

ISC

High Performance

REINVENTING

HPC

MAY 12 – 16, 2024 | HAMBURG, GERMANY

Containers and container orchestration systems

Oleksandr Shcherbakov (HLRS)



Outline

- Containers
- Docker
- Security of containers
- Container orchestration
- Hands-on I



Timeline of Containerization



Timeline of Containerization

- 1979—chroot



Timeline of Containerization

- 1979—chroot
- 2000—FreeBSD Jails
- 2001—Linux VServer
- 2004—Solaris Containers
- 2005—OpenVZ
- 2007—Cgroups



Timeline of Containerization

- 1979—chroot
- 2000—FreeBSD Jails
- 2001—Linux VServer
- 2004—Solaris Containers
- 2005—OpenVZ
- 2007—Cgroups
- 2008—LXC
- 2013—Docker

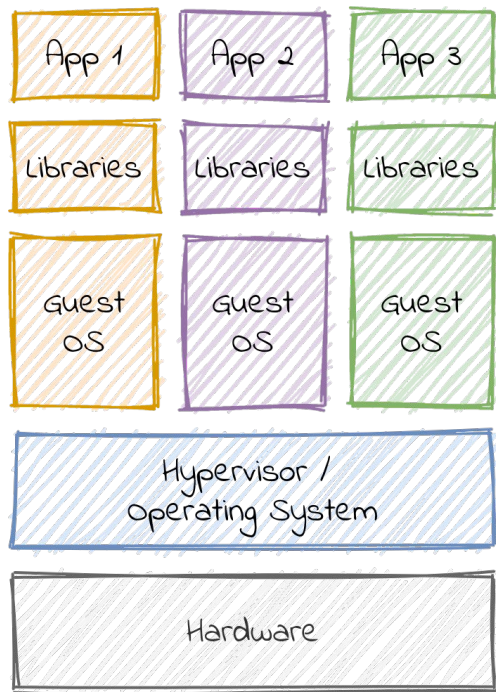


Timeline of Containerization

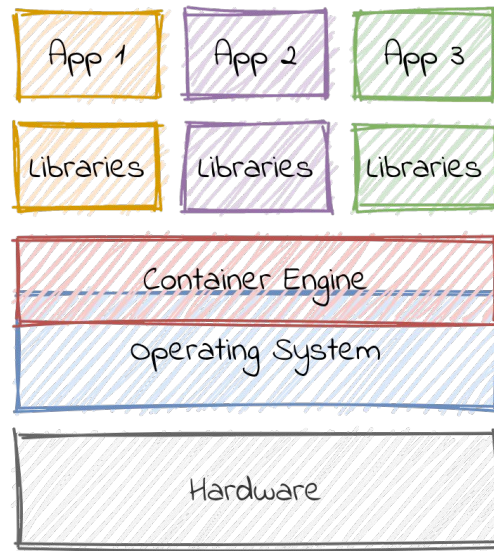
- 1979—chroot
- 2000—FreeBSD Jails
- 2001—Linux VServer
- 2004—Solaris Containers
- 2005—OpenVZ
- 2007—Cgroups
- 2008—LXC
- 2013—Docker
- 2015—OCI



Containers vs VMs



Virtualization



Containers



What Docker implemented

- A container image format
- A method for building container images (Dockerfile/docker build)
- A way to manage container images (docker images, docker rm , etc.)
- A way to manage instances of containers (docker ps, docker rm , etc.)
- A way to share container images (docker push/pull)
- A way to run containers (docker run)



Container

- cgroups + namespaces
- filesystem implementation



Running a container

```
$ docker run hello-world
```



Running a container

```
$ docker run hello-world
```

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:a26bff933ddc26d5cdf7faa98b4ae1e3ec20c4985e6f87ac0973052224d24302
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/>



Running a container

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
559c1857f440	hello-world	"/hello"	5 seconds ago	Exited (0) 5 seconds ago		festive_benz



Running a container

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
559c1857f440	hello-world	"/hello"	5 seconds ago	Exited (0) 5 seconds ago		festive_benz

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world	latest	d2c94e258dcb	12 months ago	13.3kB



Running a container

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
559c1857f440	hello-world	"/hello"	5 seconds ago	Exited (0)	5 seconds ago	festive_benz

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world	latest	d2c94e258dcb	12 months ago	13.3kB

```
$ docker run -it --rm -p 127.0.0.1:8888:8888 jupyter/base-notebook
```

...

