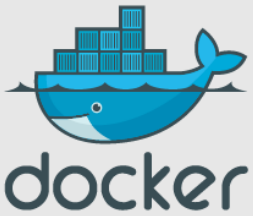


Docker

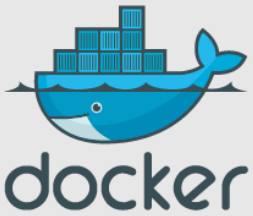
Cécile Cavet

ccavet at apc.in2p3.fr

Centre François Arago (FACe), Laboratoire APC, Université de Paris

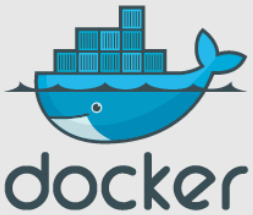
https://gitlab.in2p3.fr/cavet/tp-docker-obs/

November 20 2019



Plan

- Introduction
- Image creation and management
- Container deployment
- Docker hosts
- Hands-on
- Conclusion



Introduction

- **Container**: isolation of processes/applications.
- **Microservice**: 1 application within its specific environnement.
- The same philosophy as **Virtual Machines** (VM): isolation, share, reuse... but lighter and faster!

VM vs container

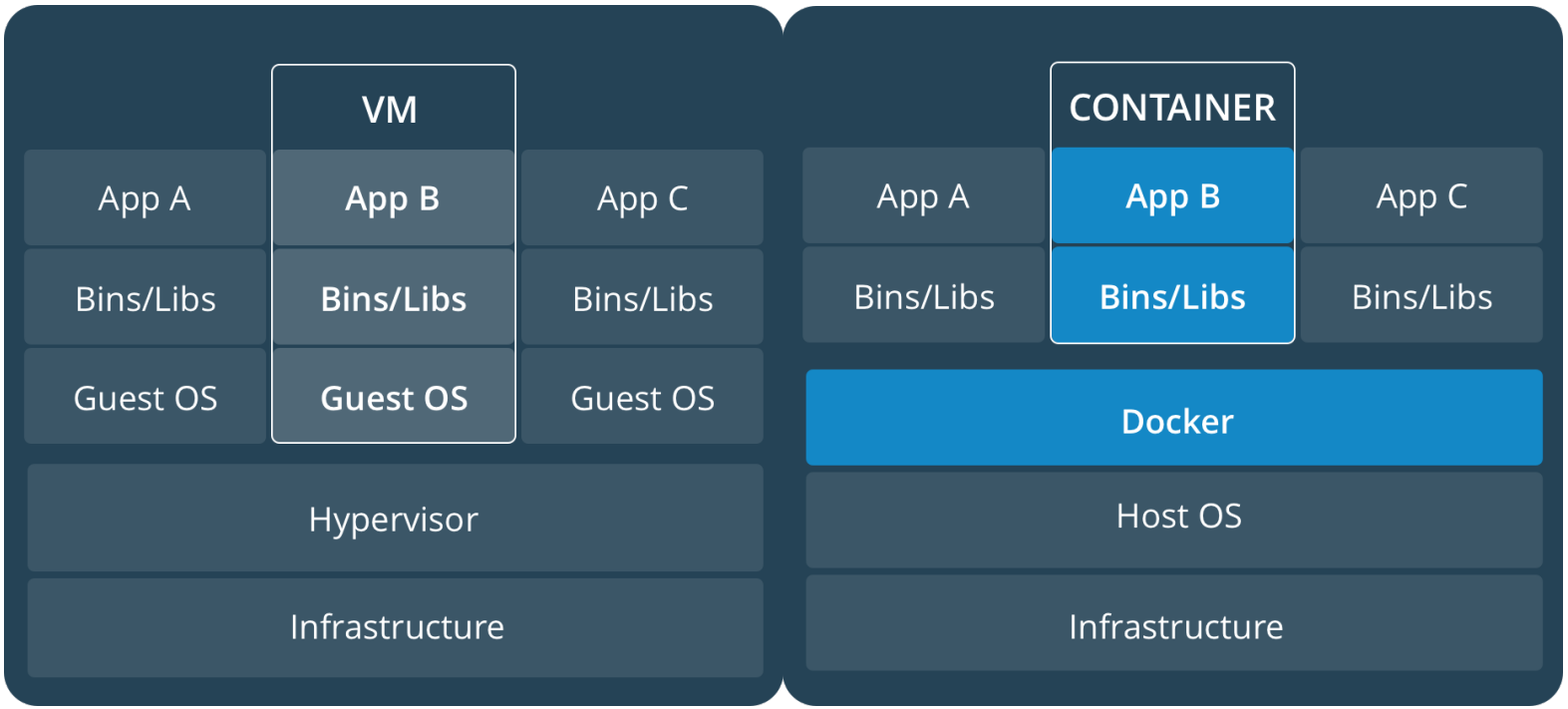
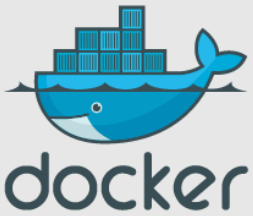


Figure: VM (left) vs Docker (right).



