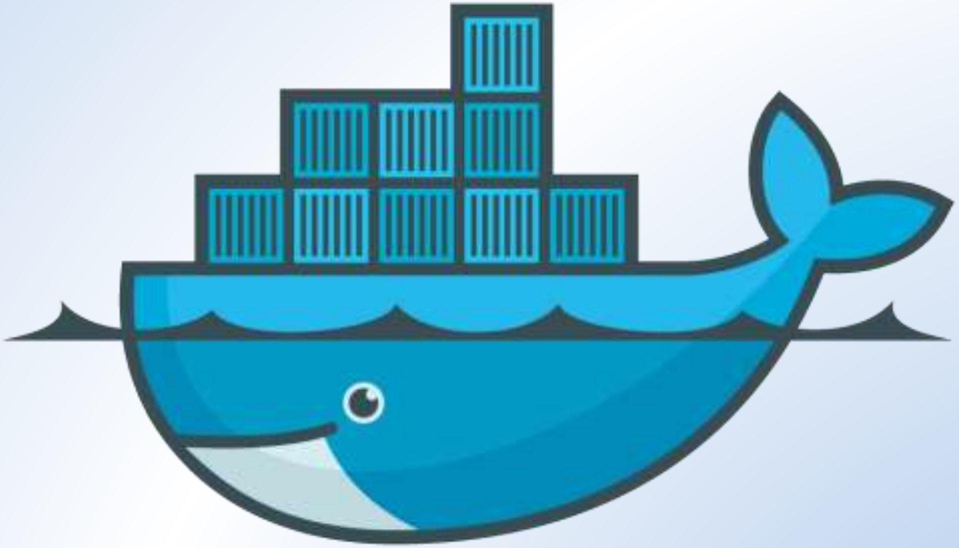


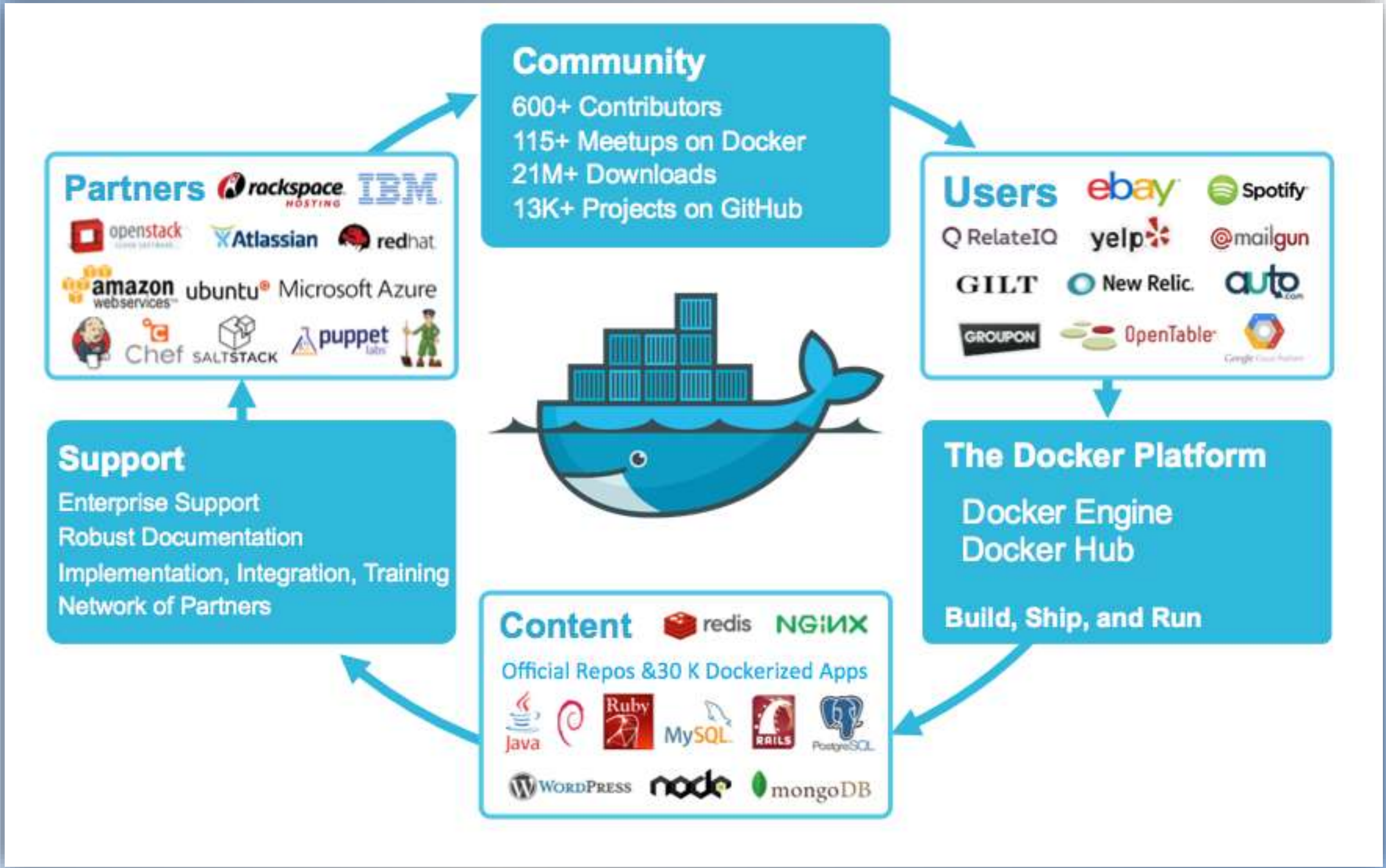
INTRO TO docker

Presenter: PhuongNQK

Goals

- Introduce you to
  - The basics of Docker
- At the end of this presentation, you will know
  - What Docker is
  - What Docker can do
  - How to work with Docker

