Cristi Cernat

https://www.linkedin.com/in/ccernat/

Tue, 26 May 2020

Solution Architect Software and Electronics

@ Delphi Technologies, "*global enterprise focused on innovating propulsion systems and aftermarket solutions so that passenger cars and commercial vehicles drive cleaner, better and further.*"

Setting expectations & credits

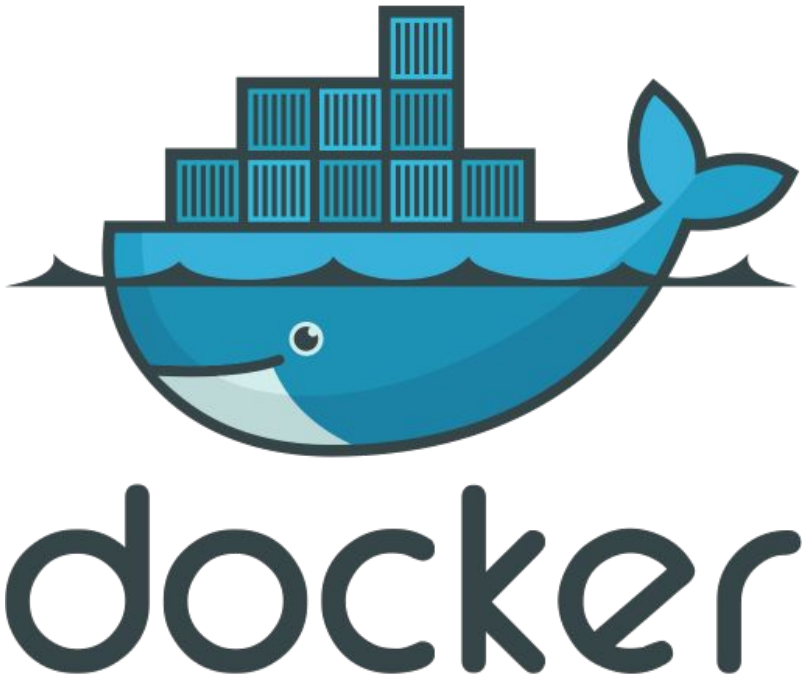
Before going further, I would like to give credits to all those great creators of tech like Docker developer teams, R developers (rocker team, ..) and their communities.

This presentation aims to build some fundamentals on Docker for R users, especially the R-Ladies Bucharest community.

It’s meant to give a flavor, not all elements.

Agenda

- What Docker is?
- Key Benefits and concepts
- Docker and VMs
- Supported platforms and tools for demo
- Demo



is a tech

No, not a salt bae recipe :-)

What can we have inside a Docker ~~recipe~~ image ?

Source code

All necessary elements to actually do something useful with the source code

Data

- end user data
- configurations
- test data
- databases of various kinds
- ...