Docker

- The **Docker** technology is written in Go and is based on:
  - LXC (Linux Containers).
  - Union File System (amalgamated FS of layers).
  - cgroups (ressource limitation), namespaces (separated environment).

Docker history

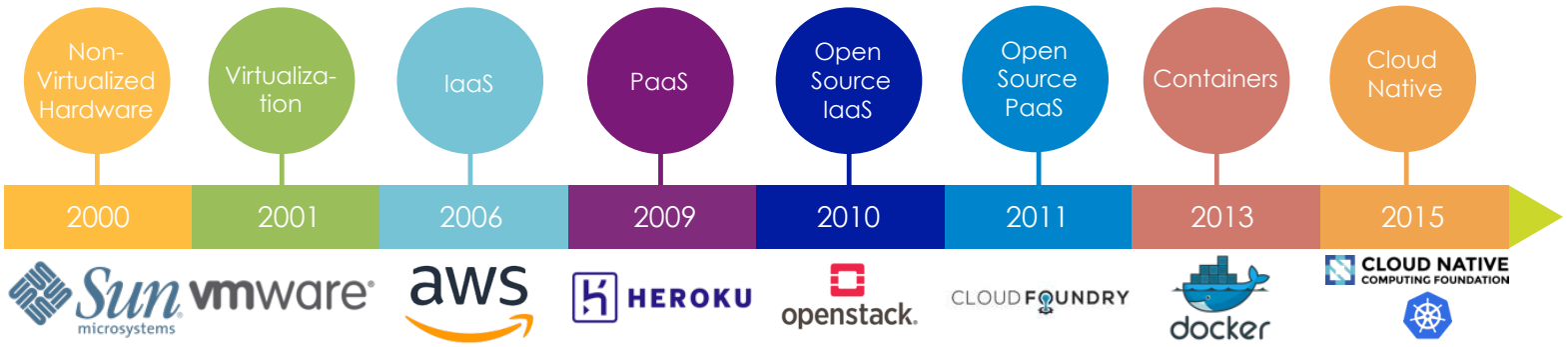
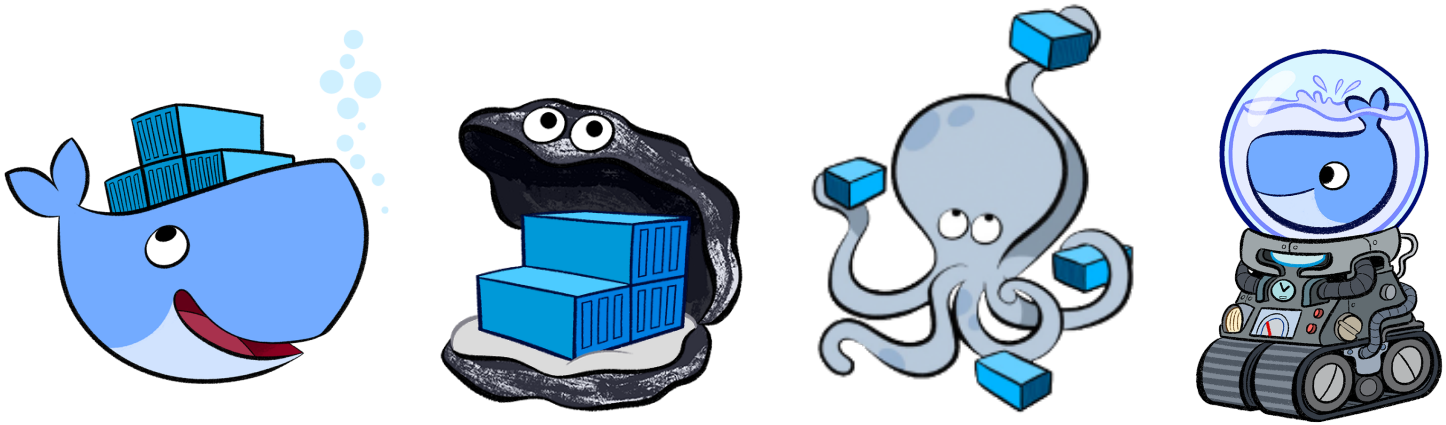


Figure: @CNCF.