Brief view of Docker



Community

600+ Contributors
115+ Meetups on Docker
21M+ Downloads
13K+ Projects on GitHub

Users

The Docker Platform

Docker Engine
Docker Hub

Build, Ship, and Run

Content

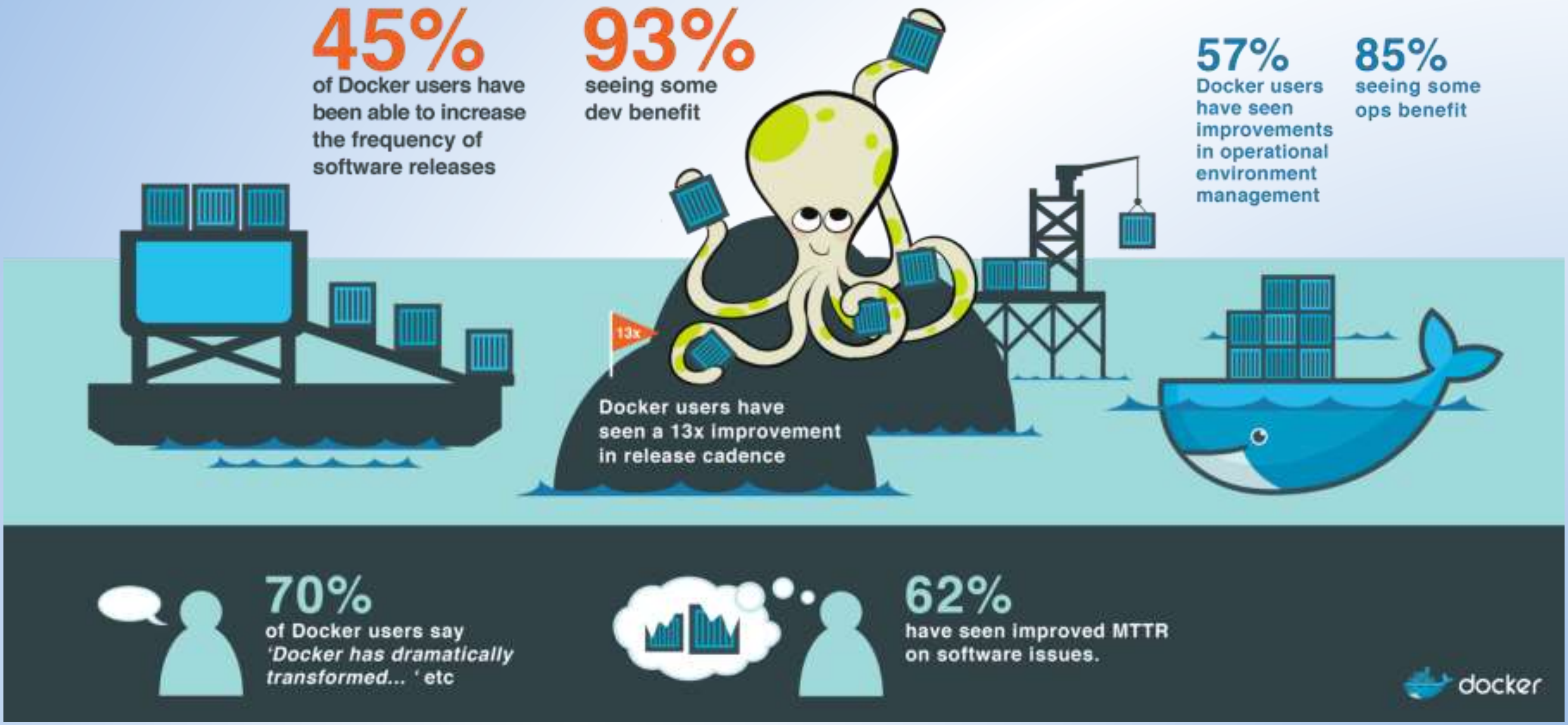
Official Repos &30 K Dockerized Apps

Support

Enterprise Support
Robust Documentation
Implementation, Integration, Training
Network of Partners

Partners

Docker has helped…



45% of Docker users have been able to increase the frequency of software releases

93% seeing some dev benefit

Docker users have seen a 13x improvement in release cadence

57% Docker users have seen improvements in operational environment management

85% seeing some ops benefit

70% of Docker users say *'Docker has dramatically transformed... '* etc

62% have seen improved MTTR on software issues.

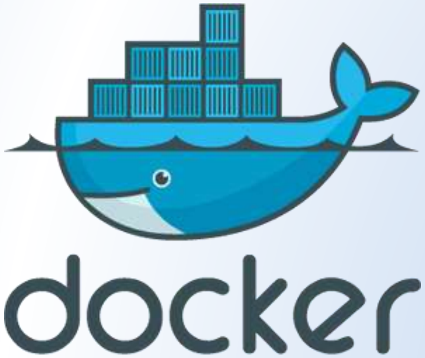
More: https://www.docker.com/survey-2016

What is Docker?

What is Docker?

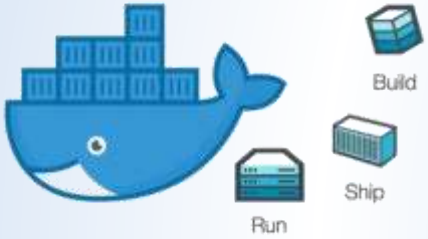
According to https://www.docker.com/what-docker

Docker is the world’s leading software container platform.



What is Docker?

Docker is a platform for developers and sysadmins to develop, ship, and run apps. It has rapidly gained popularity as one of the best tools to build, ship, and run software.

