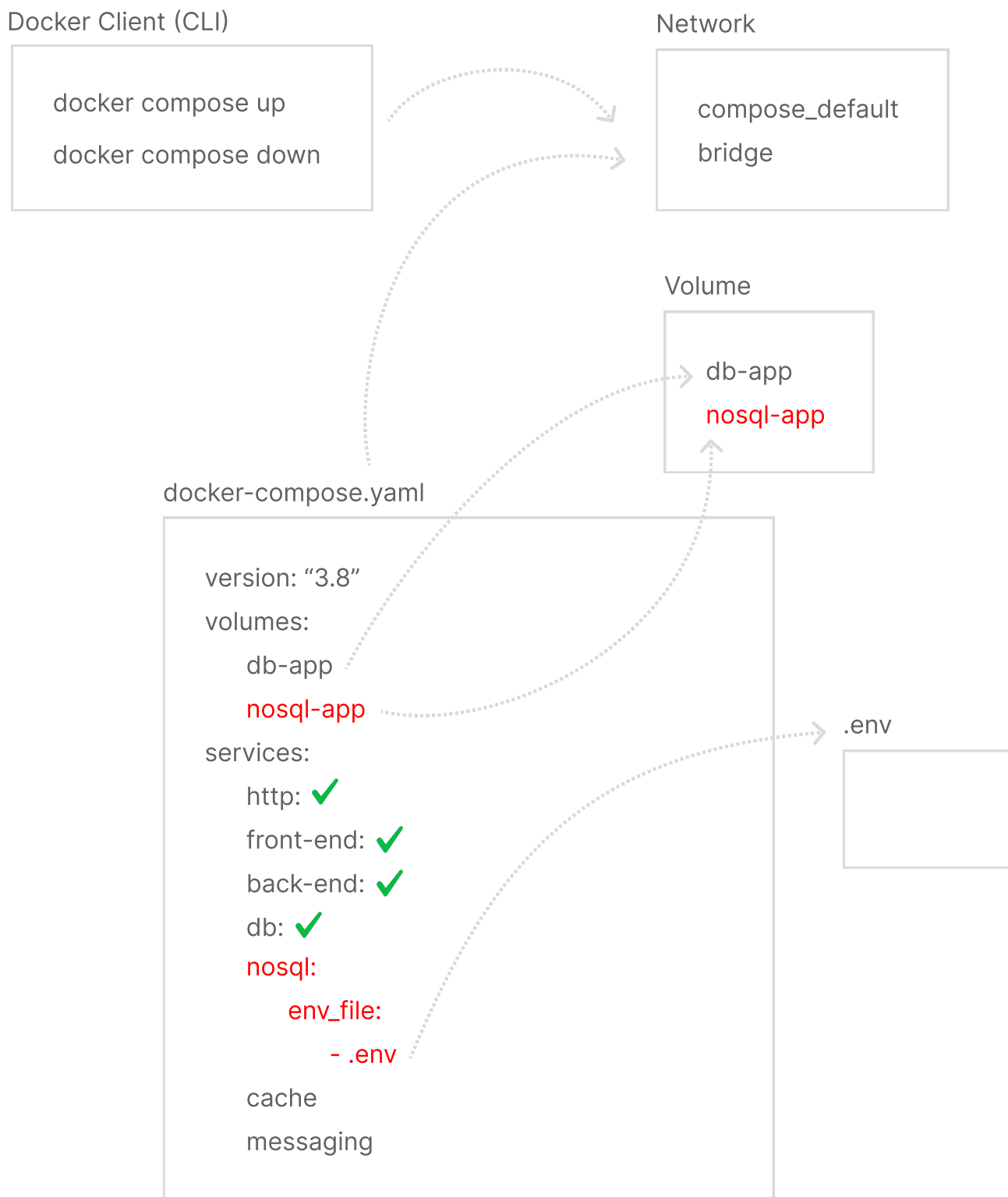
docker-compose.yml

```
version: "3.8"
volumes:
  db-app
services:
  http: ✓
  front-end: ✓
  back-end: ✓
  db: ✓
  restart: <policy>
  nosql
  cache
  messaging
```

