Docker for the new era

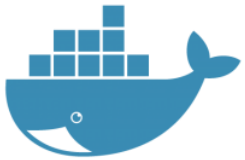
Ramit Surana

@ramitsurana

/in/ramitsurana

Agenda

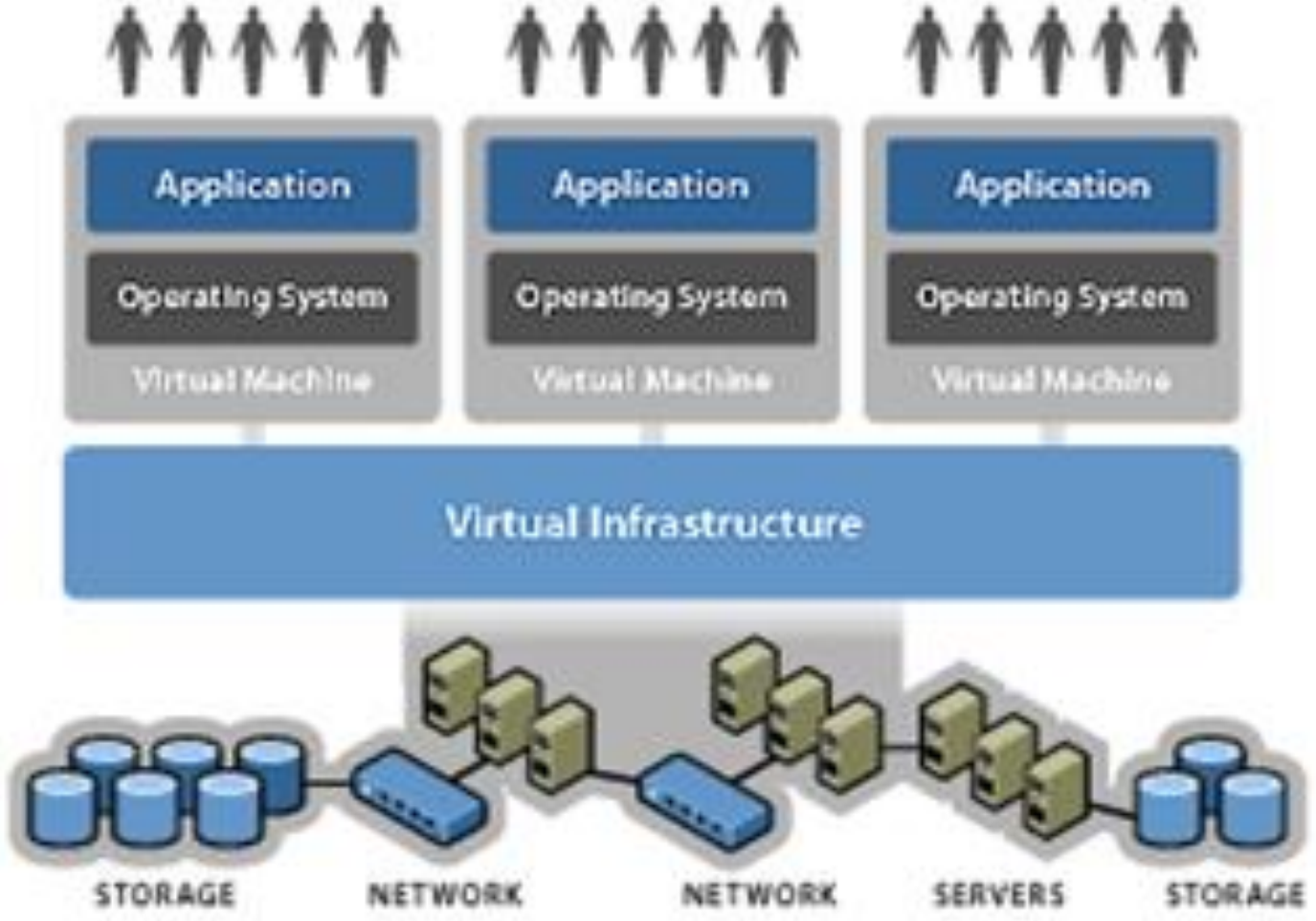
- About Docker
- Life Before and After Docker
- Other options than Docker
- Why Docker ?
- Docker Components
- Docker basics
- Docker Workflow
- Dockerfile
- Docker Hub
- Quay.io
- Docker Swarm
- Docker CI/CD
- Docker CI/CD workflow
- Dockerv1.1
- RunC
- ContainerD



