

Docker - Basic

管培勛 (phkoan)

Outline

- What's Container
- Docker Installation
- Common Commands

What's Container

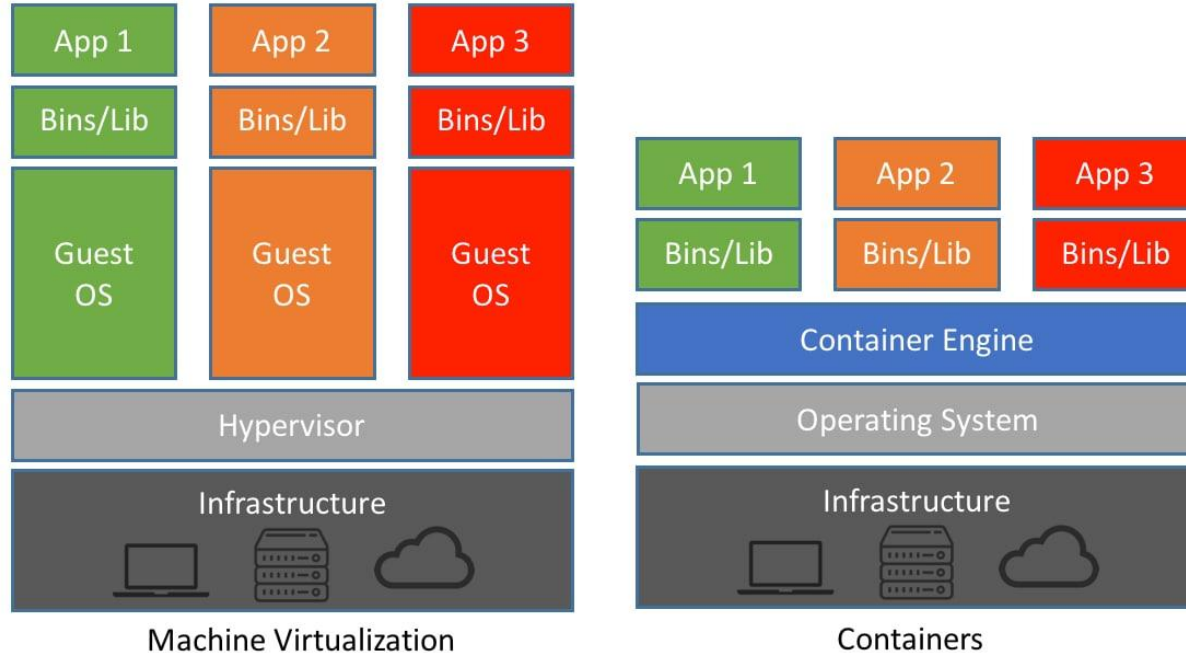
What's Container

- **Container** is a subsystem that has isolated resources.
- **Container image** is a lightweight and standalone package that includes everything needed to run software.
- On Linux, this technology is implemented through the use of Linux namespaces.
 - Linux namespace can isolate resources such as processes, network stacks, and file systems.

Docker

- Docker is a platform that enables developers to package applications and their dependencies into lightweight, portable containers that can run consistently across different environments.
- The most widely used container engine.
- Other container runtimes: Podman, Singularity, etc.

Container vs. VM



source: [Containers vs. Virtual Machines \(VMs\): What's the Difference? | NetApp Blog](#)