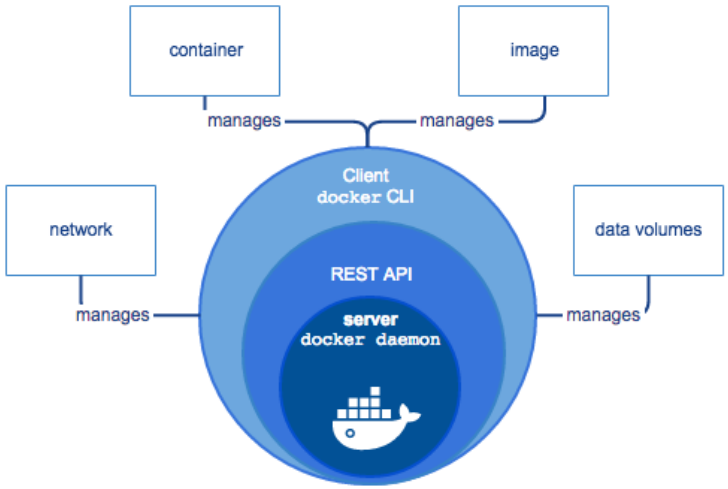
- **Docker** used to be **dotCloud** (PaaS cloud).
  - **dotCloud** started in 2008 @Montrouge.
  - The **Docker** solution has been developed since 2013 (docker-0.1).
- **Docker** is now a big project: more than 3,300 contributors.

Ecosystem



- **Engine**: daemon and CLI client.
- **Registry**: secure private registry.
- **Docker Hub/Store**: official Docker public registry.
- **Compose**: multi-container application.
- **Machine**: local and cloud VMs.

Container definition



- **Image**: a lightweight, stand-alone, executable package including the code, a runtime, libraries, environment variables, and config files.
- **Container**: an image runtime instance, what the image becomes in memory when actually executed.

User mode

Local machine:

- Linux: native **Engine**
- Non Linux: **Engine** in a light VM (HyperKit VM for macOS, Hyper-V VM for Windows).

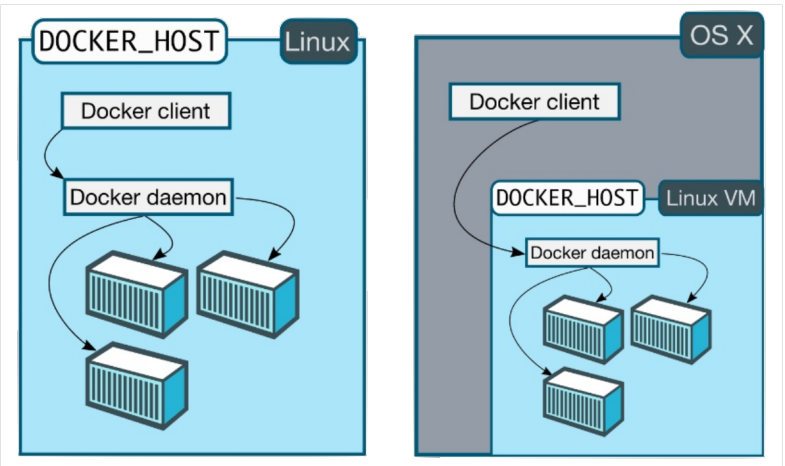
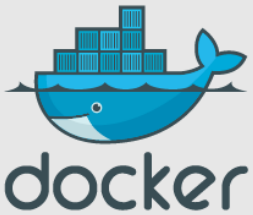


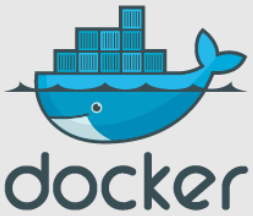
Figure: Linux vs macOS local machines.



User mode

IaaS cloud:

- Linux VM + **Engine**.
- **Machine** + Linux VM.
- **OpenStack Magnum** plugging: container orchestrator + Linux VM.



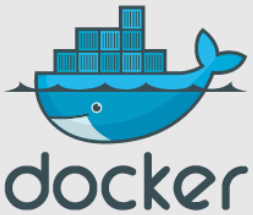
Docker installation

Version:

- Last version of Community Edition (CE): 19.03.5-ce
- Docker Engine (client/daemon) (v19.03.5-ce).
- Docker Compose (v1.24.1).
- Docker Machine (v0.16.2).