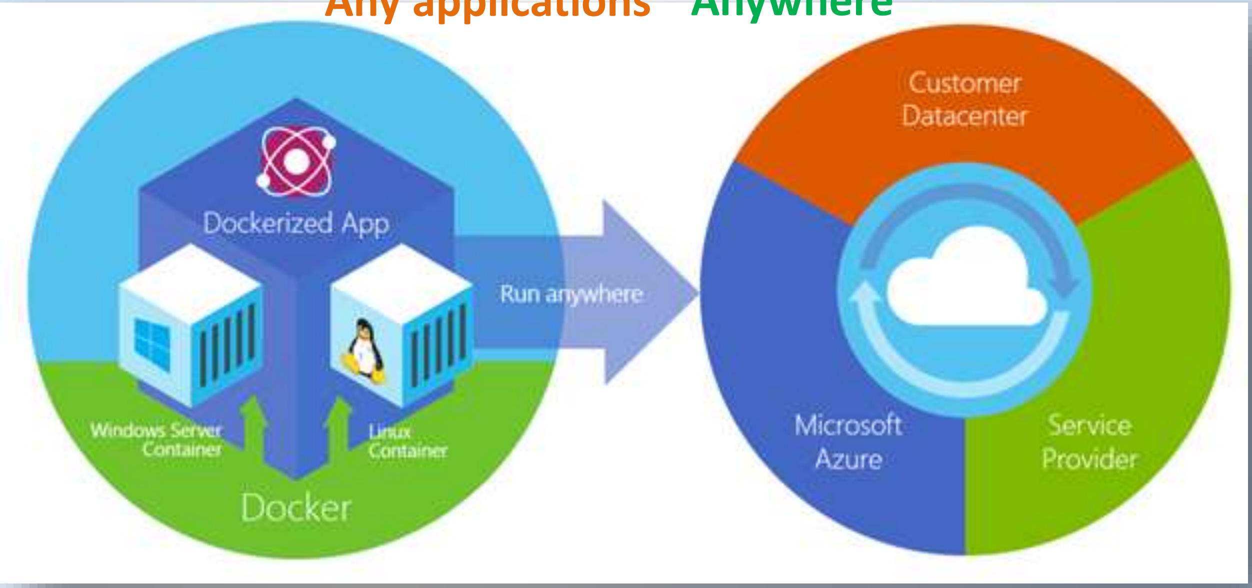


What can 🐳 do?

- Replicate the exact environment of the builds locally
- Run deployments against different environments (i.e. QA or production) consistently

BUILD → SHIP → RUN

Any applications Anywhere



