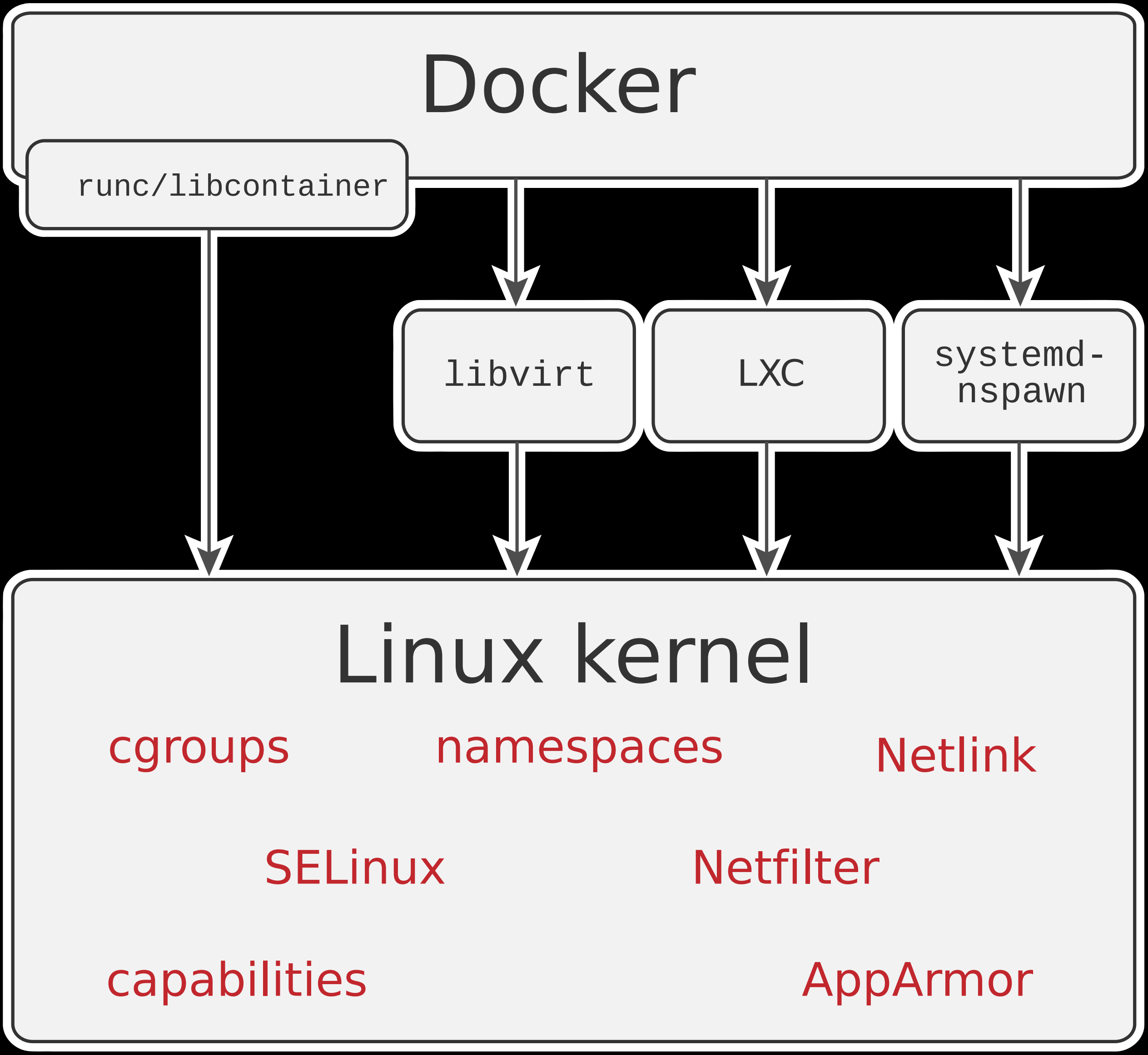
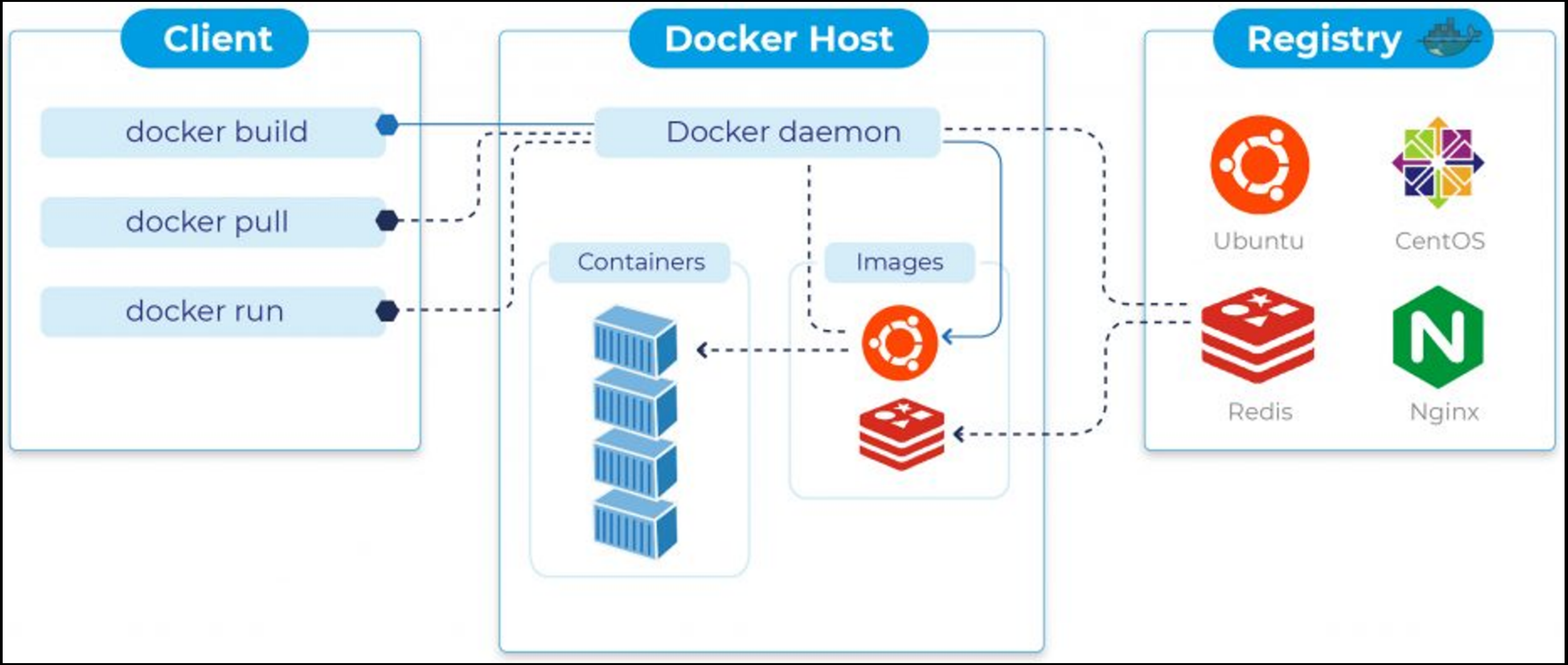