Linux:

- Package manager (yum|apt).
- Tested on CentOS 7.6.1810.



Docker installation

MacOS:

- Requirements: Mac hardware 2010+, OS El Capitan+ (10.11), 4GB of RAM.
- Docker Desktop: Engine, Compose, Machine and Kubernetes.

Windows (not tested):

- Requirements: Hyper-V.
- Docker Desktop.

Container life cycle

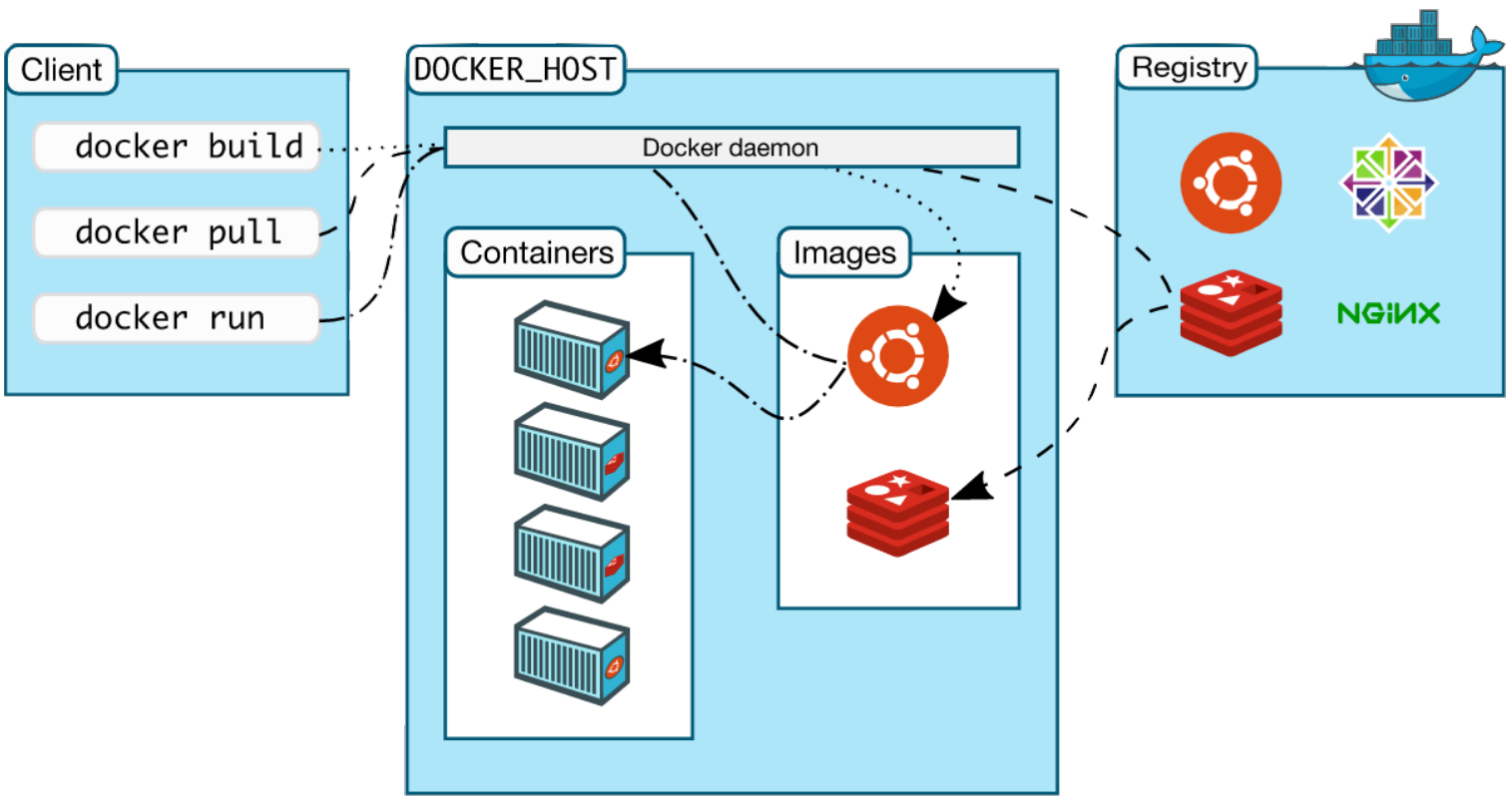
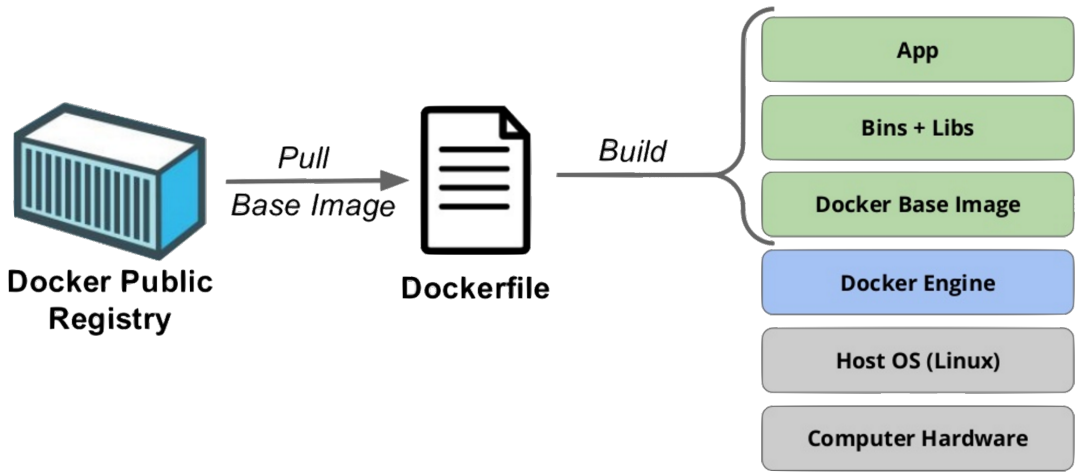