Who and What can help?

DEV

Eliminate *“works on my machine”* problems when collaborating on code with co-workers

OPERATOR

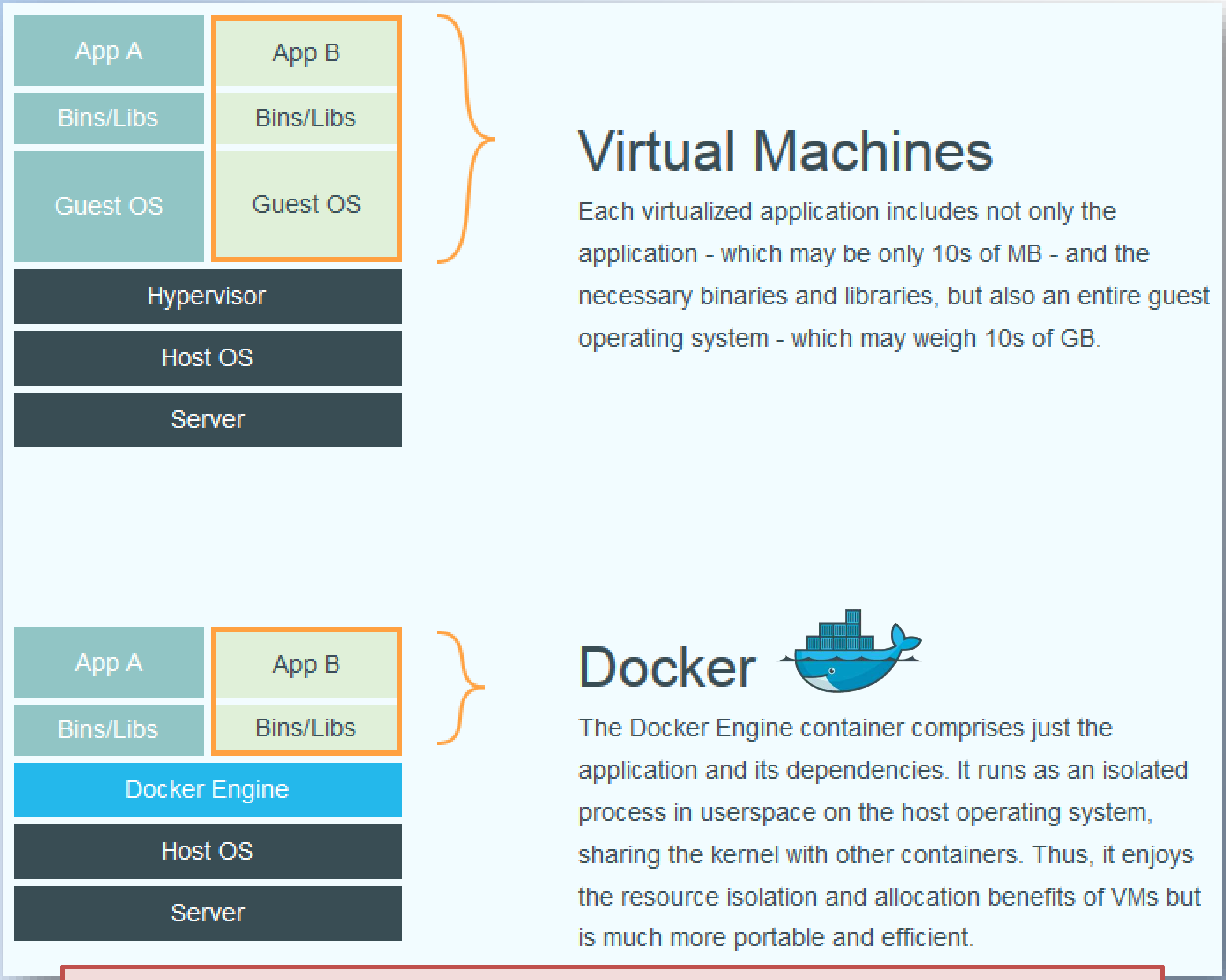
Run and manage apps side-by-side in isolated containers to get better compute density