About me

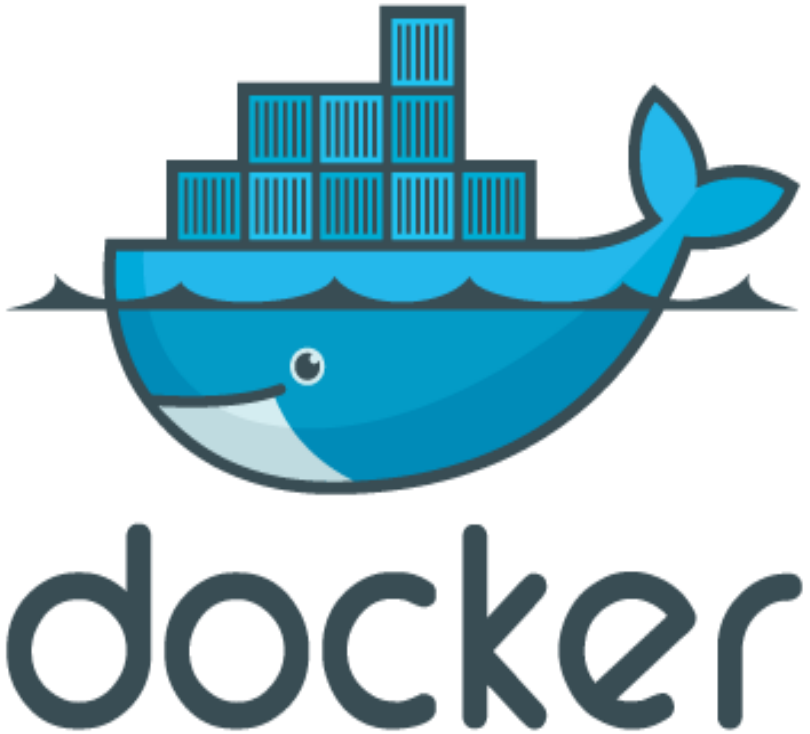
- Open source guy
- Doing cool stuff whenever possible
- Join me:
- Github: ramitsurana
- Linkedin: /in/ramitsurana
- Mail: ramitsurana@gmail.com