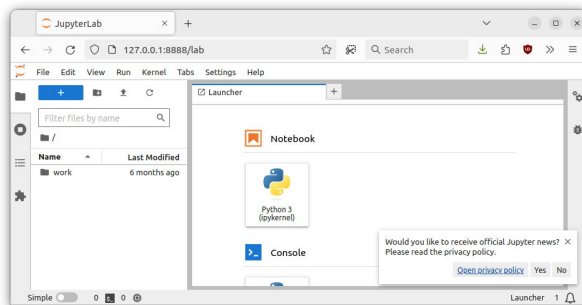
To access the server, open this file in a browser:

file:///home/jovyan/.local/share/jupyter/runtime/jpserver-7-open.html

Or copy and paste one of these URLs:

<http://7836f62fb4bc:8888/lab?token=d1073f1feb67beb0b90fe6051e9610a6d08ed23f9cde4c2b>

<http://127.0.0.1:8888/lab?token=d1073f1feb67beb0b90fe6051e9610a6d08ed23f9cde4c2b>





Dockerfile

```
FROM python:3.8
```

```
WORKDIR /home
```

```
ENV PYTHONUNBUFFERED=1
```

```
RUN apt update && apt install -y curl
```

```
RUN rm -rf /var/lib/apt/lists/*
```

```
RUN pip install "paho-mqtt<2.0.0"
```

```
COPY publisher.py /home/
```

```
ENTRYPOINT ["python3"]
```

```
CMD ["publisher.py"]
```




Dockerfile

```
FROM python:3.8
```

```
WORKDIR /home
```

```
ENV PYTHONUNBUFFERED=1
```

```
RUN apt update && apt install -y curl
```

```
RUN rm -rf /var/lib/apt/lists/*
```

```
RUN pip install "paho-mqtt<2.0.0"
```

```
COPY publisher.py /home/
```

```
ENTRYPOINT ["python3"]
```

```
CMD ["publisher.py"]
```

Check Best Practices from the [Dockerfile documentation](#)



Security



Security

```
$ docker run -it --rm ubuntu
```

```
root@8c65593bdbf3:/# id
```

```
uid=0(root) gid=0(root) groups=0(root)
```

```
root@8c65593bdbf3:/# hostname
```

```
8c65593bdbf3
```

```
root@8c65593bdbf3:/# ll /home/
```

```
total 8
```

```
drwxr-xr-x 2 root root 4096 Apr 18 2022 ./
```

```
drwxr-xr-x 1 root root 4096 Apr 28 22:20 ../
```



Security

```
$ docker run -it --rm ubuntu
```

```
root@8c65593bdbf3:/# id
```

```
uid=0(root) gid=0(root) groups=0(root)
```

```
root@8c65593bdbf3:/# hostname
```

```
8c65593bdbf3
```

```
root@8c65593bdbf3:/# ll /home/
```

```
total 8
```

```
drwxr-xr-x 2 root root 4096 Apr 18 2022 ./
```

```
drwxr-xr-x 1 root root 4096 Apr 28 22:20 ../
```

```
$ docker run -it --rm -v /:/host_root ubuntu
```

```
root@68bf8ffda46e:/# ll /host_root/home/
```

```
total 32
```

```
drwxr-xr-x 5 root root 4096 Aug 25 2023 ./
```

```
drwxr-xr-x 21 root root 4096 Apr 8 08:17 ../
```

```
drwxr-xr-x 4 root root 4096 Aug 25 2023 VMs/
```

```
drwxr-x--- 41 1000 1000 4096 Apr 26 13:24 myuser/
```

```
drwx----- 2 root root 16384 Aug 23 2023 lost+found/
```



Security (continued)

- User namespaces



Security (continued)

- User namespaces
 - [CVE-2009-1338](#), [CVE-2014-4014](#), [CVE-2016-8655](#), [CVE-2017-7184](#),
[CVE-2018-18955](#), [CVE-2019-5736](#), [CVE-2021-22555](#), [CVE-2022-0185](#),
[CVE-2022-0492](#)
- Other related CVEs
 - [CVE-2020-14386](#) - CAP_NET_RAW
 - [CVE-2021-32635](#) - Singularity
 - [CVE-2021-29657](#) - KVM
 - [CVE-2022-25636](#) - network_namespaces
 - [CVE-2022-32250](#), [CVE-2022-34918](#) - network_namespaces + nf_tables
 - [CVE-2022-4378](#) - network_namespaces
 - [CVE-2023-30549](#) - Apptainer
 - [CVE-2024-1753](#) - Podman/Buildah
 - [CVE-2024-21626](#) - runc
 - [CVE-2024-23651](#), [CVE-2024-23652](#), [CVE-2024-23653](#), [CVE-2024-23650](#),
[CVE-2024-24557](#) - Docker



Advanced topics

- Security
 - Containers are designed for dependencies isolation, not for users isolation
 - Most runtimes have binaries with setuid and/or rely on namespaces
 - This might be problematic on shared systems like HPC
- Hardware optimization
 - Images are usually generic
 - Builds for different architectures might be needed
 - Use of hardware acceleration
- Reproducibility
 - Two consequent builds may produce different images
 - Caching