Image creation process



- **Docker Registry**: a marketplace for sharing images of various OS and applications.
- **Dockerfile**: a kind of shell script with specific instructions (RUN...).

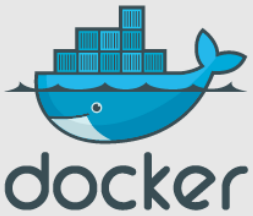
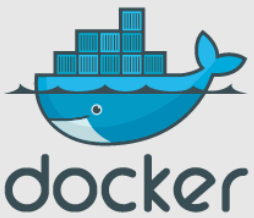


Image management

- **Docker Hub**: official Docker public registry.
- **Docker Registry**: Docker image allowing to provide a Docker registry.
- **GitLab Registry**: Docker private registry provided by GitLab.



Docker Hub

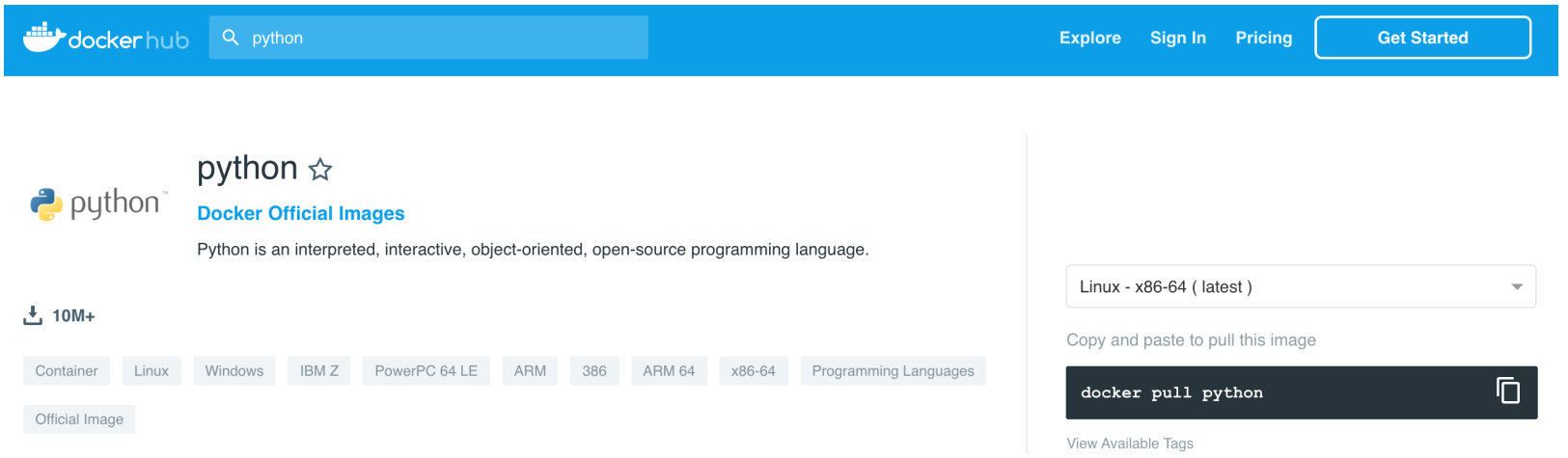
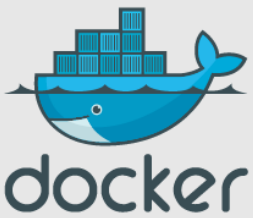


Figure: Python official images on Docker Hub.