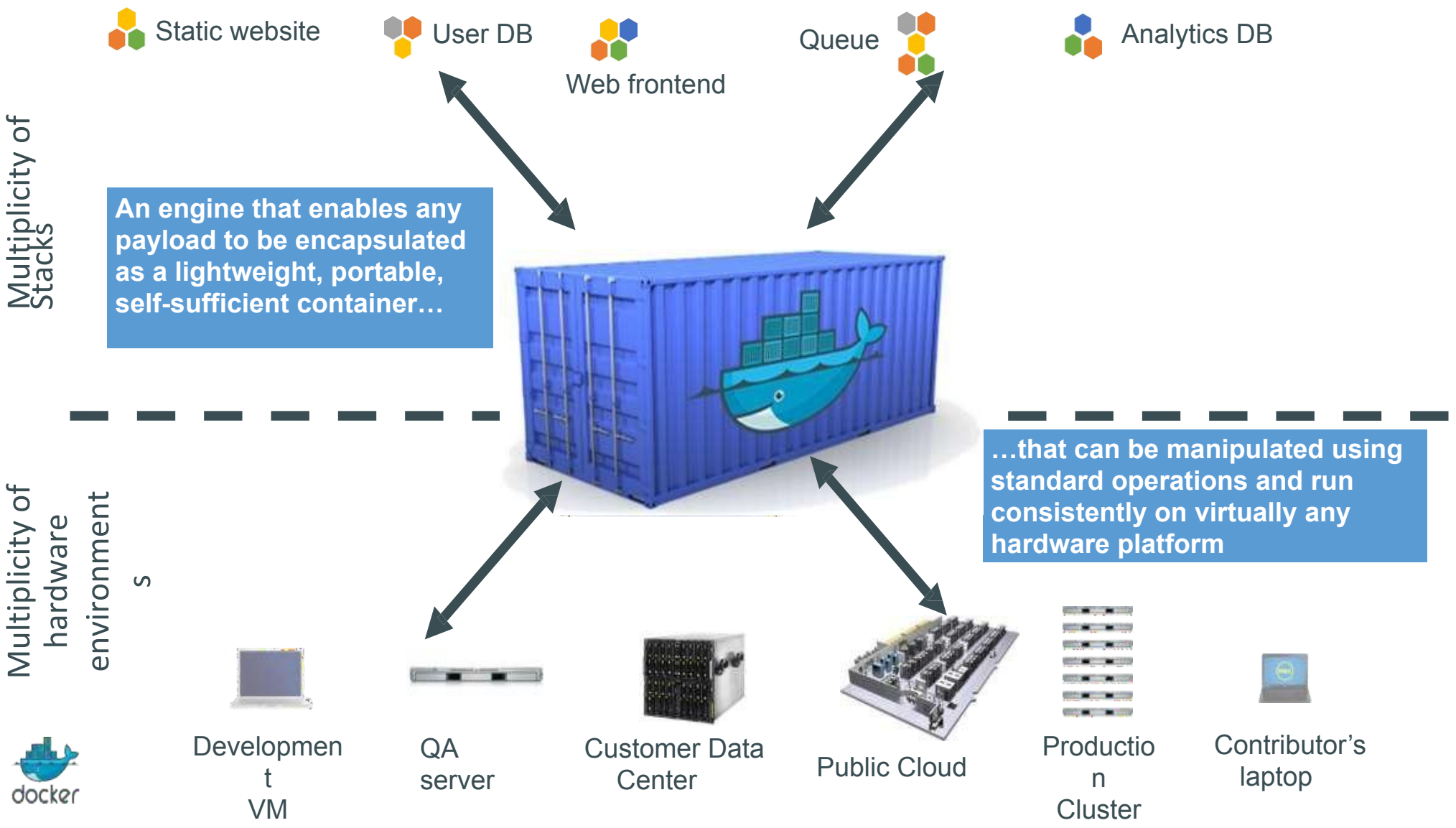
Initialization scripts

The order in which to use all "ingredients" inside image

Tools (binaries, apps with UI/UX, …)

Possibly other things ...