ENTERPRISE

Build agile software delivery pipelines → ship new features faster, more securely and with confidence for both Linux and Windows Server apps

How can do that?

By refactoring VM architecture...



Virtual Machines

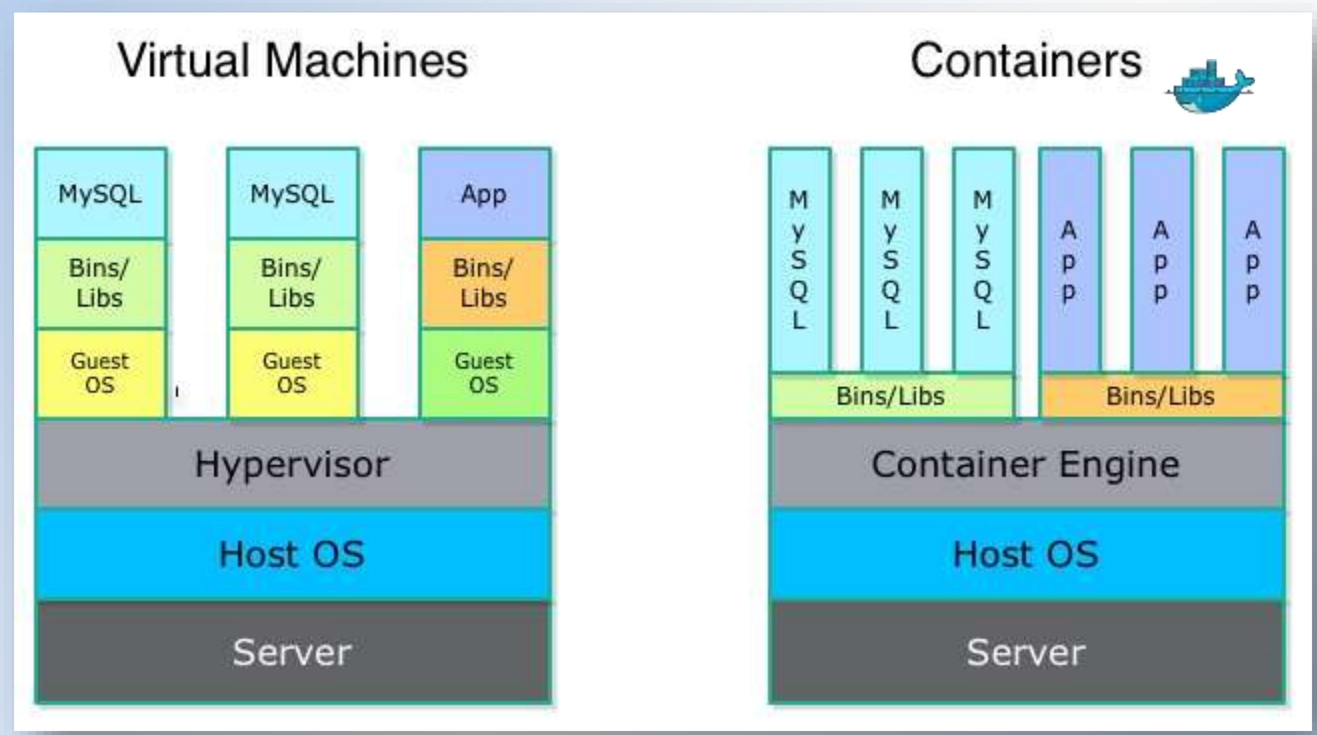
Each virtualized application includes not only the application - which may be only 10s of MB - and the necessary binaries and libraries, but also an entire guest operating system - which may weigh 10s of GB.