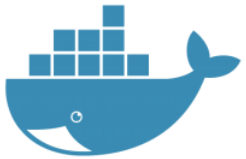
Life before Docker



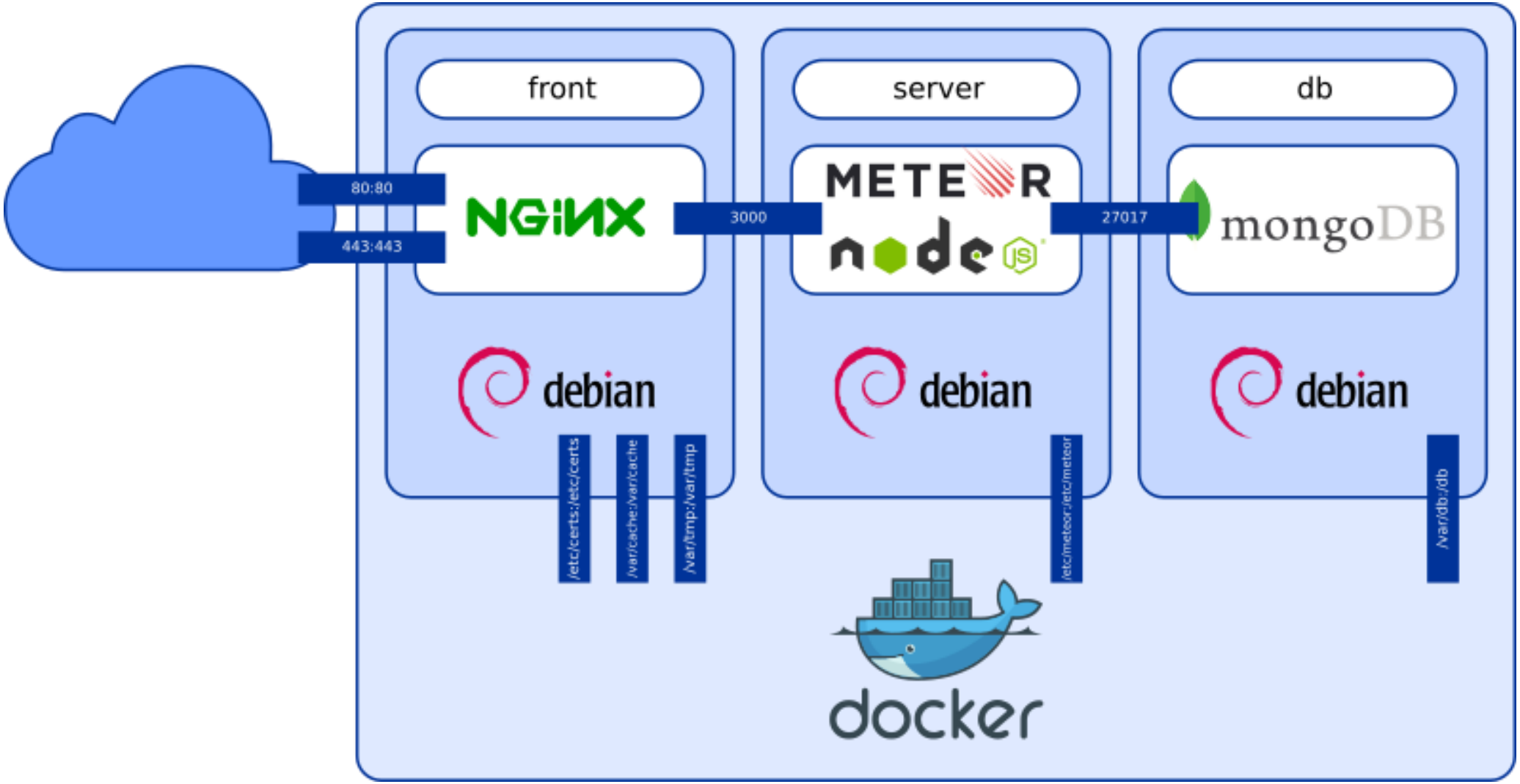
About Docker

- Docker is a modernized application container environment.
- Docker enables apps to be quickly assembled from components.
- It eliminates the friction between development, QA and production environments.