Puede usar diferentes interfaces para acceder a las características de virtualización del kernel.

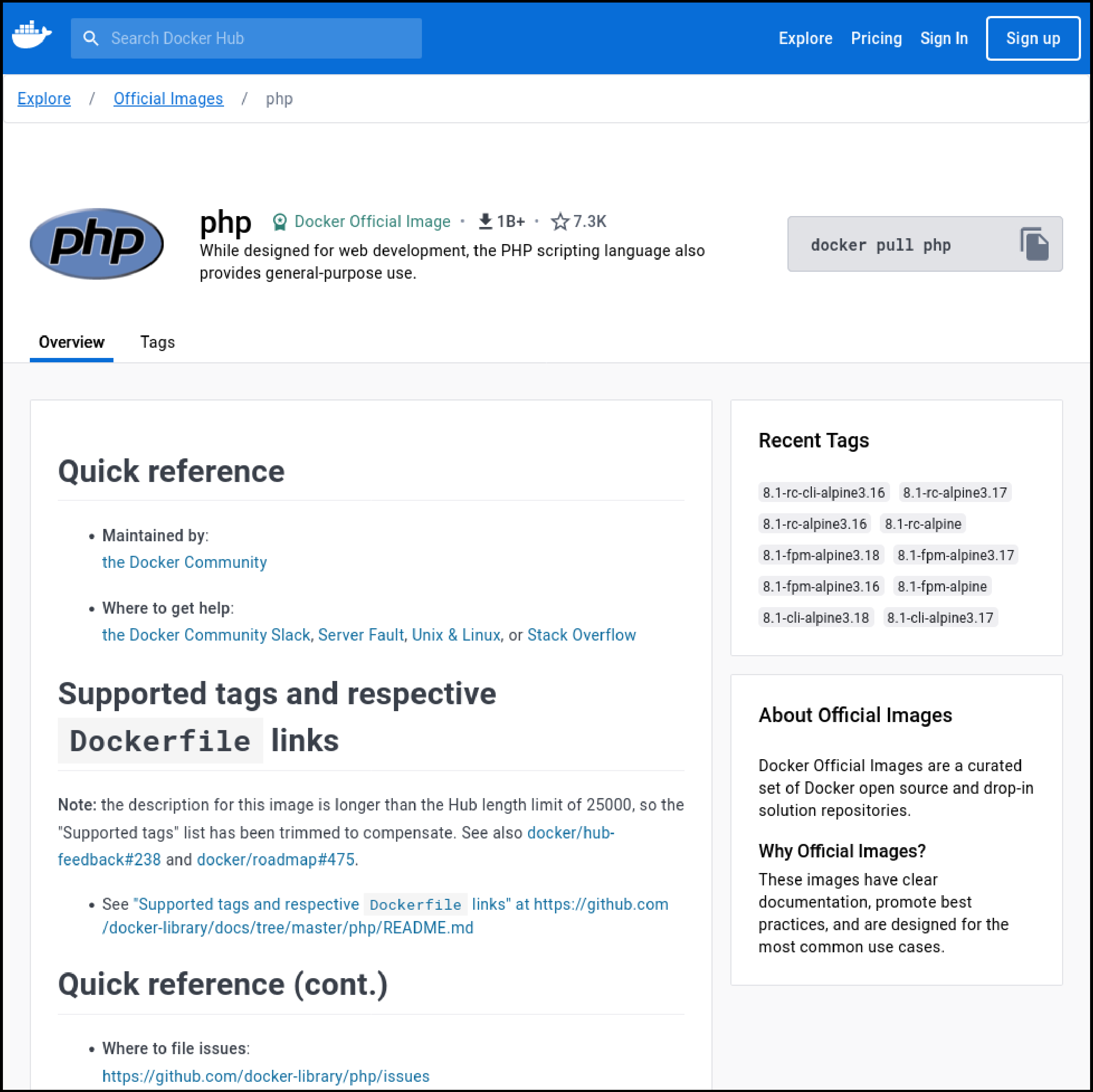
```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest:
sha256:88ec0acaa3ec199d3b7eaf73588f4518c25f9d34f58ce9a0df68429c5af48e8d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

The Docker Architecture



Docker Image Name
(Official Image)

$$\begin{tikzcd}
 & \texttt{hello-world} \arrow[d] & \\
 & \underline{\textbf{library/}}\texttt{hello-world}\underline{\textbf{:latest}} \arrow[d] & \\
 & \underline{\texttt{account/projectname}}\texttt{:}\underline{\texttt{version}} \arrow[dl] \arrow[dr] & \\
\texttt{repository} & & \texttt{tag}
\end{tikzcd}$$


Search Docker Hub

Explore Pricing Sign In Sign up

Explore / Official Images / php

php Docker Official Image • 1B+ • 7.3K

While designed for web development, the PHP scripting language also provides general-purpose use.