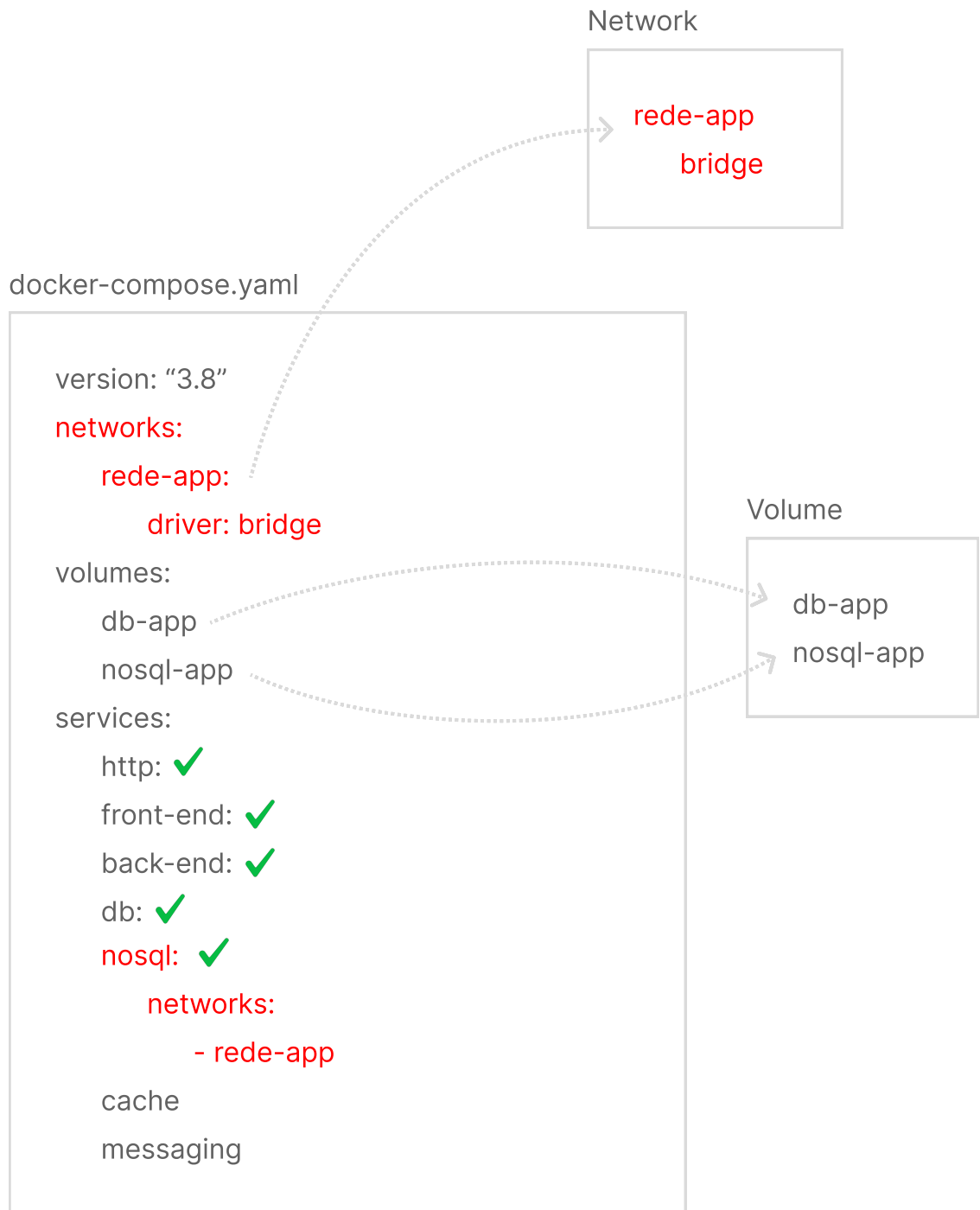
"no" (padrão)
always
on-failure
unless-stopped