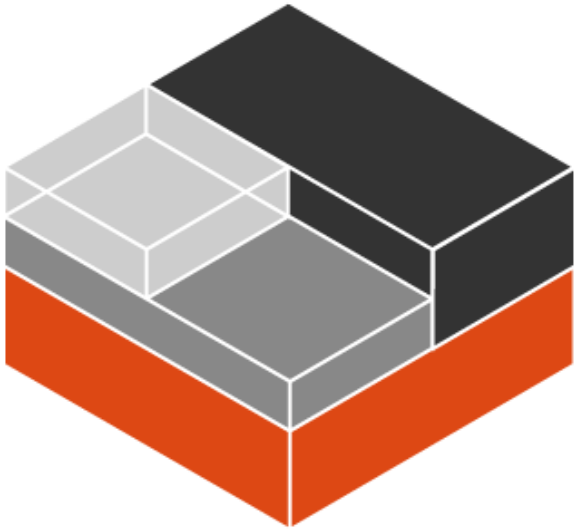


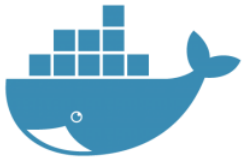
Life after Docker



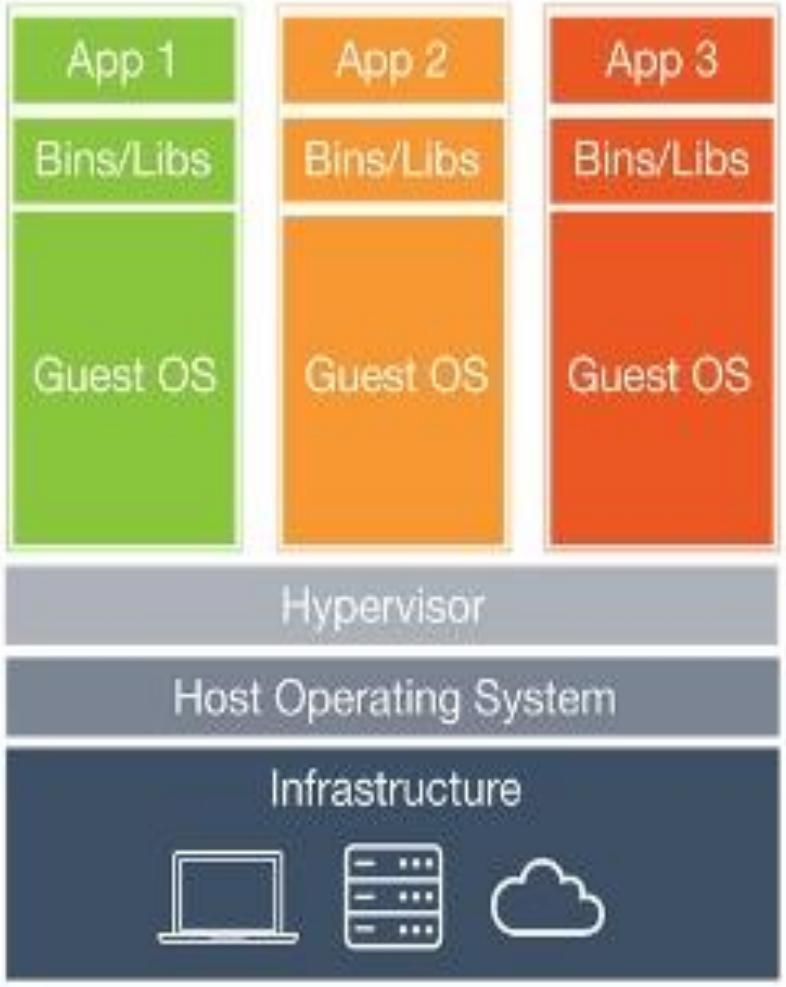
But are containers new in the market ?