```
docker pull php
```

Overview Tags

Quick reference

- **Maintained by**:
  the Docker Community
- **Where to get help**:
  the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow

Supported tags and respective `Dockerfile` links

Note: the description for this image is longer than the Hub length limit of 25000, so the "Supported tags" list has been trimmed to compensate. See also docker/hub-feedback#238 and docker/roadmap#475.

- See "Supported tags and respective `Dockerfile` links" at https://github.com/docker-library/docs/tree/master/php/README.md

Quick reference (cont.)

- **Where to file issues**:
  https://github.com/docker-library/php/issues

Recent Tags

8.1-rc-cli-alpine3.16 8.1-rc-alpine3.17
8.1-rc-alpine3.16 8.1-rc-alpine
8.1-fpm-alpine3.18 8.1-fpm-alpine3.17
8.1-fpm-alpine3.16 8.1-fpm-alpine
8.1-cli-alpine3.18 8.1-cli-alpine3.17

About Official Images

Docker Official Images are a curated set of Docker open source and drop-in solution repositories.

Why Official Images?

These images have clear documentation, promote best practices, and are designed for the most common use cases.

Global and Container Commands

```
info               # Display system-wide information
search             # Search the Docker Hub for images

container

  ls (*ps)         # List containers
  *run             # Run a command in a new container
  *exec            # Run a command in a running container
  *logs            # Fetch the logs of a container
  *top             # Display the running processes of a container
  *port            # List port mappings or a specific mapping for the container
  *stop            # Stop one or more running containers
  *start           # Start one or more stopped containers
  *kill            # Kill one or more running containers
  *rm              # Remove one or more containers
  prune            # Remove all stopped containers
  *cp              # Copy files/folders between a container and local filesystem
```

Run a Container