Docker

The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.

(From http://virtualization.info/en/news/2014/06/release-docker-docker-engine-1-0.html)

… to improve shareability



Bins/Libs and Guest OS layers cannot be shared between VMs

Bins/Libs layers are shareable between Containers

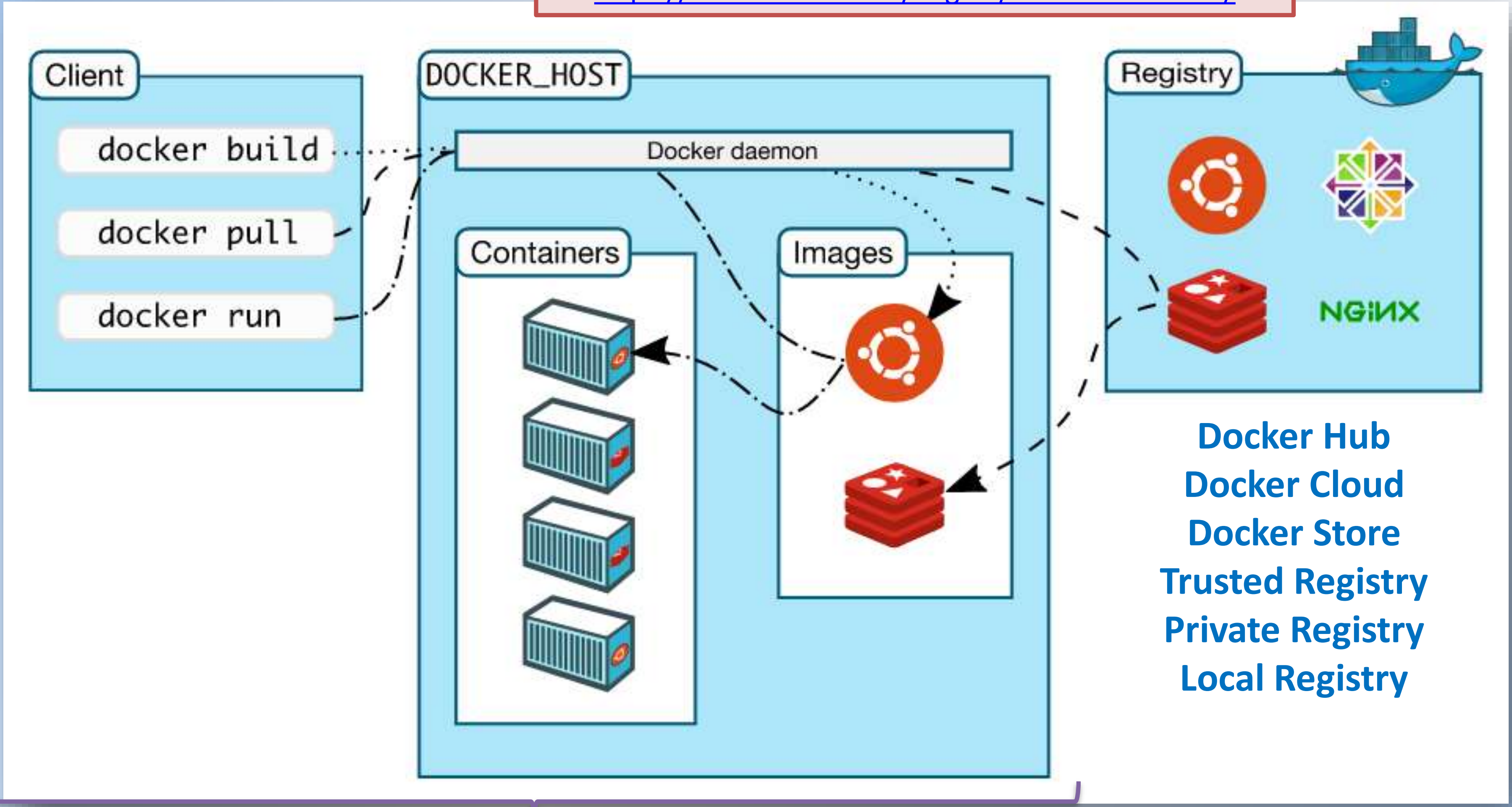
http://patg.net/containers,virtualization,docker/2014/06/05/docker-intro/