- No
- Some other options available in the market are:
- LXC
- LXD
- OpenVZ

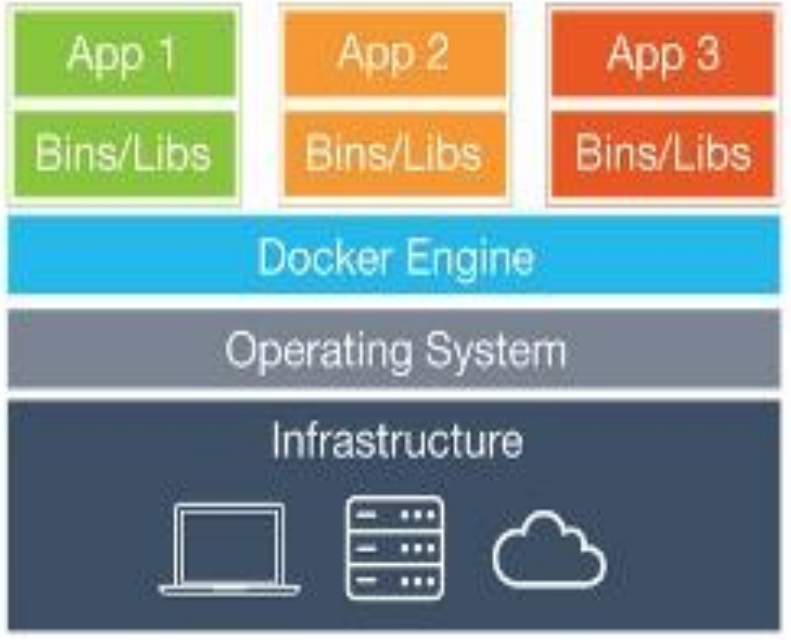




Containers vs VM's