GitLab Registry

Registry linked to a project:

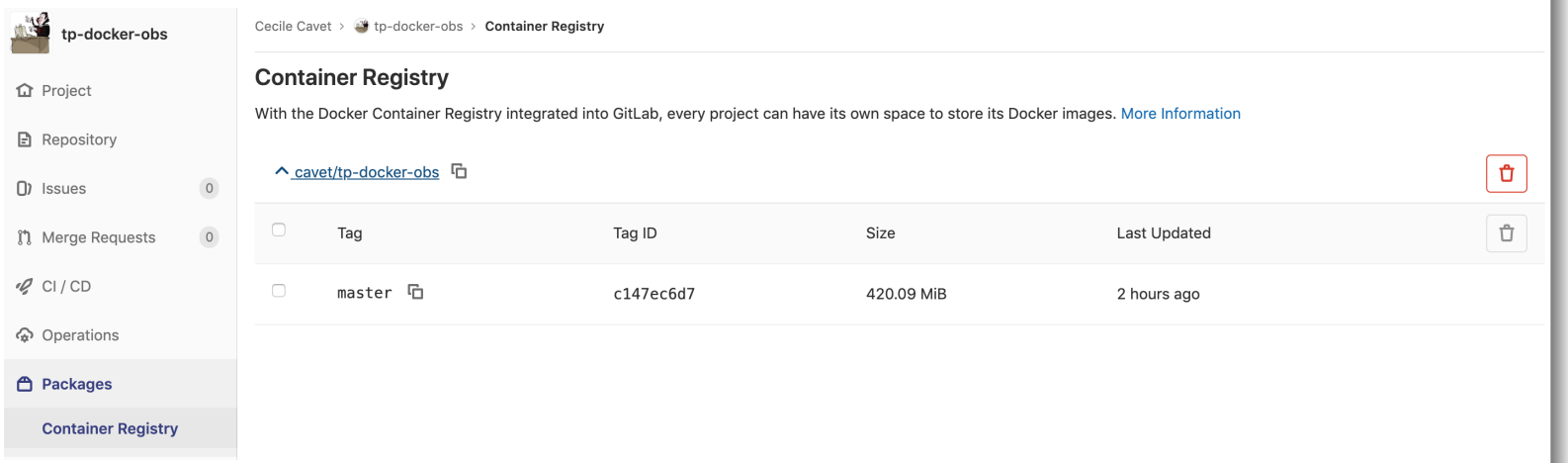
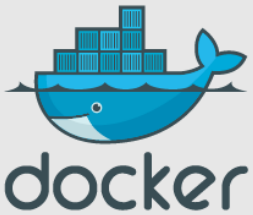
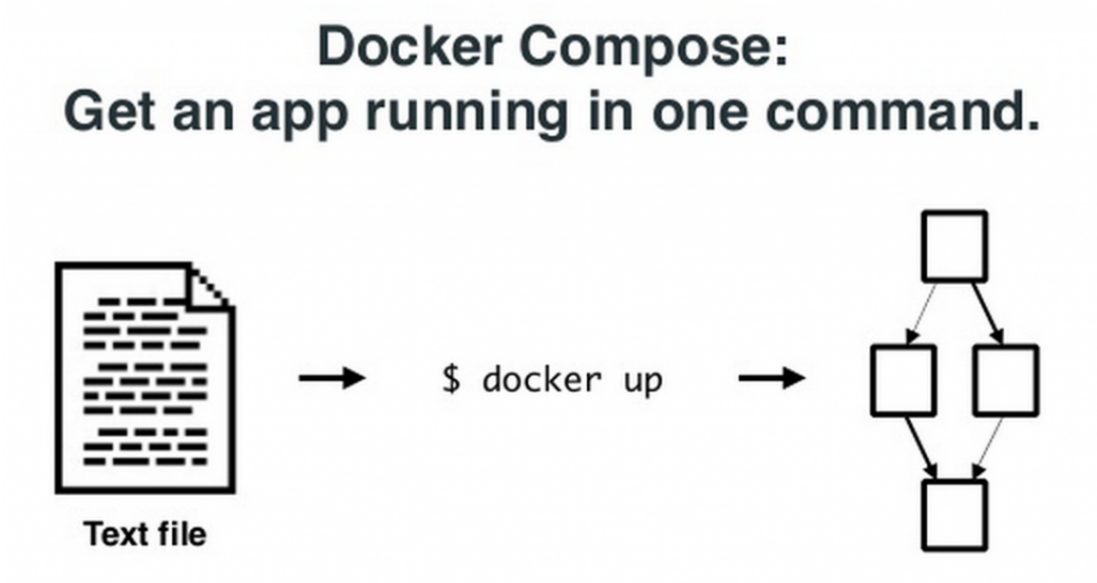