Docker is a shipping container system



Top 3 Docker benefits

Speed

- No OS to boot = applications online in seconds

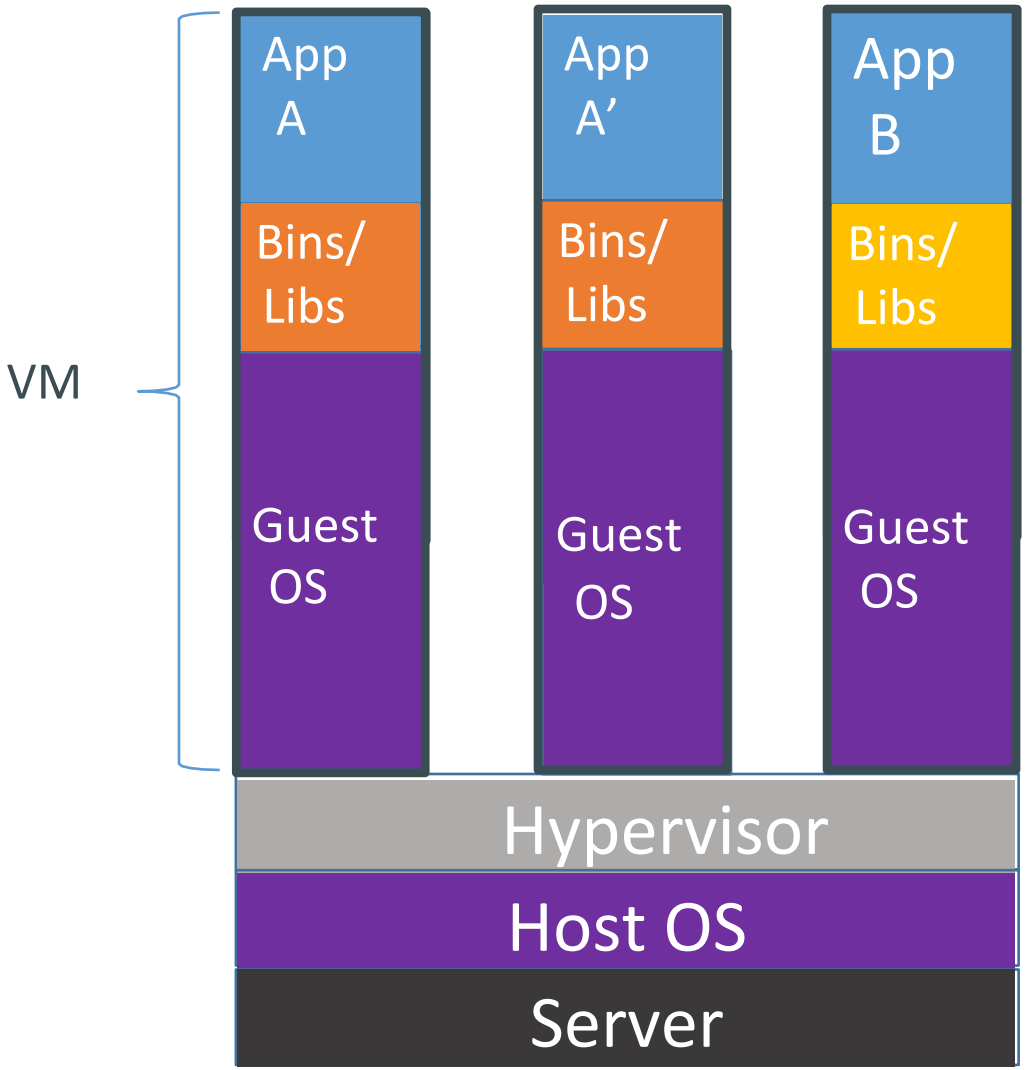
Portability

- Less dependencies between process layers = ability to move between infrastructure

Efficiency

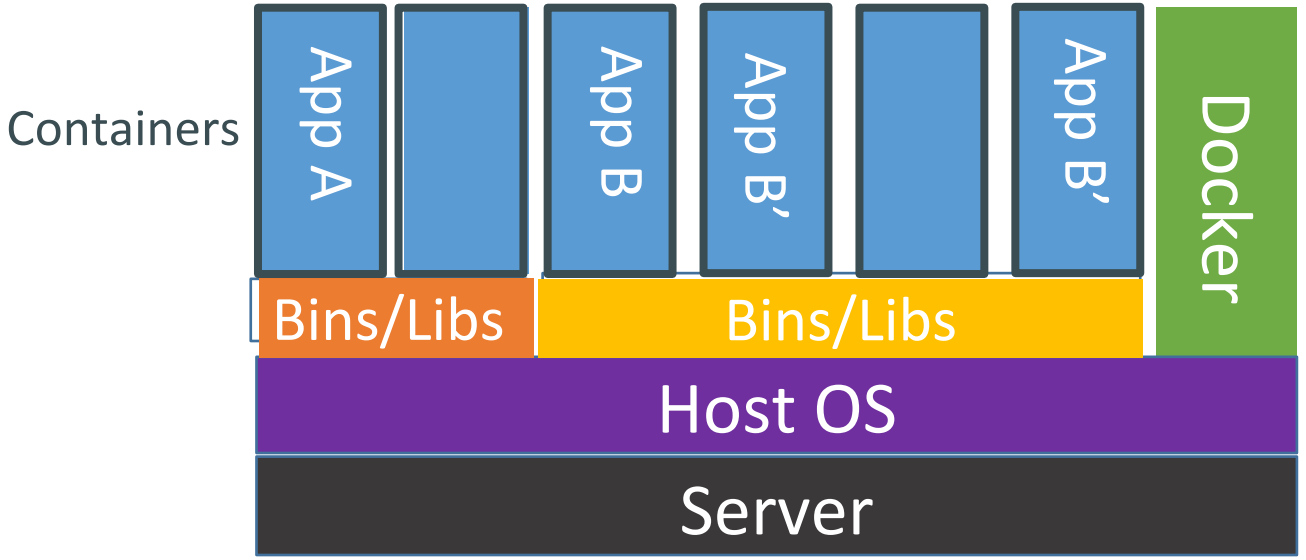
- Less OS overhead
- Improved VM density