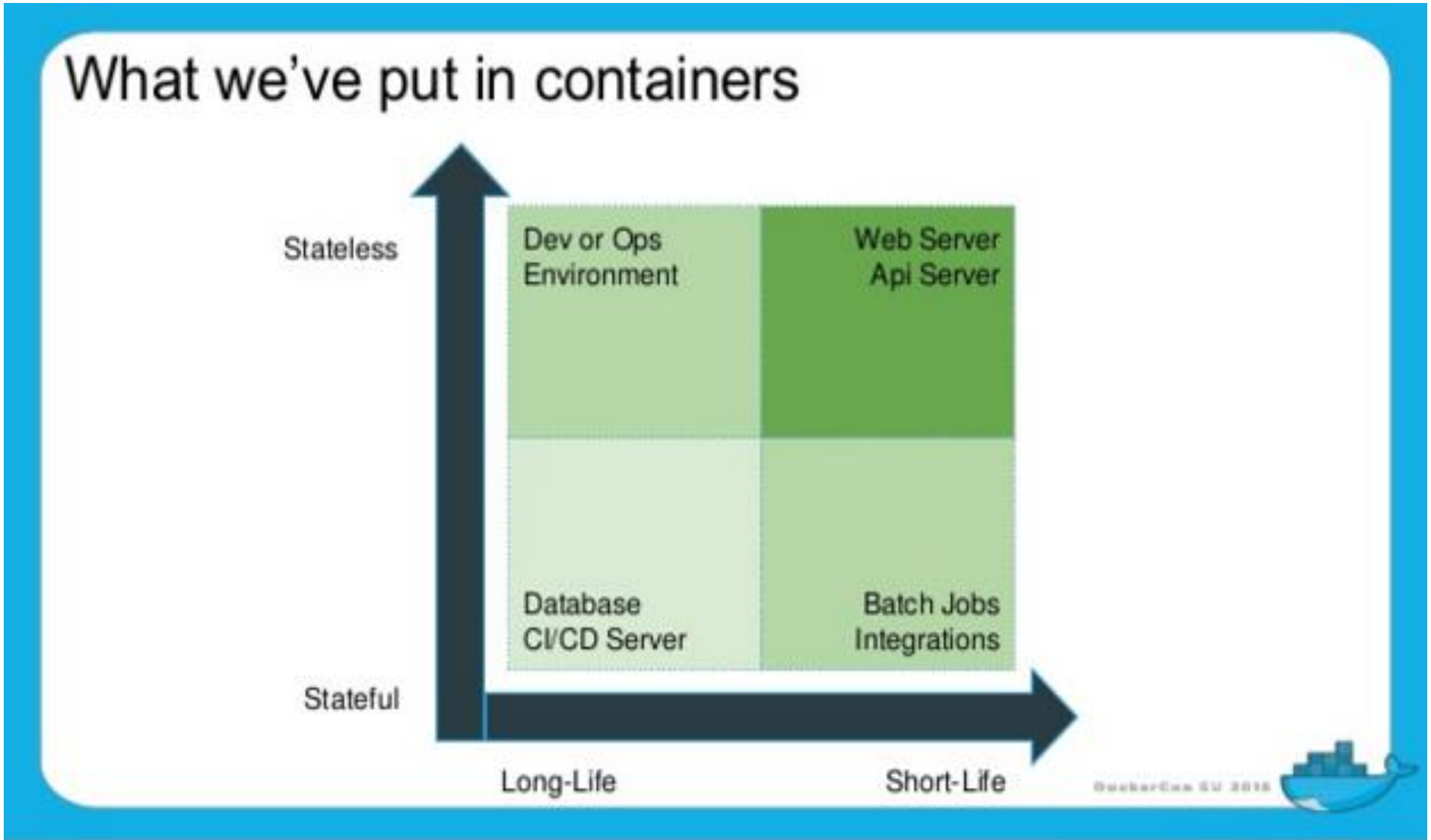


Virtual Machines

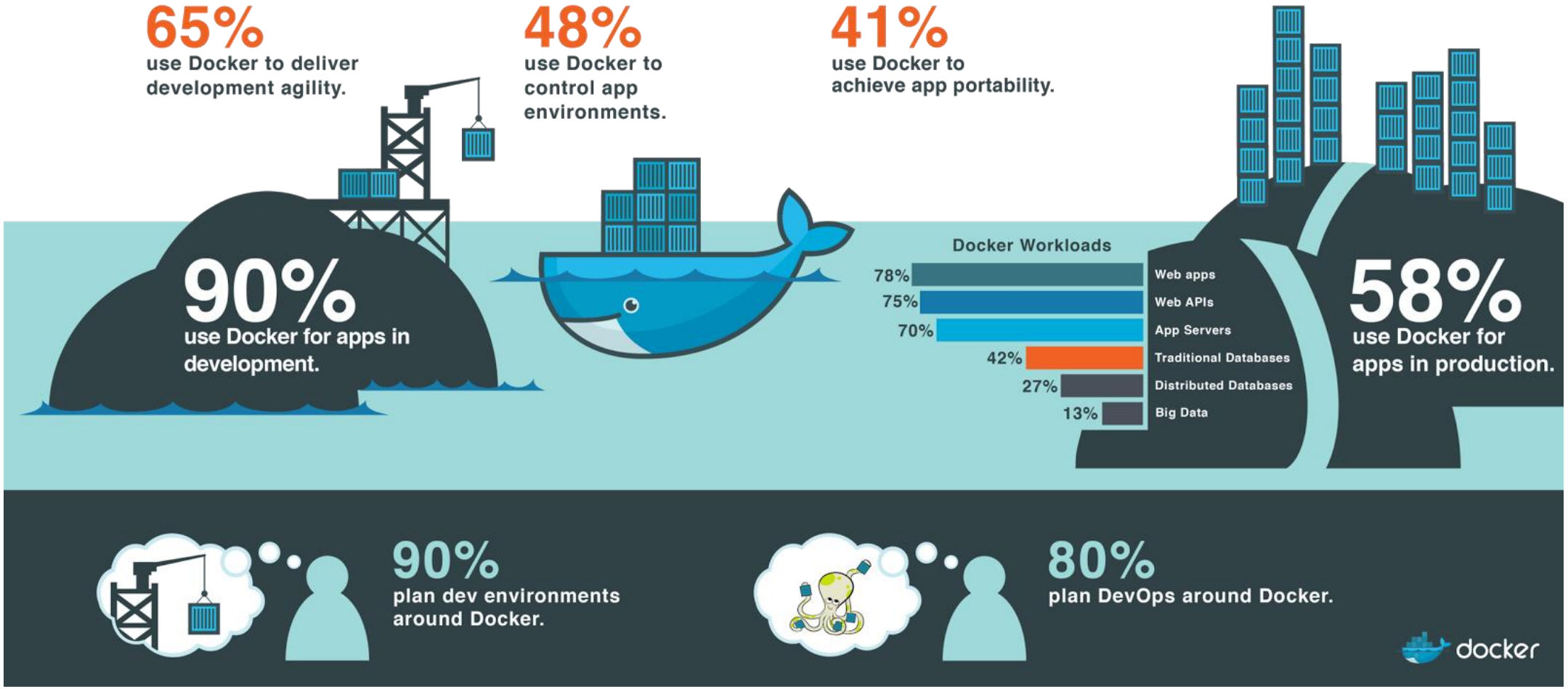


Containers

Stateless and Stateful Containers



Why Docker ?