```
# Run an 'alpine' container
docker run alpine
# List running containers
docker ps [-a]
## Options:
# * [-i|--interactive] -> Keep STDIN open even if not attached
# * [-t|--tty]         -> Allocate a pseudo-TTY
# * [-e|--env]         -> Set environment variables
# * [--name]           -> Assign a name to the container
# * [--rm]             -> Automatically remove container when exits
docker run --rm --interactive --tty --name pepe alpine
```

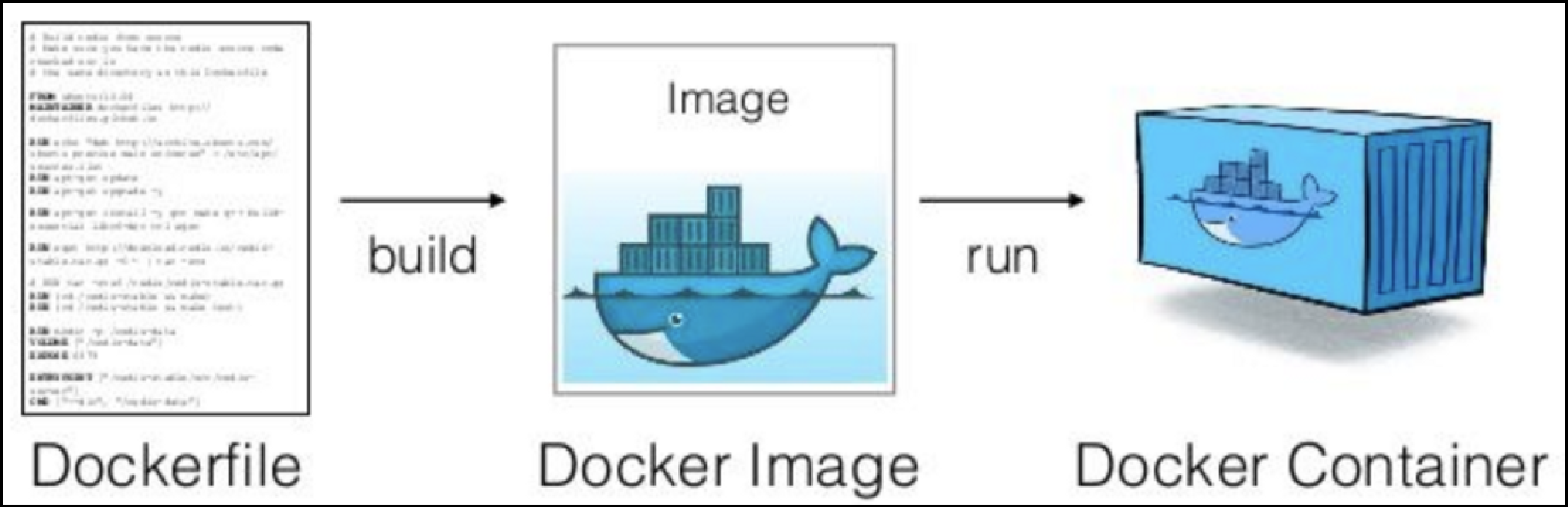
Run Two Instances

```
## Options:
# * [-d|--detach]  -> Run container in the background
docker run --rm --detach --name="alfa" nginx:alpine
# Enter running instance
# * create files
docker exec --interactive --tty alfa /bin/sh
# Consume from another container
# * ping instances
docker run --rm -it alpine
```

Publish Ports

```
## Options:
# * [-p|--publish] -> Publish a container's port
# * [-v|--volume]  -> Create a bind mount. A file or directory on
#                     the host machine is mounted into a container
docker run --rm --detach --name alfa --publish 8090:80 nginx:alpine
# Create basic/tiny HTML
echo '<h1>Hola</h1><p>Mundo</p>' > index.html
# Run web server and mount volume
docker run --rm --detach --name beta --publish 8091:80 \
    --volume $(pwd):/usr/share/nginx/html:ro nginx:alpine
# Edit HTML file and refresh browser
```

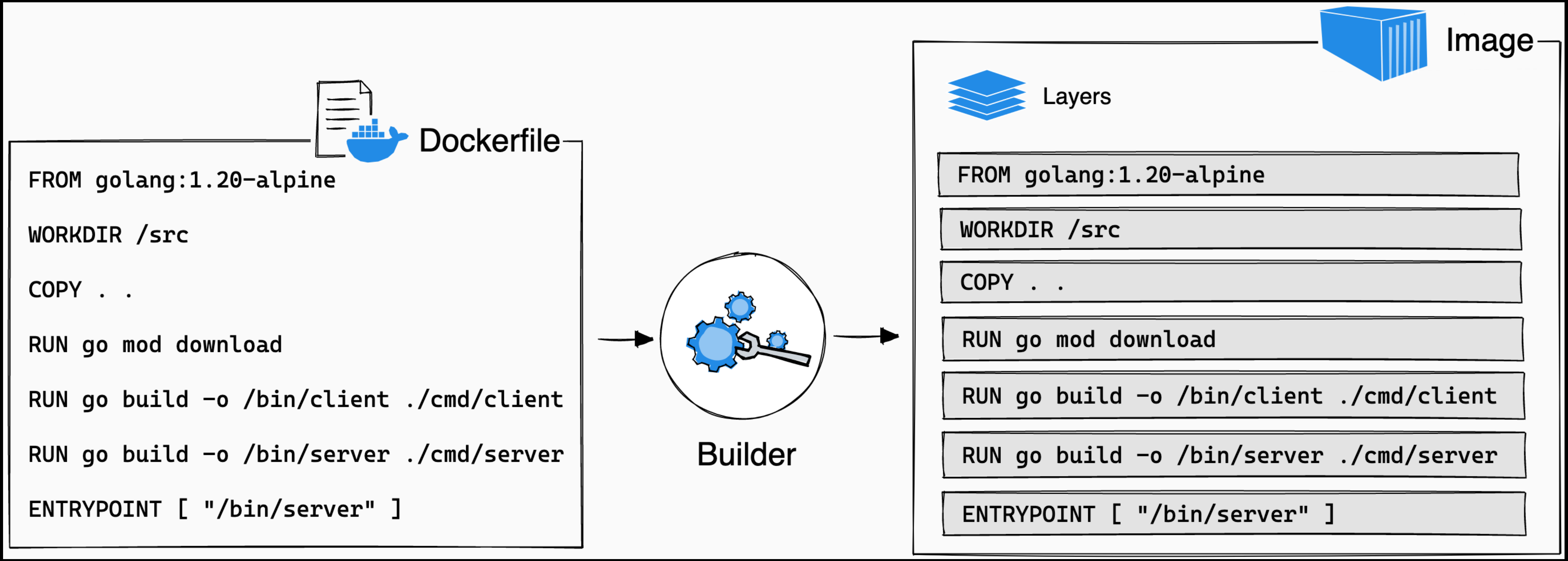
Build Docker Image



Images Commands