► Docker Compose

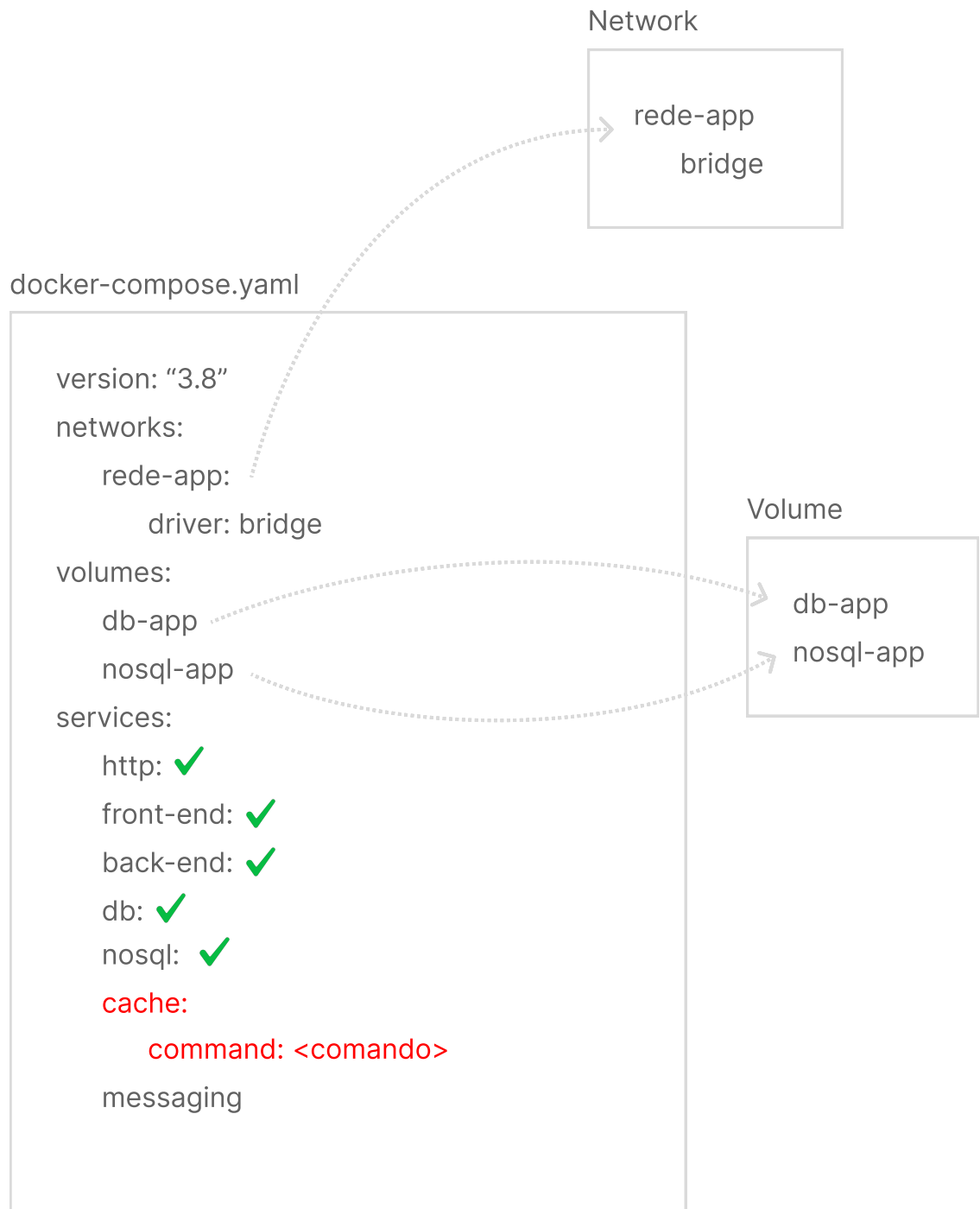
Serviço NoSQL (MongoDB) - Definindo variáveis de ambiente via arquivo externo