Docker containers vs. VMs



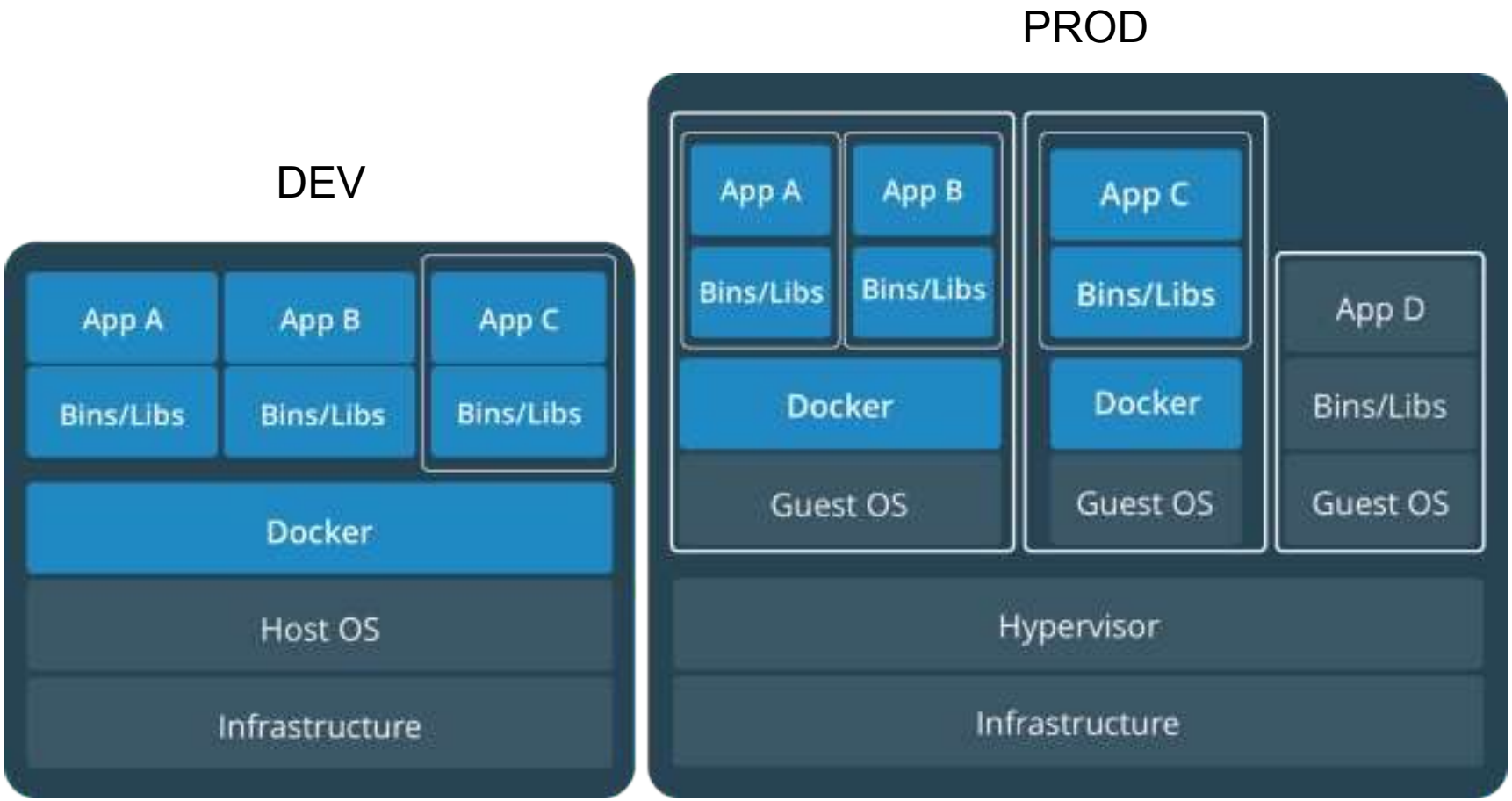
Containers are isolated, but share OS and, where appropriate, bins/libraries

…result is significantly faster deployment, much less overhead, easier migration, faster restart