```
image

  ls (*images) # List images
  *build       # Build an image from a Dockerfile
  *tag         # Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  rm (*rmi)    # Remove one or more images
  prune        # Remove unused images
  *load        # Load an image from a tar archive or STDIN
  *save        # Save one or more images to tar archive (STDOUT by default)
```

Docker Image Layers

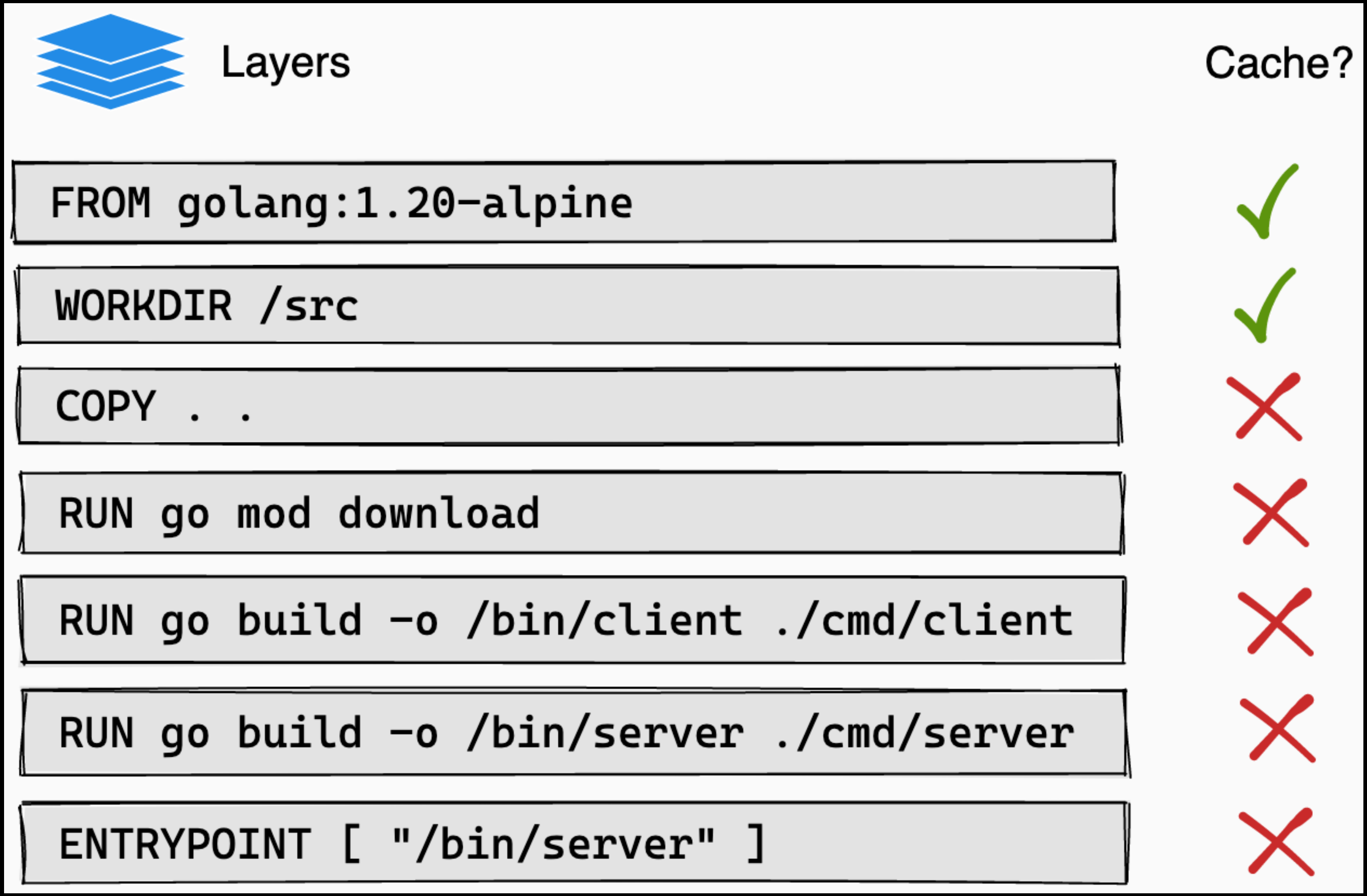


```
FROM golang:1.20-alpine

WORKDIR /src

COPY . .