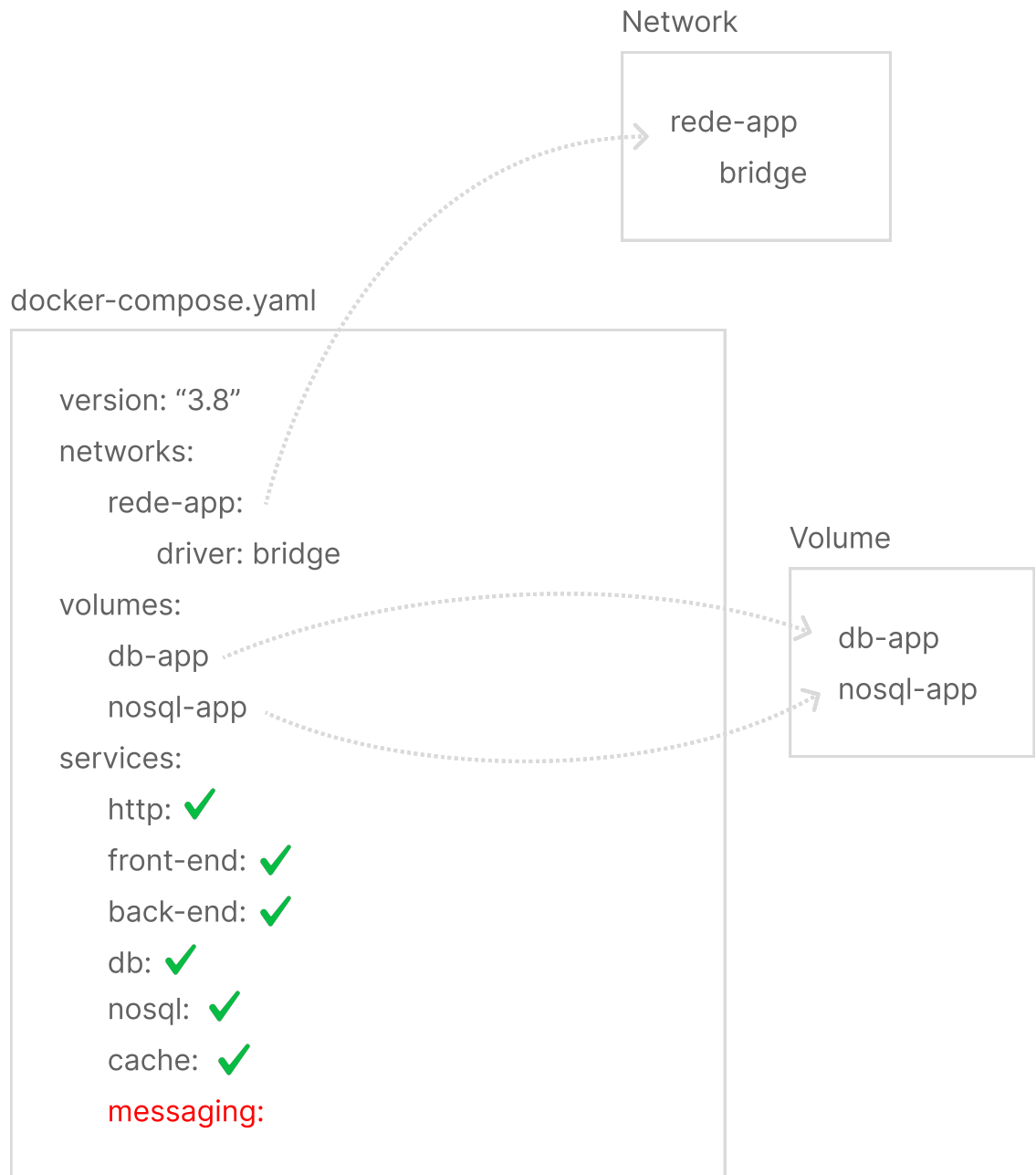
Criando uma nova rede



Serviço de Cache (Redis) - Definindo