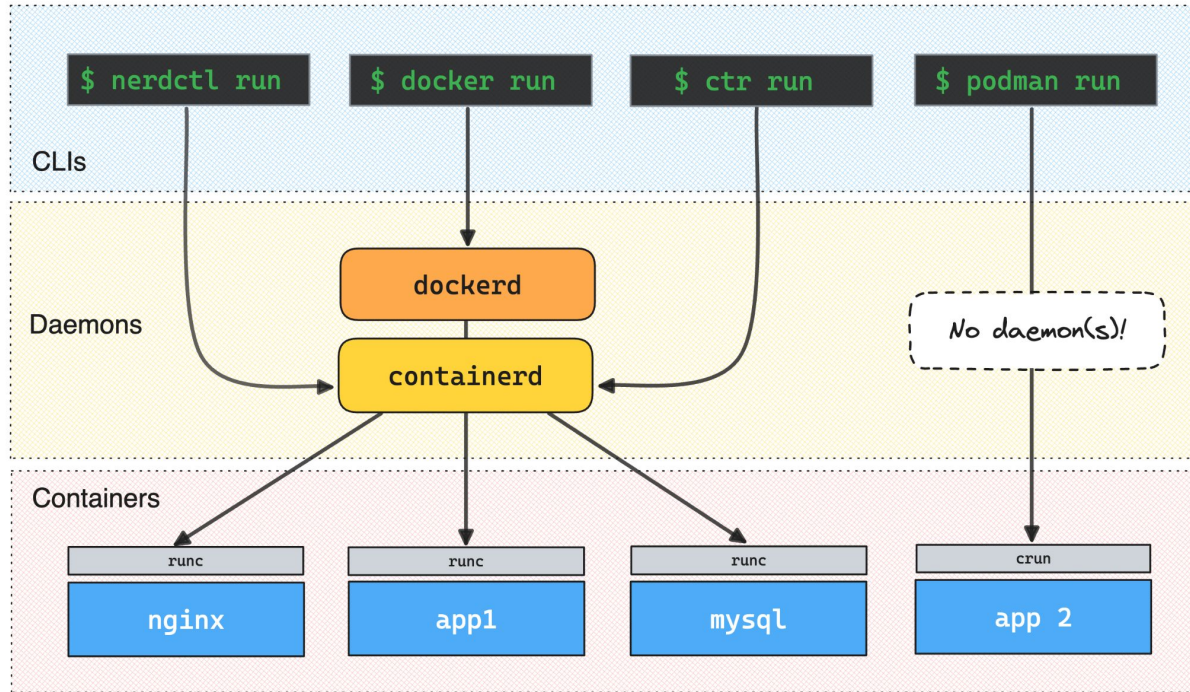
Container vs. VM

- VM has fully independent environment, including OS.
- Container is faster and more lightweight since it shares the kernel.
- VM is safer.
- Container is relatively reusable.

Docker, Containerd & Runc

- Docker itself is actually a CLI tool to utilize the container runtime called Containerd.
- Containerd manages the lifecycle of containers, and it uses runc to interact with containers via OCI.
 - OCI (Open Container Initiative) is a spec for containers.
 - Containerd uses shim to track the status of runc containers.
- Runc is a low-level container runtime, and it can directly utilize the kernel.

Docker, Containerd & Runc



source: [Beyond Docker: Run Containers Across Runtimes](#)

Docker Installation

Docker Installation

- Docker provides a install script that fits many Linux distributions.

```
$ curl -fsSL https://get.docker.com -o get-docker.sh  
$ sudo sh ./get-docker.sh
```

- After Installation, you may need to enable and start Docker.

```
$ sudo systemctl enable docker  
$ sudo systemctl start docker
```