RUN go mod download

RUN go build -o /bin/client ./cmd/client

RUN go build -o /bin/server ./cmd/server

ENTRYPOINT [ "/bin/server" ]
```

```
FROM golang:1.20-alpine
WORKDIR /src
COPY . .
RUN go mod download
RUN go build -o /bin/client ./cmd/client
RUN go build -o /bin/server ./cmd/server
ENTRYPOINT [ "/bin/server" ]
```

- **Documentación:** https://docs.docker.com/build/guide/layers/
- **Viejo Sistema de Archivos:** aufs
- **Nuevo Sistema de Archivos:** OverlayFS

Docker Image Cache



```
FROM golang:1.20-alpine
WORKDIR /src
COPY . .
RUN go mod download
RUN go build -o /bin/client ./cmd/client
RUN go build -o /bin/server ./cmd/server
ENTRYPOINT [ "/bin/server" ]
```

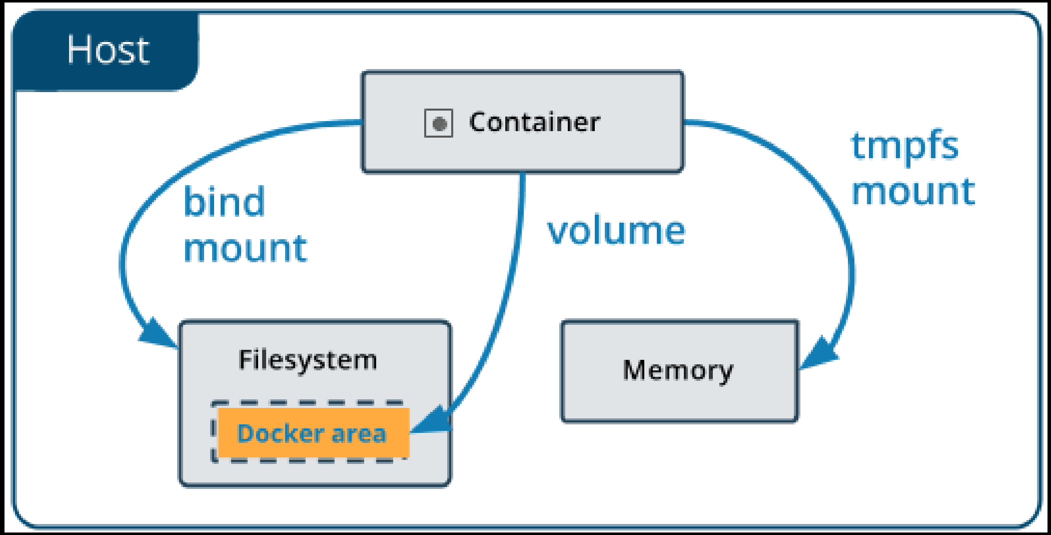
Layers	Cache?
FROM golang:1.20-alpine	✓
WORKDIR /src	✓
COPY . .	✗
RUN go mod download	✗
RUN go build -o /bin/client ./cmd/client	✗
RUN go build -o /bin/server ./cmd/server	✗
ENTRYPOINT ["/bin/server"]	✗

✔ **Primero:** Cambios infrecuentes
✔ **Ultimo:** Cambios frecuentes

Build Image