a instrução command

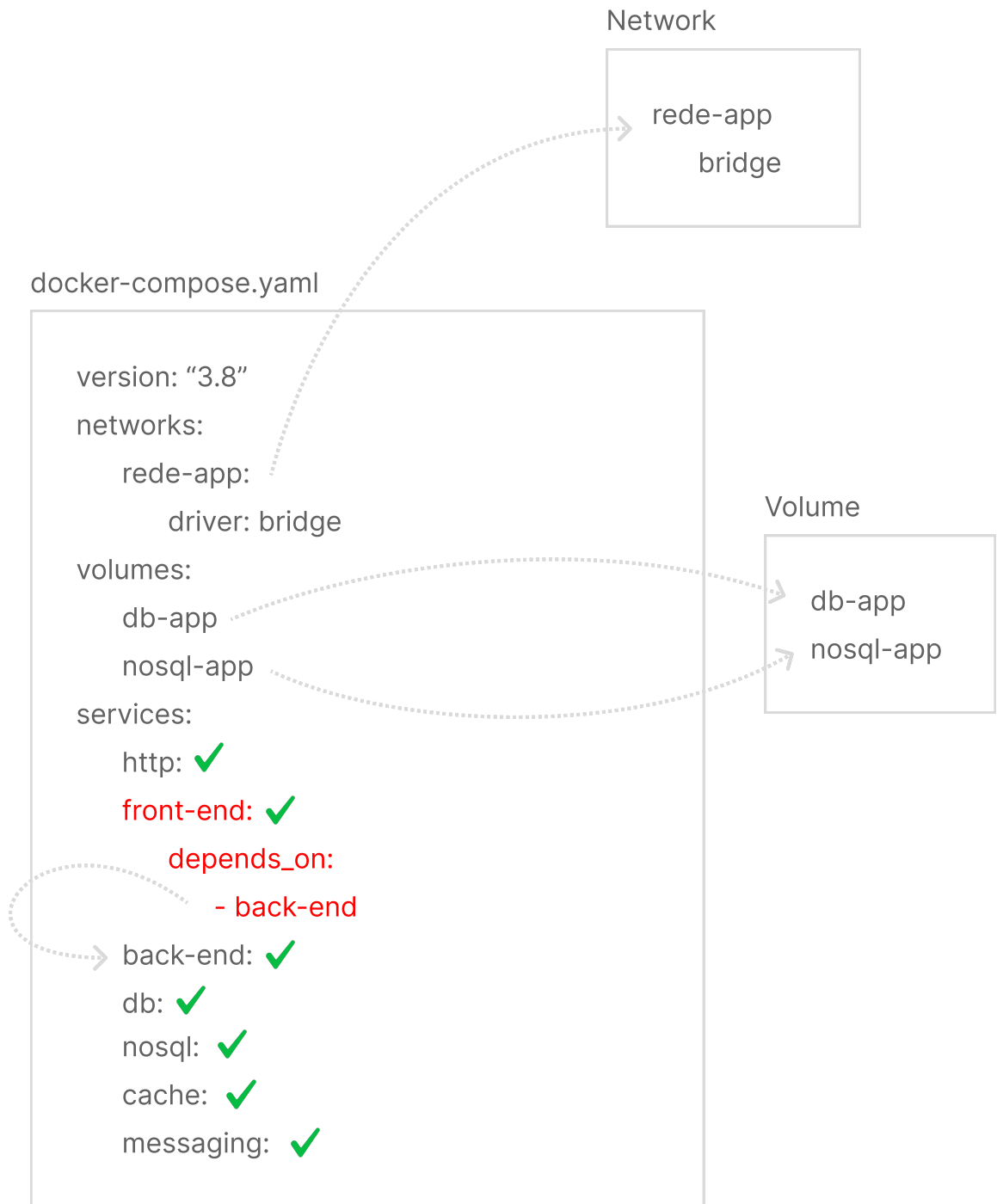


Serviço de Messaging (Kafka) – Consolidando os conhecimentos

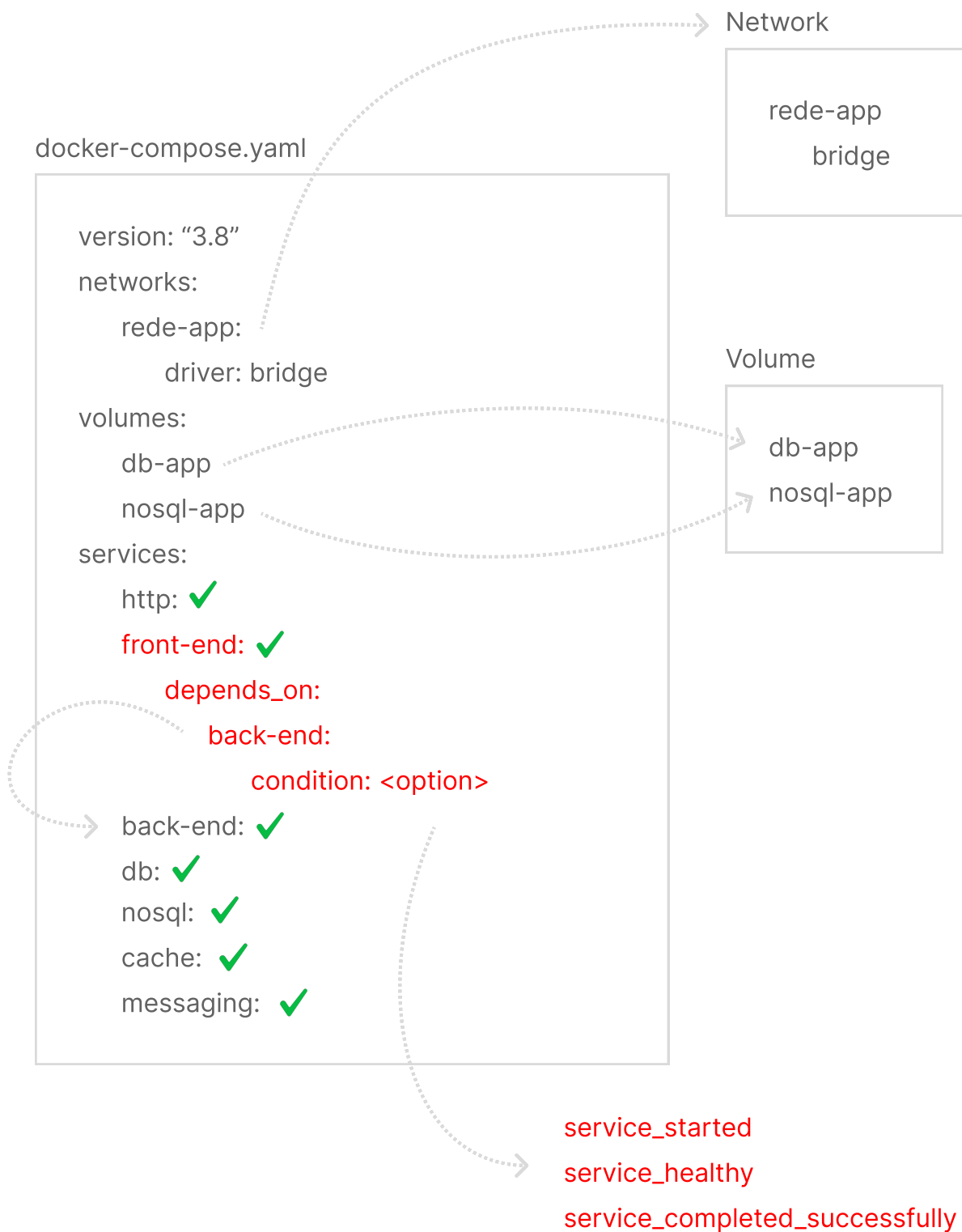


► Docker Compose