As a result, Docker has the combined strengths of:
- VMs: provide a portable environment
- Processes: are much faster and more lightweight than VMs
- app-get: can download apps from the Internet fast and easily

I want to know more.

Docker architecture

https://docs.docker.com/engine/docker-overview/



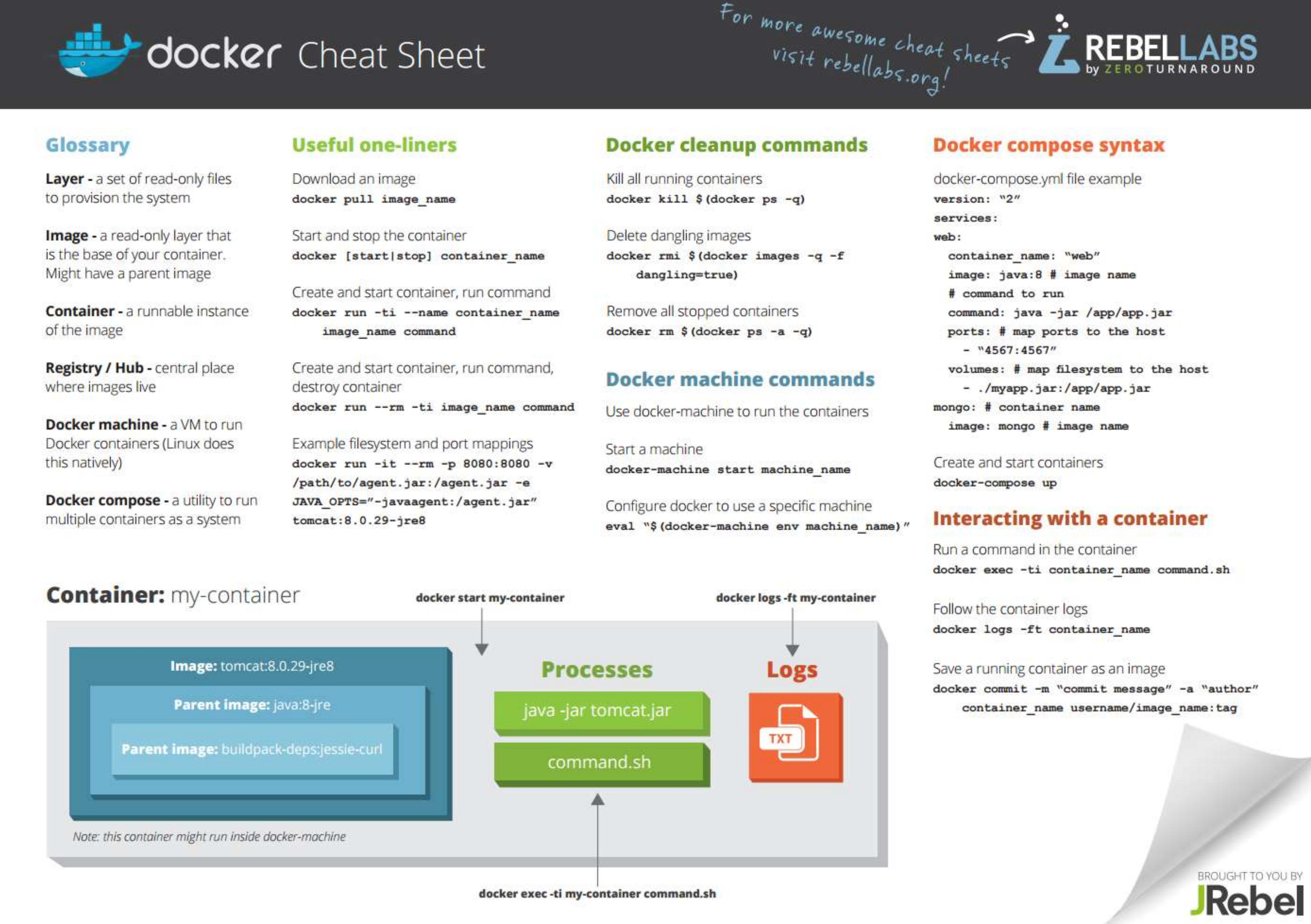
Docker Hub
Docker Cloud
Docker Store
Trusted Registry
Private Registry
Local Registry