```
cat Dockerfile
  FROM nginx:alpine
  WORKDIR /usr/share/nginx/html
  COPY . .
## Options:
# * [--rm] Remove intermediate containers after a successful build
# * [--force-rm] Always remove intermediate containers
# * [-t|--tag] Repository name (and optionally tags) for resulting image
docker build --force-rm --tag "edi2/web:demo" .
# Run new image
docker run --rm --name="edi2" --publish 8080:80 edi2/web:demo
# Update Dockerfile
tail -n1 Dockerfile
  RUN rm Dockerfile
docker build --force-rm --tag "edi2/web:demo" .
```

Docker Volumes



<u>Bind Mount:</u>

- Existen desde los inicios de Docker.
- Tienen menos funcionalidades que los volúmenes.
- Se monta un archivo o directorio del host en el contenedor usando su ruta absoluta.

<u>Volume:</u>

- Crea un nuevo directorio en el almacenamiento de Docker y gestionado por Docker

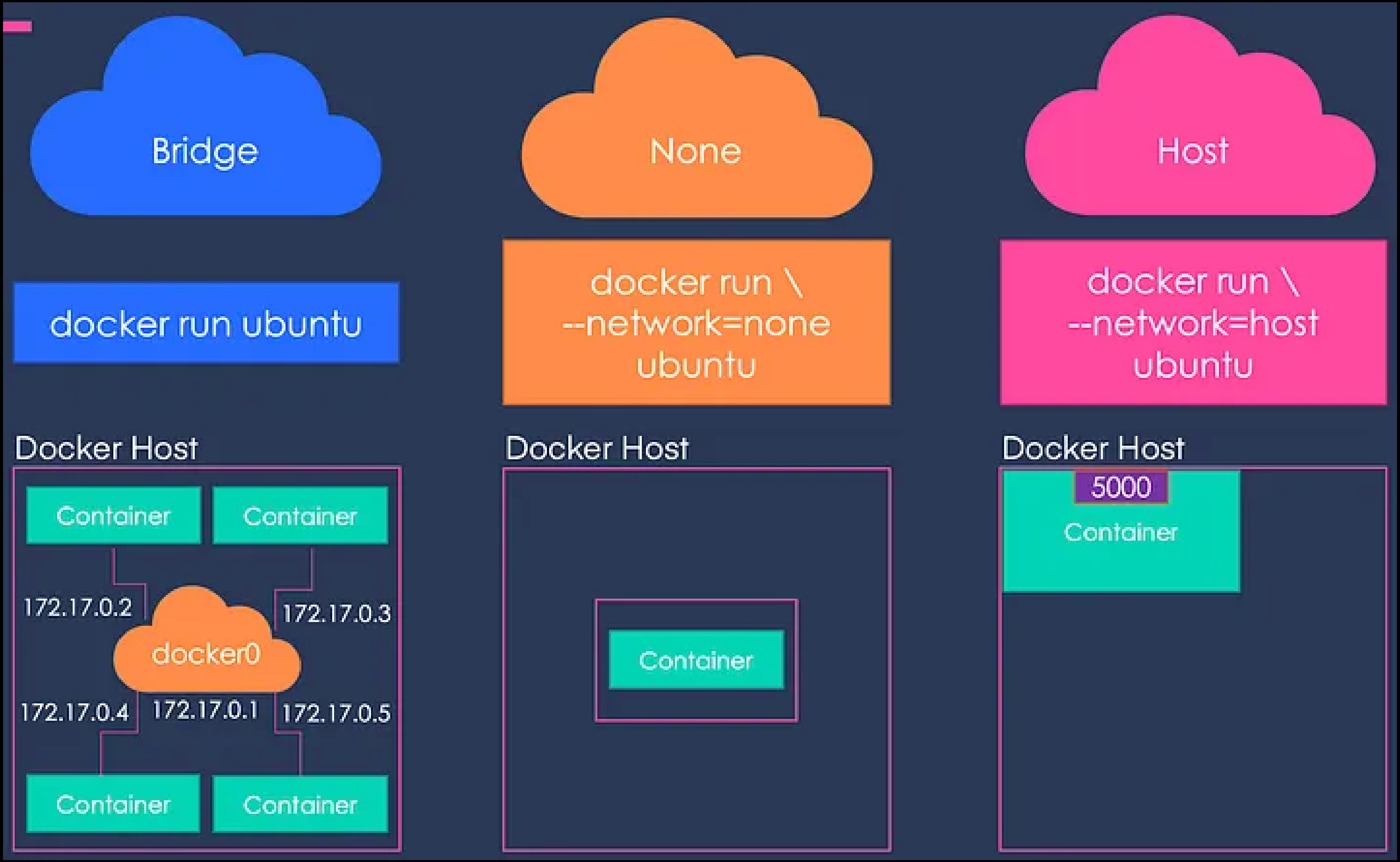
Docker Volumes