Figure: GitLab Registry of the tutorial.

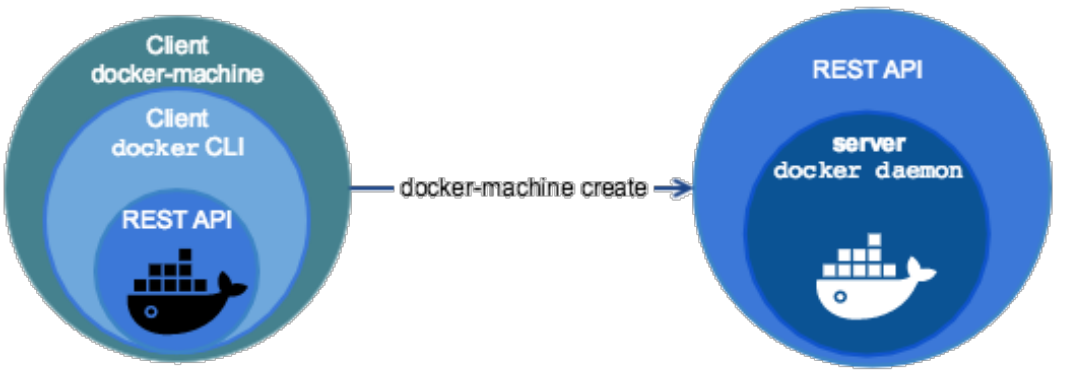


Container deployment

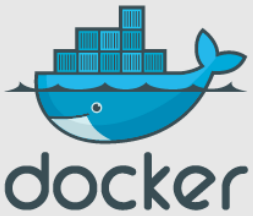


▶ **Compose file**: a YAML file allowing to automatize the building of a multi-container application.

Docker hosts



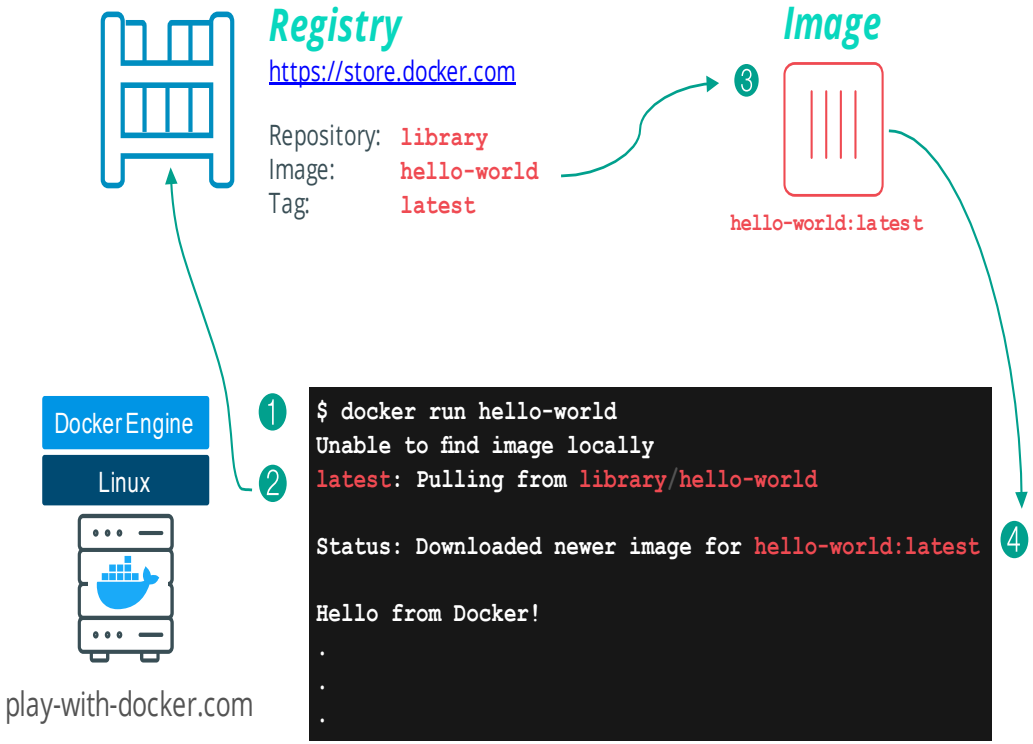
- **Docker Machine**: Docker host provisioning on local and cloud VMs.