Docker Engine

Docker terminology

- Build
  - Dockerfile, image
- Ship
  - Registry, repository, index
  - Docker ID
- Run
  - Engine
  - Container, machine
  - .yml, service, stack (service group), node – swarm

https://docs.docker.com/engine/docker-overview/
http://blog.thoward37.me/articles/where-are-docker-images-stored/



docker Cheat Sheet

For more awesome cheat sheets visit rebellabs.org!

REBELLABS by ZEROTURNAROUND

Glossary

Layer - a set of read-only files to provision the system

Image - a read-only layer that is the base of your container. Might have a parent image

Container - a runnable instance of the image

Registry / Hub - central place where images live

Docker machine - a VM to run Docker containers (Linux does this natively)

Docker compose - a utility to run multiple containers as a system

Useful one-liners

Download an image

```
docker pull image_name
```

Start and stop the container

```
docker [start|stop] container_name
```

Create and start container, run command

```
docker run -ti --name container_name
    image_name command
```

Create and start container, run command, destroy container

```
docker run --rm -ti image_name command
```

Example filesystem and port mappings

```
docker run -it --rm -p 8080:8080 -v
/path/to/agent.jar:/agent.jar -e
JAVA_OPTS="-javaagent:/agent.jar"
tomcat:8.0.29-jre8
```

Docker cleanup commands

Kill all running containers

```
docker kill $(docker ps -q)
```

Delete dangling images

```
docker rmi $(docker images -q -f
    dangling=true)
```

Remove all stopped containers

```
docker rm $(docker ps -a -q)
```

Docker machine commands

Use docker-machine to run the containers

Start a machine

```
docker-machine start machine_name
```

Configure docker to use a specific machine

```
eval "$(docker-machine env machine_name)"
```

Docker compose syntax

docker-compose.yml file example

```
version: "2"
services:
web:
  container_name: "web"
  image: java:8 # image name
  # command to run
  command: java -jar /app/app.jar
  ports: # map ports to the host
    - "4567:4567"
  volumes: # map filesystem to the host
    - ./myapp.jar:/app/app.jar
mongo: # container name
  image: mongo # image name
```

Create and start containers

```
docker-compose up
```

Interacting with a container

Run a command in the container

```
docker exec -ti container_name command.sh
```

Follow the container logs

```
docker logs -ft container_name
```

Save a running container as an image

```
docker commit -m "commit message" -a "author"
    container_name username/image_name:tag
```

Container: my-container

docker start my-container

docker logs -ft my-container

Image: tomcat:8.0.29-jre8

Parent image: java:8-jre

Parent image: buildpack-deps:jessie-curl

Processes

java -jar tomcat.jar

command.sh

Logs

TXT

Note: this container might run inside docker-machine

docker exec -ti my-container command.sh

BROUGHT TO YOU BY JRebel

http://files.zeroturnaround.com/pdf/zt_docker_cheat_sheet.pdf

Let’s play…

- Source
- Run Docker client/daemon
- Dockerfile and build
- compose.yml and deploy
- Docker ID and push
- Now pull and run from anywhere