DSRI workshop

09-2019 - Alexander Malic

Agenda

1. Short Docker recap
2. Docker vs. Kubernetes vs. OpenShift
3. OC command overview
4. Practical example

1. Short Docker recap

```
FROM ubuntu

LABEL maintainer Alexander Malic

RUN apt-get update && \
  apt-get install -y apache2 && \
  apt-get clean && \
  rm -rf /var/lib/apt/lists/*

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2

EXPOSE 80

CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]
```

Docker is a Runtime for containers defined within so called Dockerfiles.

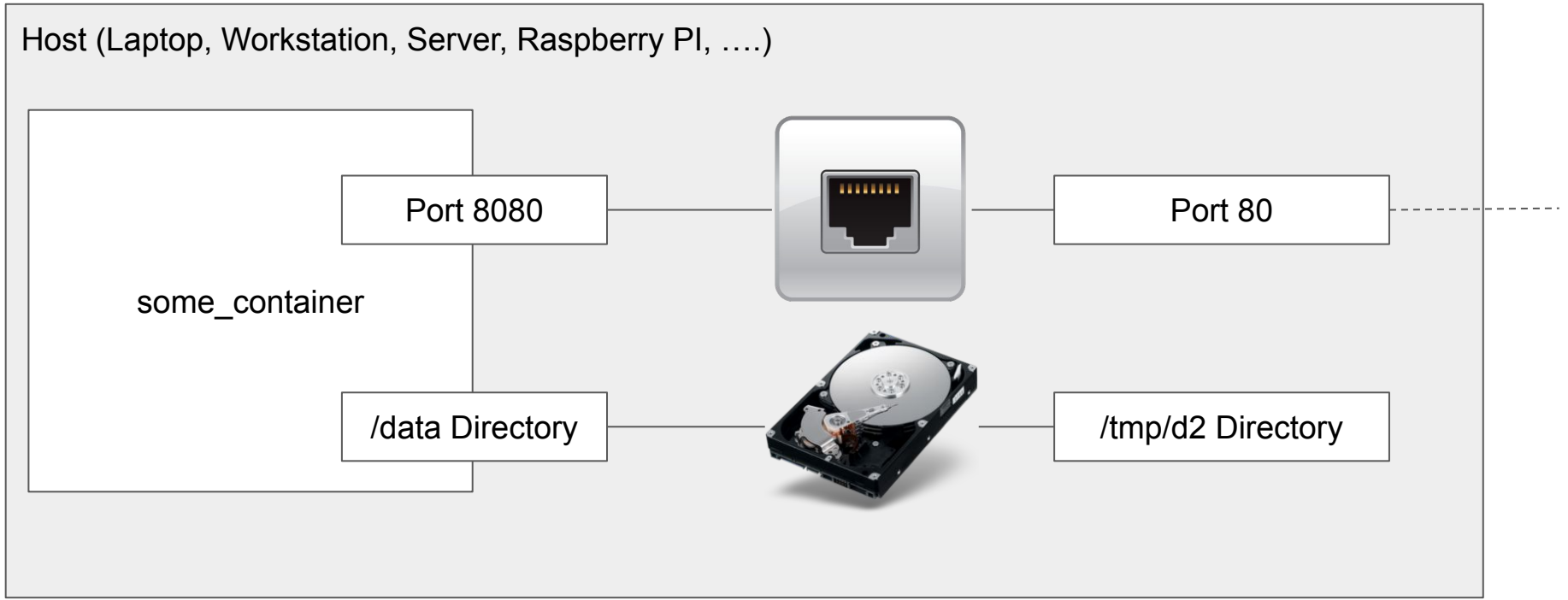
Dockerfiles can extend other Dockerfiles and add config files, or change how the container starts.

Directories from the host can be mounted into Docker containers.