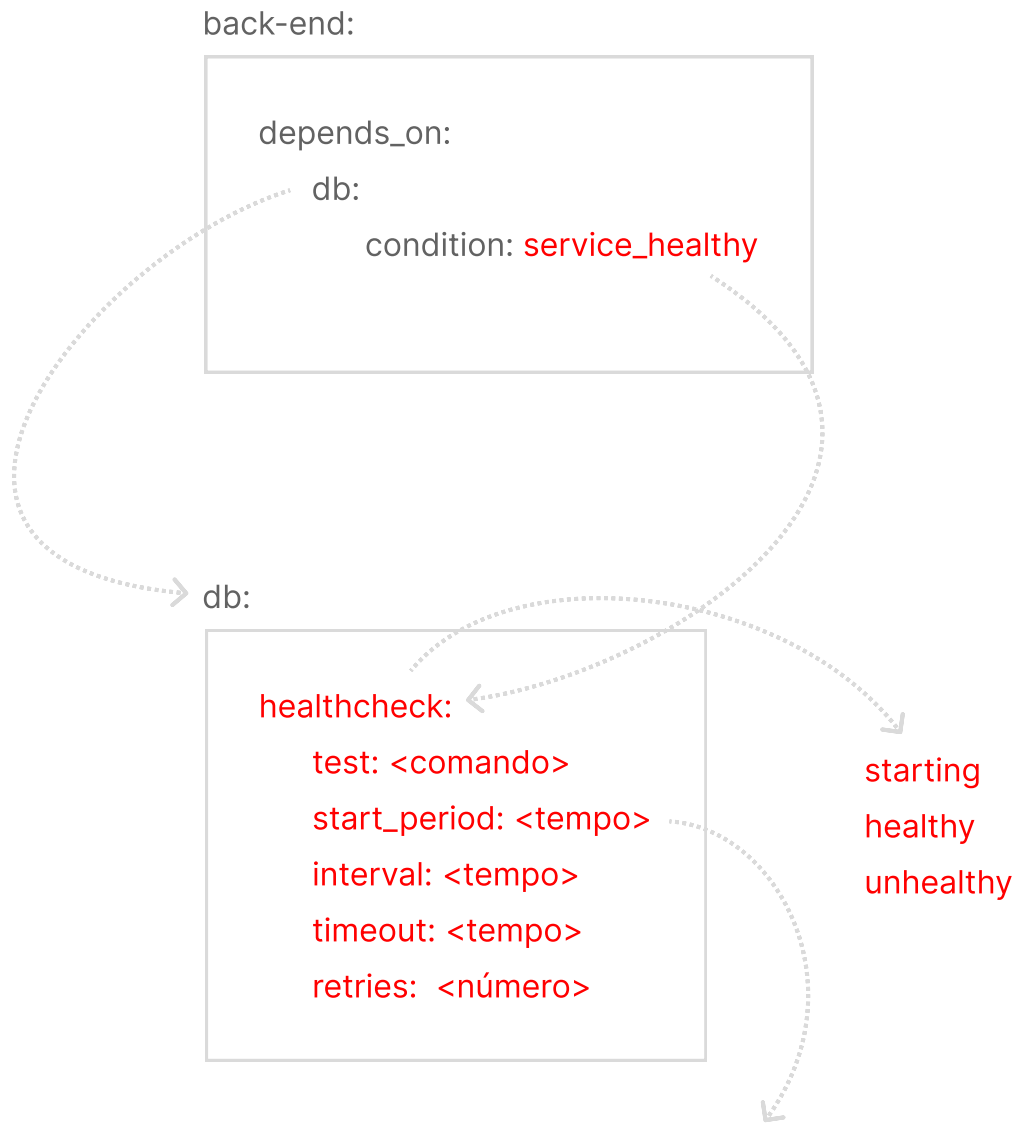
Depends On - Relação de dependências entre serviços