Container orchestration





Orchestration tasks

- Deployment and scheduling
- Networking
- Load balancing, traffic routing and service discovery
- Health monitoring and self-healing
- Scaling



Orchestration systems

- **Kubernetes**
- Docker Swarm
- (Docker-compose)
- Apache Mesos
- Nomad
- xOpera





Tutorial: Compute Continuum

Sachin Nanavati

The Numerical Algorithms Group (NAG) Ltd.

sachin.nanavati@nag.com



HAICGU cluster

HAICGU : Huawei AI and Computing at Frankfurt's Goethe University (HAICGU)

28 X TaiShan 200 (Standard compute node)

ARM architecture

Maintained by Open Edge and HPC initiative (OEHI).

More details at <https://haicgu.github.io/index.html>

Compute Continuum tutorial – Hands on 1

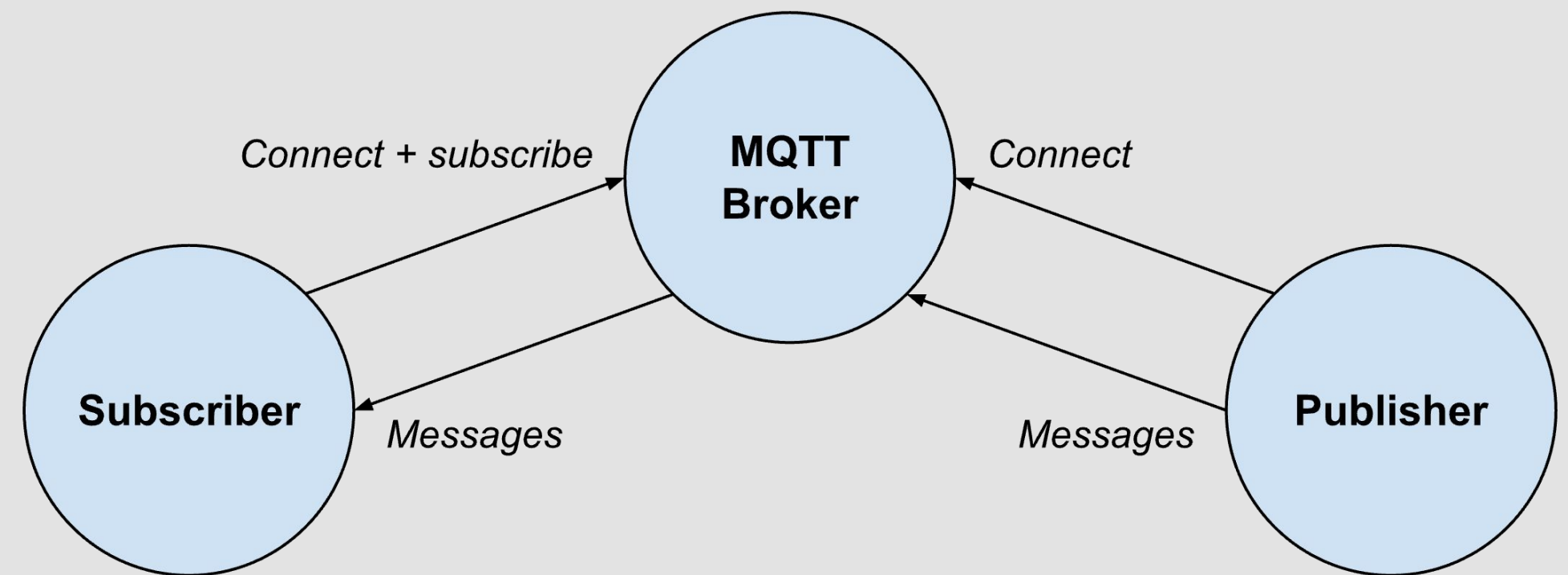
Build and Push MQTT Application

MQTT (mosquito): Light weight messaging protocol

Goal: Deploy on the cluster a MQTT publisher and subscriber and check that they can write/read on a predefined topic.

In the GitHub repository, you will find:

- Python scripts for the publisher and subscriber.
- Dockerfiles to build the images for publisher and subscriber.
- YAML files to deploy the containers on the k8s cluster.





SSH Access to Training Cluster



1. Collect account name and passphrase
2. Download ZIP archive with SSH key pairs from <https://shorturl.at/tvF56>
3. Unzip archive and select key pair that corresponds the assigned account
4. Use this key pair to connect to login server:

```
ssh -i <path_to_private_key> -l <account_name> 141.2.112.17
```

5. From login server connect to development server:

```
ssh dev
```

Tutorial materials: https://github.com/haicgu/HPC_compute_continuum