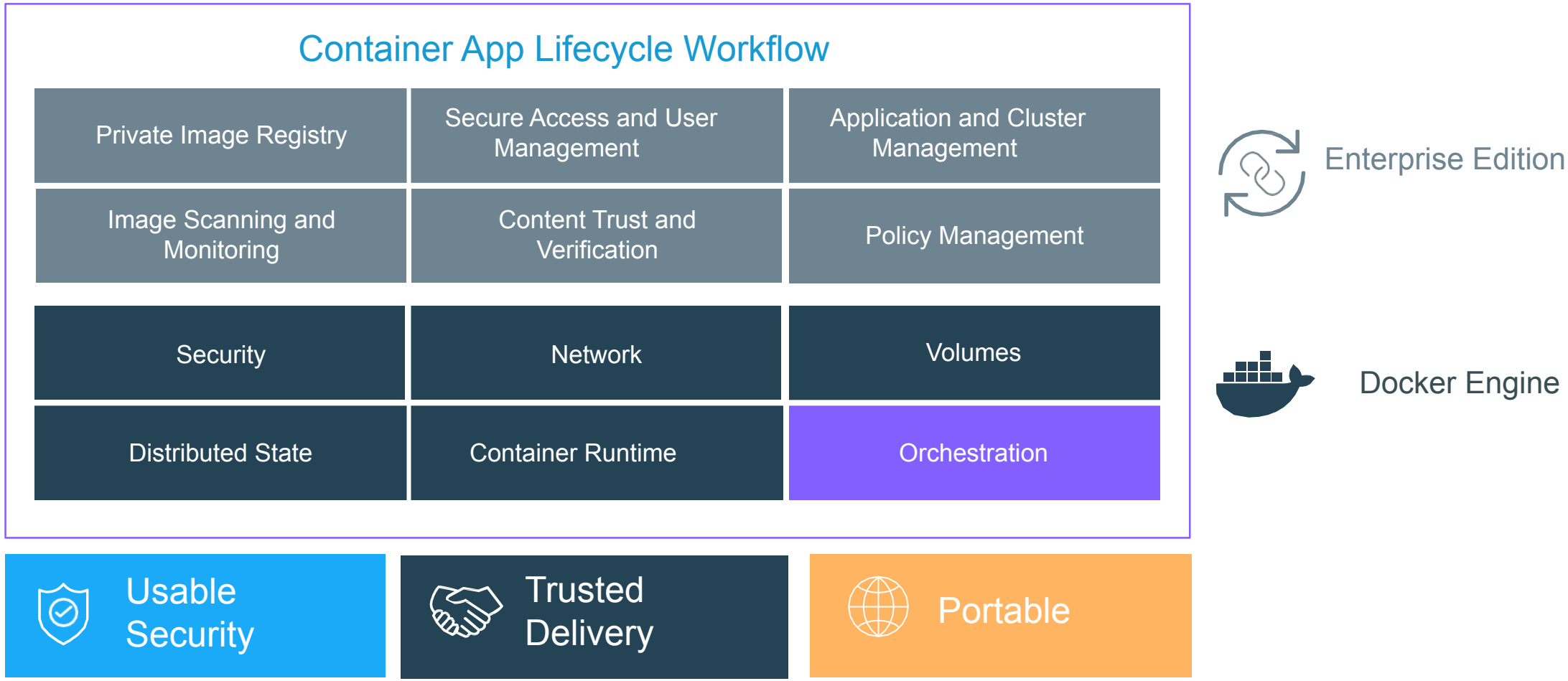
Supported platforms

Containers and VMs together



Containers and VMs together provide a tremendous amount of flexibility for IT to optimally deploy and manage apps.

Docker Engine & Enterprise Edition



Enterprise Edition is suitable for business critical production apps, has a subscription model and provides additional features (management, security…) out-of-the-box.