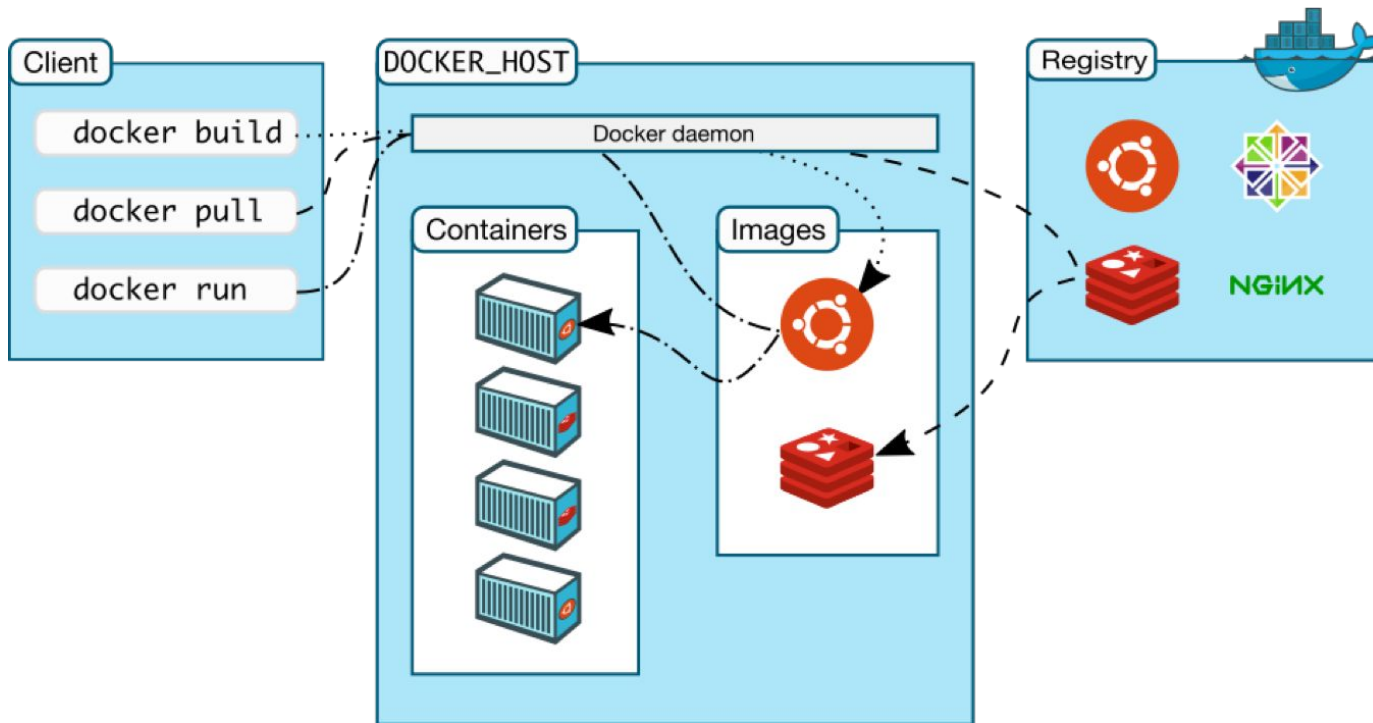
Docker Installation

- You can add users to docker group to allow them use Docker without sudo.

```
$ sudo groupadd docker  
$ sudo usermod -aG docker $USER  
$ docker -v
```

Common Commands

Docker Workflow



source: [Container workflows at Pawsey: Introduction to Docker](#)

docker pull

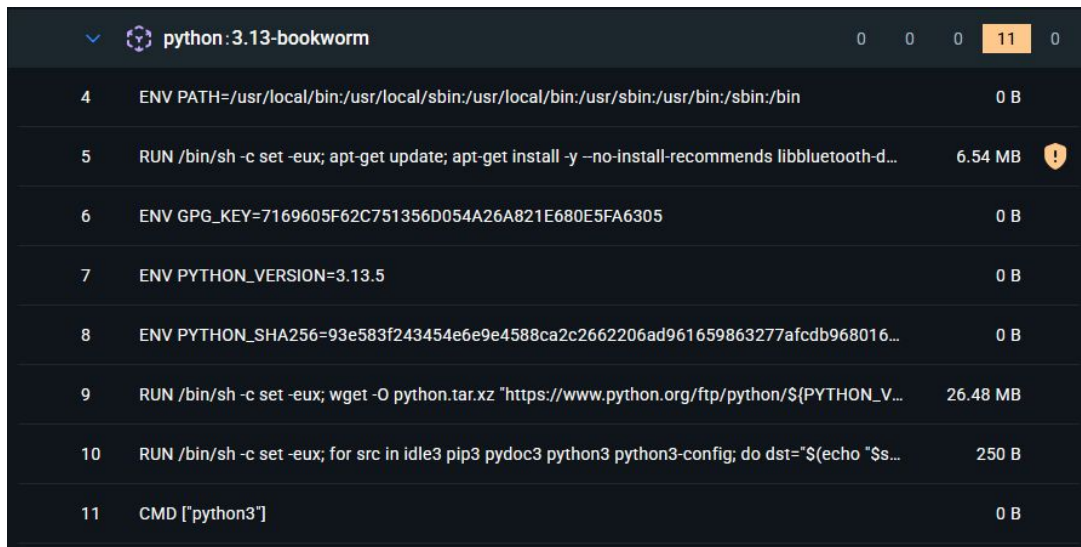
- Before launching a container, we first need to have an image.
- The command `docker pull` allows you download container image to your local storage.
- Usage: `docker pull <image>:[<tag>]`
 - You can find image from [Docker Hub](#), which is a repository of container images.
 - For example, we can `docker pull debian:trixie` to get Debian Trixie image on Docker Hub.
 - You can also use other image registries such as [AWS ECR](#) by prefix the image name with `<registry_address>/<repository_name>`.