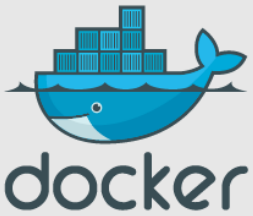


Hands-on plan

Conteneurs

Hello World: What Happened?

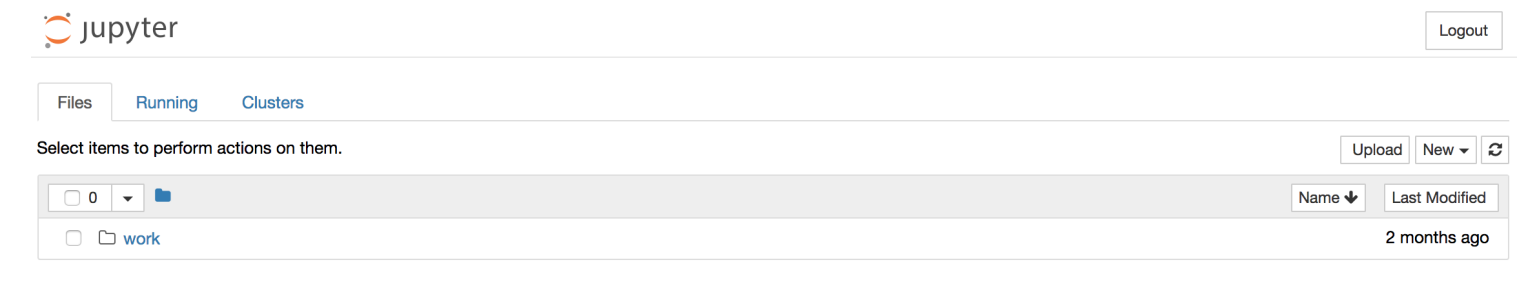




Applications

Web service:

- Image: Jupyter server for running Python Notebook.
- Environment: Conda Python 3.x.
- Librairies: Pandas, Matplotlib, Scipy...



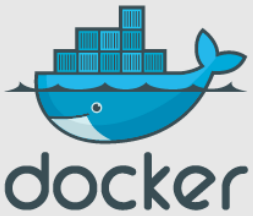


Image build

Dockerfile :

- Base image: Python
- Librairies: Python modules.
- Scientific code: LOSC_Event_tutorial.

Dockerfile.app

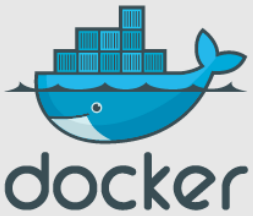
```
FROM python:3.7
MAINTAINER Cecile Cavet "ccavet@apc.in2p3.fr"

ENV PYTHONUNBUFFERED 1

WORKDIR /app
COPY . /app/

RUN pip install --no-cache-dir -r requirements.txt

ENTRYPOINT ["python"]
CMD ["LOSC_Event_tutorial.py"]
```



Container deployment

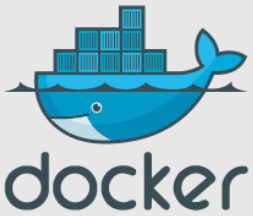
Docker Compose file:

- Jupyter Notebook image

docker-compose.yml

```yaml
version: "3"
services:
  jupyter:
    image: jupyter/scipy-notebook
    container_name: jupyter
    volumes:
      - $LOCAL_PATH:/home/jovyan/work/local
    ports:
      - "8888:8888"

volumes:
  workspace:
```



Conclusion

Useful links:

- Play-with-Docker: `https://training.play-with-docker.com`
- IN2P3 tutorials: `https://gitlab.in2p3.fr/MaitresNageurs/EnBarque`
- Ecole informatique de l'IN2P3 : conteneur en production : `https://indico.in2p3.fr/event/17124/`

The End