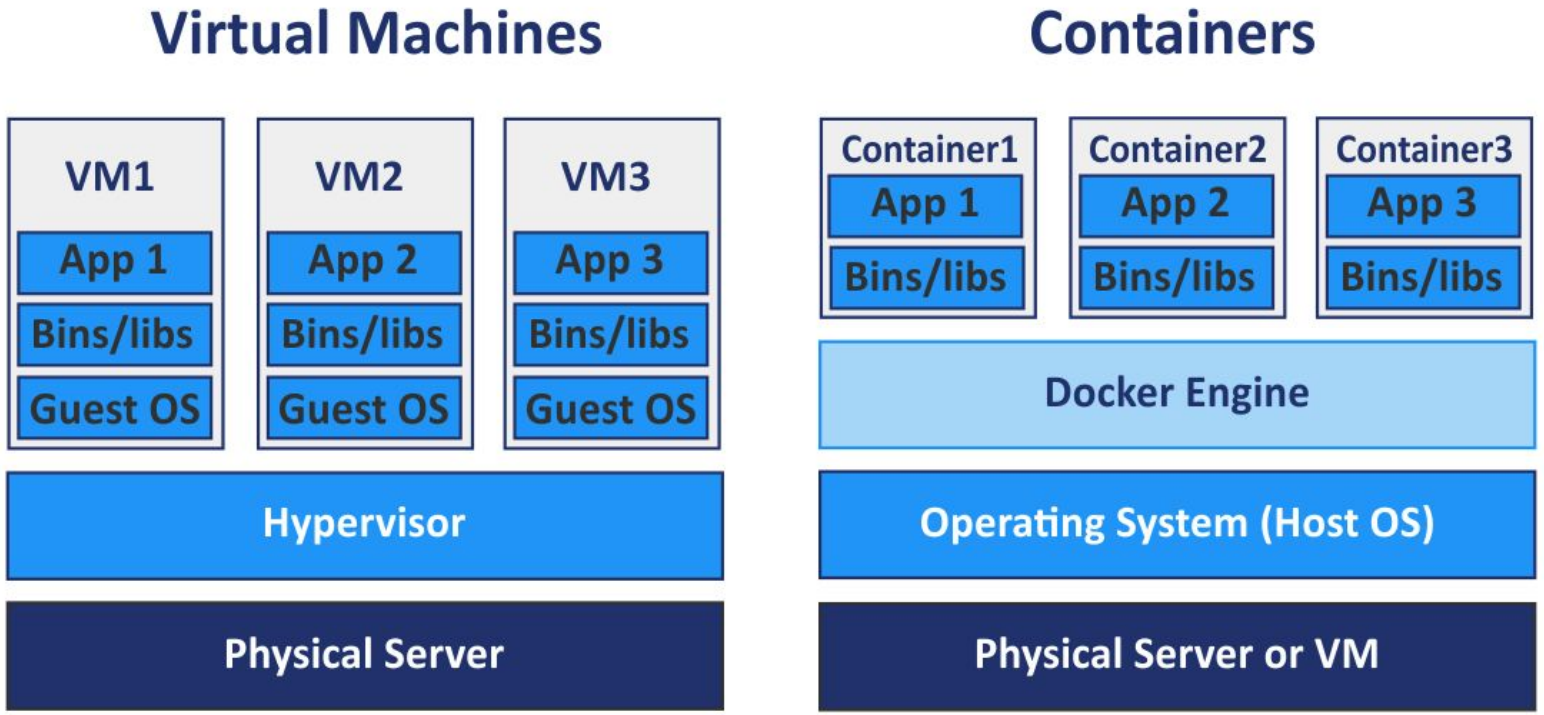
Ports of the Docker container can be mapped to ports of the host.