docker pull - Container Image Tag

- The image tag contains the version and attributes of image.
- Take Debian for example:
 - `trixie`: The image of Debian Trixie
 - `trixie-slim`: The image of Debian Trixie, but more lightweight
- Take Python for example:
 - `3.13-bookworm`: The image of Python 3.13, based on Debian Bookworm
 - `3.13.5-alpine3.22`: The image of Python 3.13.5, based on Alpine 3.22

docker pull - Container Image

- A container image is composed of multiple layers.
- You can visit Image Layer Details page to see the layers.



The screenshot shows the 'Image Layer Details' page for the Docker image 'python:3.13-bookworm'. The table lists 11 layers, each with a layer number, a description of the layer's content, and its size. Layer 11 is the final layer, which is the command to run the container.

	python:3.13-bookworm	0	0	0	11	0
4	ENV PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin				0 B	
5	RUN /bin/sh -c set -eux; apt-get update; apt-get install -y --no-install-recommends libbluetooth-d...				6.54 MB	!
6	ENV GPG_KEY=7169605F62C751356D054A26A821E680E5FA6305				0 B	
7	ENV PYTHON_VERSION=3.13.5				0 B	
8	ENV PYTHON_SHA256=93e583f243454e6e9e4588ca2c2662206ad961659863277afcdb968016...				0 B	
9	RUN /bin/sh -c set -eux; wget -O python.tar.xz "https://www.python.org/ftp/python/\${PYTHON_V...				26.48 MB	
10	RUN /bin/sh -c set -eux; for src in idle3 pip3 pydoc3 python3 python3-config; do dst="\$(echo "\$s...				250 B	
11	CMD ["python3"]				0 B	

source: [python:3.13:bookworm](#)

docker run

- After obtaining the image, you can launch a container.
- Usage: `docker run <image>`
- Frequently used options:
 - `-it`: Interactive and open tty, often used together
 - `-d`: Detach to background
 - `--name <name>`: Container name
 - `--rm`: Delete on container stop
 - `-v <host path / volume name>:<container path>`: Volume mount
 - `-p <host port>:<container port>`: Port forwarding

docker run - Volumes

- A Docker volume is a persistent storage mechanism that allows containers to store and share data independently of their lifecycle, surviving restarts and deletions.
 - In short, data will be persistent even though the container is gone.
- Two volumes we often use
 - Named Volumes
 - Gives the volume a name, and they will be placed under /var/lib/docker/volumes.
 - `docker run -v my_vol:/data debian:trixie`
 - Bind Mount:
 - Binds a specific path on host to container.
 - `docker run -v ./data:/data debian:trixie`

docker exec

- When the container is launched and running, you can exec into the container.
- Usage: `docker exec <container name> <command>`
 - `<command>` is the command executed in container. It's often bash or sh.
- Frequently used options
 - `-it`: Interactive and open TTY. It's often used when the command launching a shell.

docker logs

- When a container runs into some troubles, you may want to check the logs.
- Usage: `docker logs <container name>`
- Frequently used options
 - `-f`: Show logs and wait for new logs

Other Commands

- `docker ps`: List the containers
 - `-a`: List all containers, including stopped
- `docker stop <name>`: Stop a container
- `docker restart <name>`: Restart a container
- `docker rm <name>`: Remove a container