Depends On - Condições de dependência



Depends On - Avaliando a integridade do serviço



Sintaxe de definição dos tempos

2,5s

10s

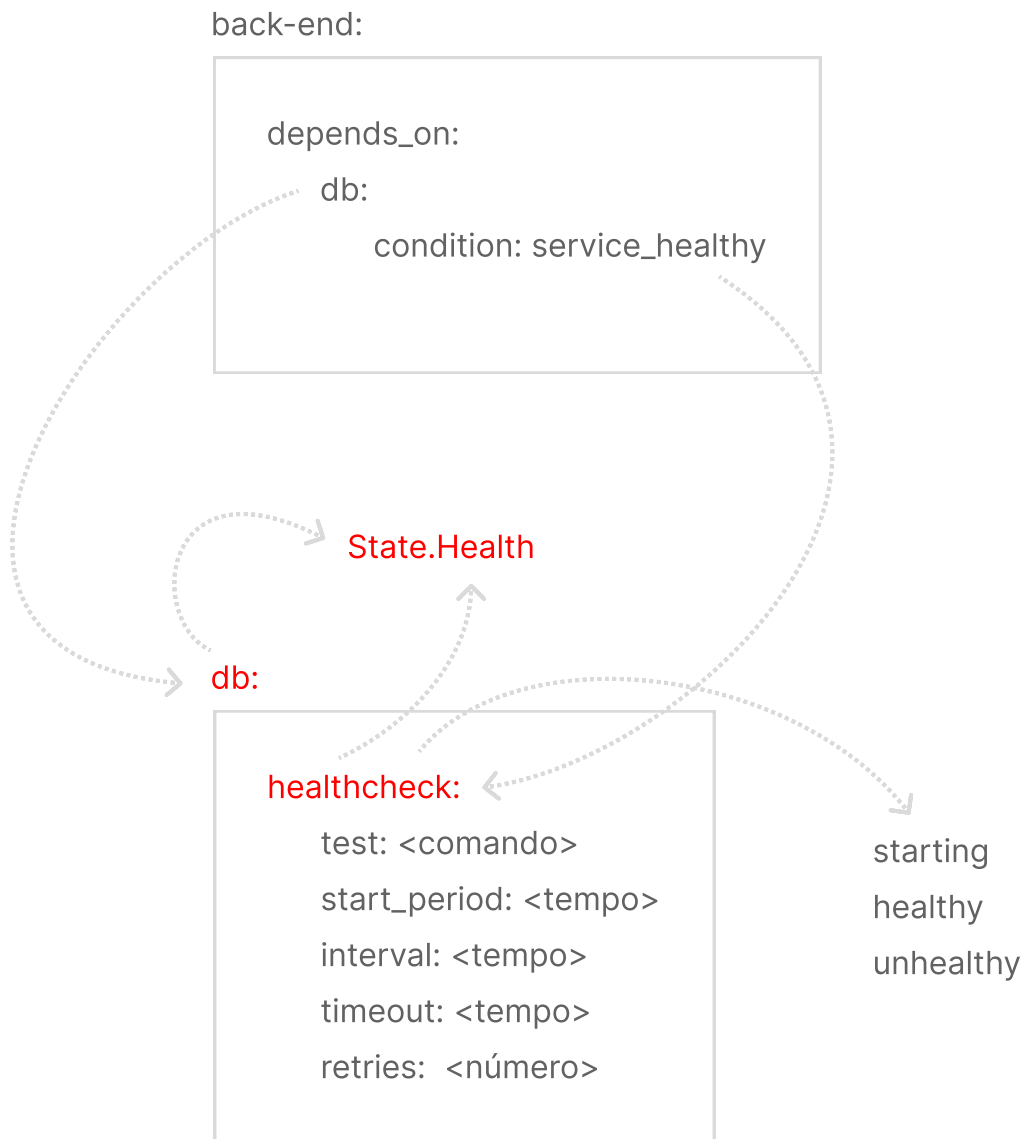
1m30s

1h15m

2h45m15s

► Docker Compose

Healthcheck - Logs de estado da integridade do serviço



► Docker Compose

Especificando o arquivo docker compose a ser interpretado

```
docker compose up
```



```
docker-compose.yaml  
docker-compose.yml  
compose.yaml  
compose.yml
```

```
docker compose -f <nome_arquivo> up
```



```
docker-compose.dev.yaml  
docker-compose.hml.yaml  
docker-compose.prn.yaml
```

ARGUS ACADEMY
www.argus-academy.com