1. Short recap

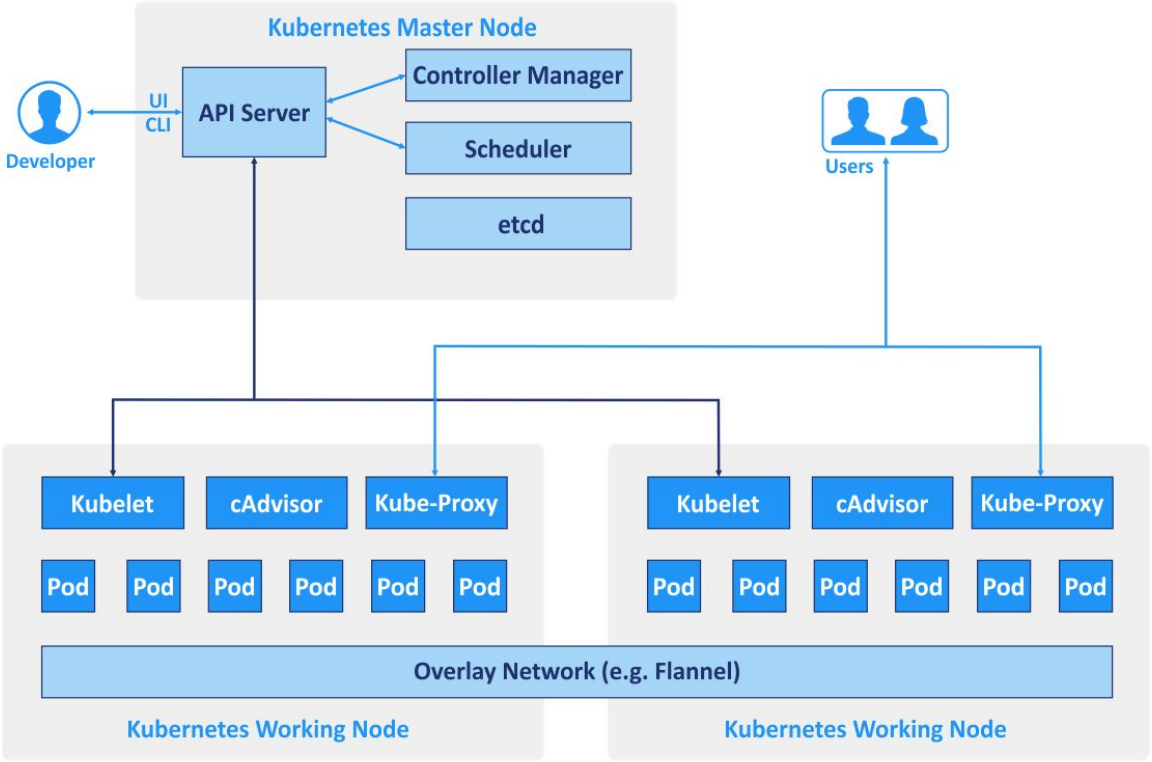
Command: `docker run -it -p 80:8080 -v /tmp/d2:/data <some_container>`