```
volume

  ls            # List volumes
  create        # Create a volume
  rm            # Remove one or more volumes
  prune         # Remove all unused local volumes
```

Using Volumes

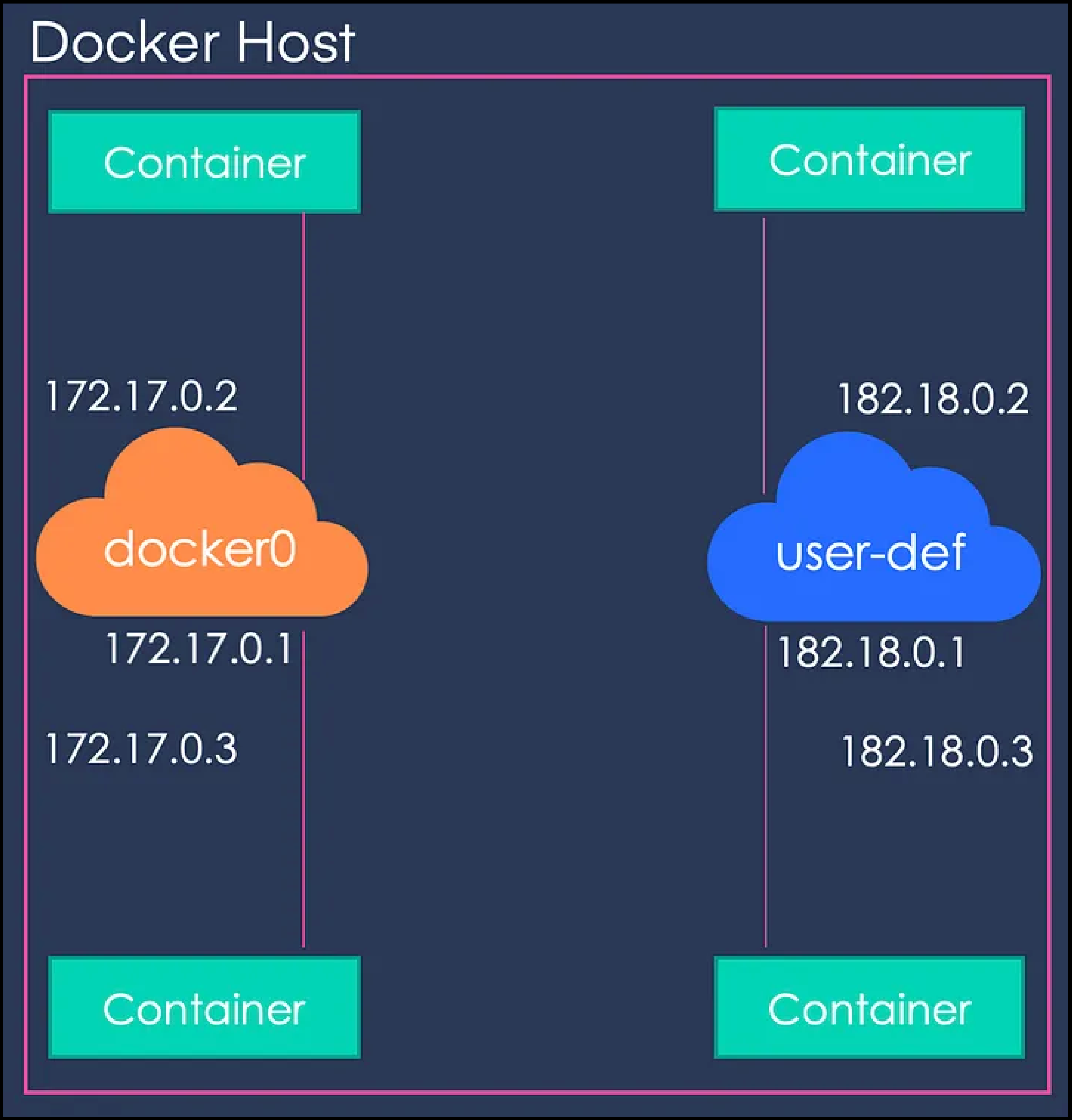
```
# List volumes
docker volume ls
# Create volume
docker volume create pepe-volume
# Mount (use) a volume
docker run --rm -it \
  --mount source=pepe-volume,target=/opt \
  alpine
```

Docker Network



Docker Network

```
network

  ls            # List networks
  create        # Create a network
  connect       # Connect a container to a network
  disconnect    # Disconnect a container from a network
  rm            # Remove one or more networks
  prune         # Remove all unused networks
```


Docker Host
Container
Container
172.17.0.2
182.18.0.2
docker0
user-def
172.17.0.1
182.18.0.1
172.17.0.3
182.18.0.3
Container
Container

```
# Create custom network
$ docker network create --driver bridge \
    --subnet 182.18.0.0/16 edi2-net-test
$ docker network ls
# Run container in custom network
$ docker run --rm --interactive --tty --network edi2-net-test \
    --name alfa alpine /bin/sh
/ # ifconfig
# Detach container in custom network
$ docker run --rm --detach --network edi2-net-test --name alfa alpine \
    /bin/sh -c 'trap exit INT TERM; while true; do sleep 1 & wait; done;'
# Run second container in default network
$ docker run --rm --interactive --tty --name bravo \
    alpine /bin/sh
/ # ifconfig
/ # ping alfa
/ # exit
# Run third container in custom network
$ docker run --rm --interactive --tty --network edi2-net-test \
    --name charlie alpine /bin/sh
/ # ifconfig
/ # ping alfa
/ # exit
$ docker stop alfa
$ docker network rm edi2-net-test
```

Docker Compose

Compose es una herramienta para definir y ejecutar una aplicación compuesta por más de un contenedor. Con compose, se usa un archivo YAML para configurar los servicios de la aplicación. Luego, con un solo comando, se crean e inician todos los servicios definido en el archivo de configuración.

Documentación: https://docs.docker.com/compose/

```
docker-compose up [--detach]
```

Crear las imágenes del proyecto **"votar"** (`'docker build'`) y ejecutar todos los servicios (`'docker-compose'`).

```
# Clone VOTE server (REST API)
git clone https://repo.or.cz/asis23-votoe-server.git
# Enter repository directory
cd asis23-votoe-server
# Build server image
docker build --rm --tag 'edi2/server' .
# Move to parent directory
cd ..
# Clone VOTE client
git clone https://repo.or.cz/asis23-votoe-client.git
# Enter repository directory
cd asis23-votoe-client
# Build client image
docker build --rm --tag 'edi2/client' .
# Update `hosts` file
# Using the tool: https://github.com/guumaster/hostctl
# More info: https://en.wikipedia.org/wiki/Hosts_(file)
sudo hostctl add domains edi2 api.vot.ar --ip "127.0.0.1"
sudo hostctl add domains edi2 www.vot.ar --ip "127.0.0.1"
# Run application with docker compose
docker-compose up
# Visti http://www.vot.ar
# Control-C to stop docker compose
```