Docker concepts

Container

- lightweight virtual machine
- is dynamic, mutable

Dockerfile

- describe all the steps of creating and configuring a container

Image

- snapshot of a container
- is static, immutable
- reuse existing images with a precise version of OS and software

Stateless

- data is connected but is not part of the container / application
- Uses Volumes to have “containerized data”

Docker Glossary, https://docs.docker.com/glossary/

Some key Docker commands

run	creates a container over the specified image, starts it using the specified command. docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
image ls	List images docker image ls [OPTIONS] [REPOSITORY[:TAG]]
container ls	List containers docker container ls [OPTIONS]
stop	Stop one or more running containers docker stop [OPTIONS] CONTAINER [CONTAINER...]
start	Start one or more stopped containers docker start [OPTIONS] CONTAINER [CONTAINER...]
exec	Run a command in a running container docker exec [OPTIONS] CONTAINER COMMAND [ARG...]
build	Build an image from a Dockerfile docker build [OPTIONS] PATH | URL | -

And a lot more… https://docs.docker.com/engine/reference/commandline/docker/

Data in Docker

Volumes	Bind Mounts
Saved in /var/lib/docker/volumes/ on the host	Saved in /in/your/path/ on the host OS
Managed by Docker, guarantees isolation from the host machine	System processes or Docker containers can access to the data
The container see the volume as a directory	Rely on the host filesystem
can be mounted by many containers at the same time	Files and directory from the host are mounted inside the container at runtime
can be an object in the cloud (GCP , AWS , etc) , the user must use a proper driver	Managed by the host OS. Read/Write from outside the Docker container.

Multiple volumes and mount points can be combined into a single container.

tmpfs : 3rd option for memory storage, useful for ephemeral storage

Data in Docker

Create a volume

```
$ docker volume create VOLUME-NAME
```

List volumes

```
$ docker volume ls
```

Inspect volume, shows useful info about a specific volume

```
$ docker volume inspect VOLUME-NAME
```

Run a container with Ubuntu 18.04 and create a file with something inside

```
$ docker run --rm -v VOLUME-NAME:/data -it ubuntu:18.04 /bin/bash
$ echo “dummy content for test file” > /data/seed
```

Mount host directory inside the container

```
$ docker run -v /HOST-FOLDER:/host/opt --name C-NAME --rm -it ubuntu:18.04  /bin/bash
```

Why R in Docker ?!

Migrate projects between R versions, such as 3.6 to 4

Much lower resources required for the same workload compared to VM exclusive scenario

Run multiple R versions in parallel with various workloads.

Easily transfer data between running R containers (Docker has internal built-in network)

Create images for complex R environments that contain upfront all dependencies, raw data, R versions , etc…

Share and reproduce complex R environments in a fast and efficient way

Demo. Running R in Docker

Tools :

- Windows 10 Pro for host OS
- VMware® Workstation 15 Player
- Linux Ubuntu 19 for guest OS in VM
- Docker Engine (installed as stated here : https://docs.docker.com/engine/install/ubuntu/#install-using-the-repository)

Demo. Running R in Docker

Usage scenarios, most simple :

- run R file in CLI (command line interface)
- results are simple files, such as CSV
- worked out-of-the-box with one of the simplest Rocker images (rocker/r-ver)

Demo. Running R in Docker

Usage scenarios, a little more complicated :

- run R file in CLI (command line interface)
- results are files suitable for reports or end user delivery, such as HTML
- may require more dependencies in R and OS (Linux Ubuntu in this example)
- tested with rocker/rstudio image + Flexdashboard example NBA Scoring
- required to install a few extra OS Linux dependencies (pandoc and libpng-dev)

Demo. Running R in Docker

Usage scenarios, R Server :

- use R in a browser, very similar to R Studio desktop
- can require more dependencies in R and OS (Linux Ubuntu in this example)
- tested with rocker/rstudio image
- create new Rmarkdown in R Server ---> extra R packages required ---> got errors on packages install ---> required extra OS Linux dependencies (pandoc and libpng-dev)
- Key Takeaway : do not give up when you see errors, google for a fix

References

Official Docker docs, https://docs.docker.com/engine/reference/commandline/docker/

Rocker repositories on Docker Hub, https://hub.docker.com/u/rocker

An Introduction to Docker for R Users, https://colinfay.me/docker-r-reproducibility/

The Rocker Project, https://www.rocker-project.org/

Introduction to Docker, https://www.slideshare.net/Docker/introduction-to-docker-2017

Docker Simplified: A Hands-On Guide for Absolute Beginners, https://www.freecodecamp.org/news/docker-simplified-96639a35ff36/

Difference Between Docker vs VMs, https://www.educba.com/docker-vs-vms/

Docker Storage introduction, http://bit.ly/2EzR13M

Docker for beginners, https://docker-curriculum.com

Flexdashboard examples, https://rmarkdown.rstudio.com/flexdashboard/examples.html

mulțumesc. thank you

get in touch

cristicernat@gmail.com