2. Docker vs. Kubernetes vs. OpenShift



2. Docker vs. Kubernetes vs. OpenShift

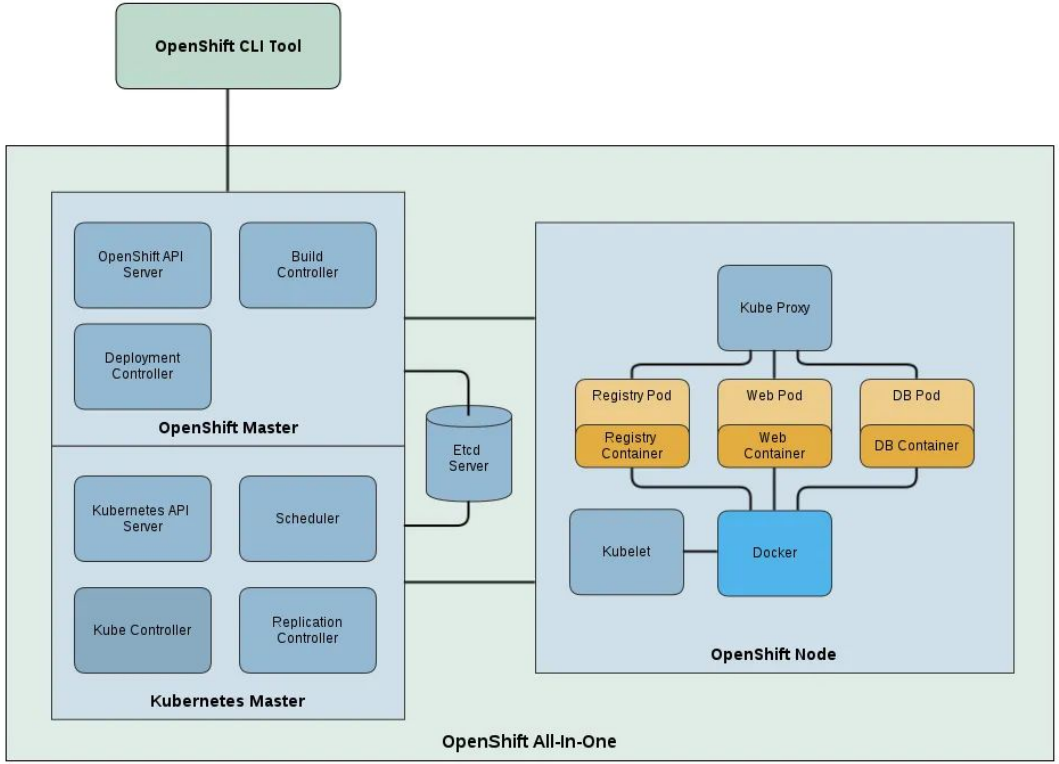


Kubernetes can run Docker containers in an orchestrated fashion on one or more host computers connected into a network.

Kubernetes adds an additional layer of abstraction around containers providing virtual networking, storage, monitoring, scheduling, …

Kubernetes makes containers also fault tolerant and scalable.

Docker vs. Kubernetes vs. OpenShift



Openshift adds a management layer on top of Kubernetes providing User management, multi-tenancy, and a simple self-service interface around Kubernetes.

The `oc` command allows users to interact with the OpenShift Api (Rest). The Api can also be utilized directly.

3. oc command overview

Command	Description
oc login <host> -u <user>	Login to openshift Api
oc get projects	List all available and authorized projects
oc project <project>	Switch to project
oc get pods	Get running pods (a pod can run one or multiple containers)
oc rsh <pod_name> <command>	Remote shell to pod
oc cp <from> <to>	Copy files from host to container or vice versa e.g. from host: oc cp <local dir> <pod>:<pod_dir> or from to host: oc cp <pod>:<pod_dir> <local dir>
oc rsync <from> <to>	Similar to rsync command on Linux to synchronize directories between container and host or the other way around

4. Practical example

We will create:

- Virtual storage
- Define secrets for easier config management
- A filebrowser application to move files easier between storage and local computer
- A Jupyterlab and/or R-Studio application accessing the same files as the storage application
- Add another user to your project via the UI

4. Practical example

filebrowser/filebrowser

Network port: `80`
Mounted volume: `/srv`

Additional volumes (to persist config and users):
`/database.db`
`/.filebrowser.json`

amalic/jupyterlab

Network port: `8888`
Mounted volume: `/notebooks`
Environment variables:

- `PASSWORD=<secret>`

rocker/rstudio

Network port: `8787`
Mounted volume: `/srv`
Environment variables:

- `ROOT=TRUE`
- `PASSWORD=<secret>`

Volume: data
Storage-class: `maprfs ephemeral`
Access mode: `RWX` (Read Write Many)

More info can be found on the web

OKD (Origin Kubernetes Distribution): https://docs.okd.io/3.11/welcome/index.html

Openshift documentation: https://docs.openshift.com/container-platform/3.11/welcome/index.html