65%
use Docker to deliver development agility.
48%
use Docker to control app environments.
41%
use Docker to achieve app portability.
90%
use Docker for apps in development.
Docker Workloads
78%
Web apps
75%
Web APIs
70%
App Servers
42%
Traditional Databases
27%
Distributed Databases
13%
Big Data
58%
use Docker for apps in production.
90%
plan dev environments around Docker.
80%
plan DevOps around Docker.
docker

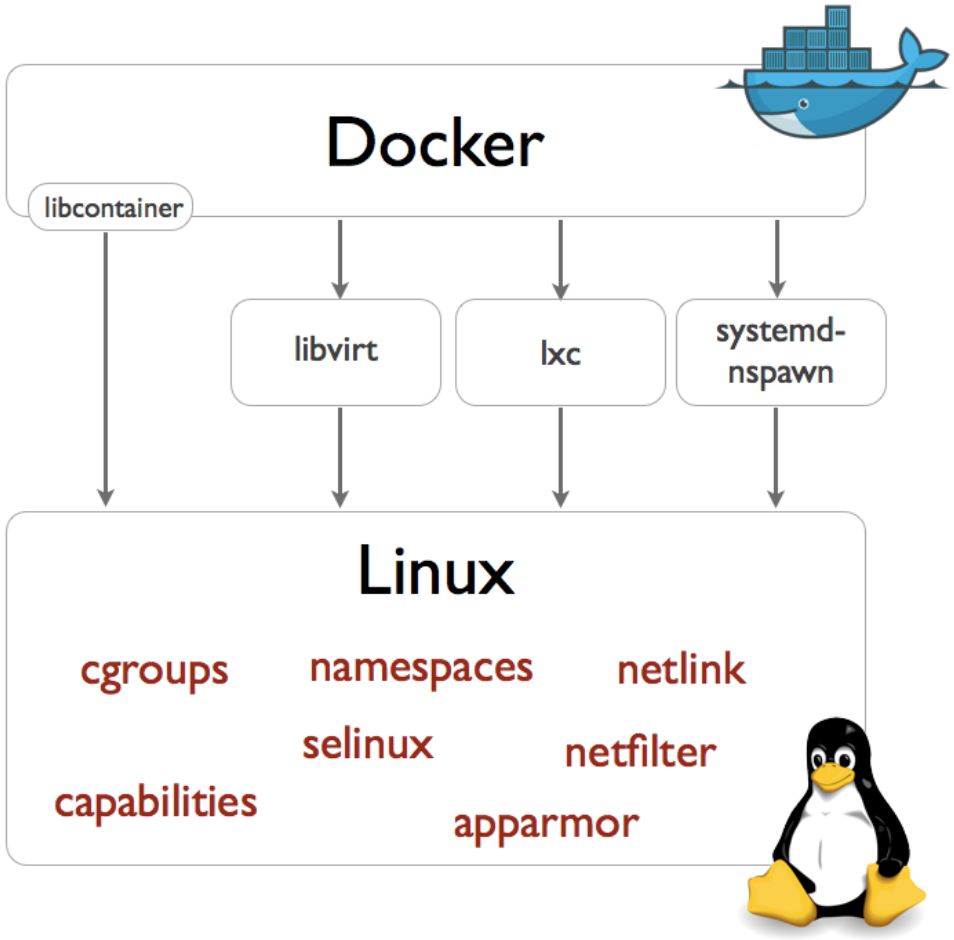
3 basic components of docker

machine

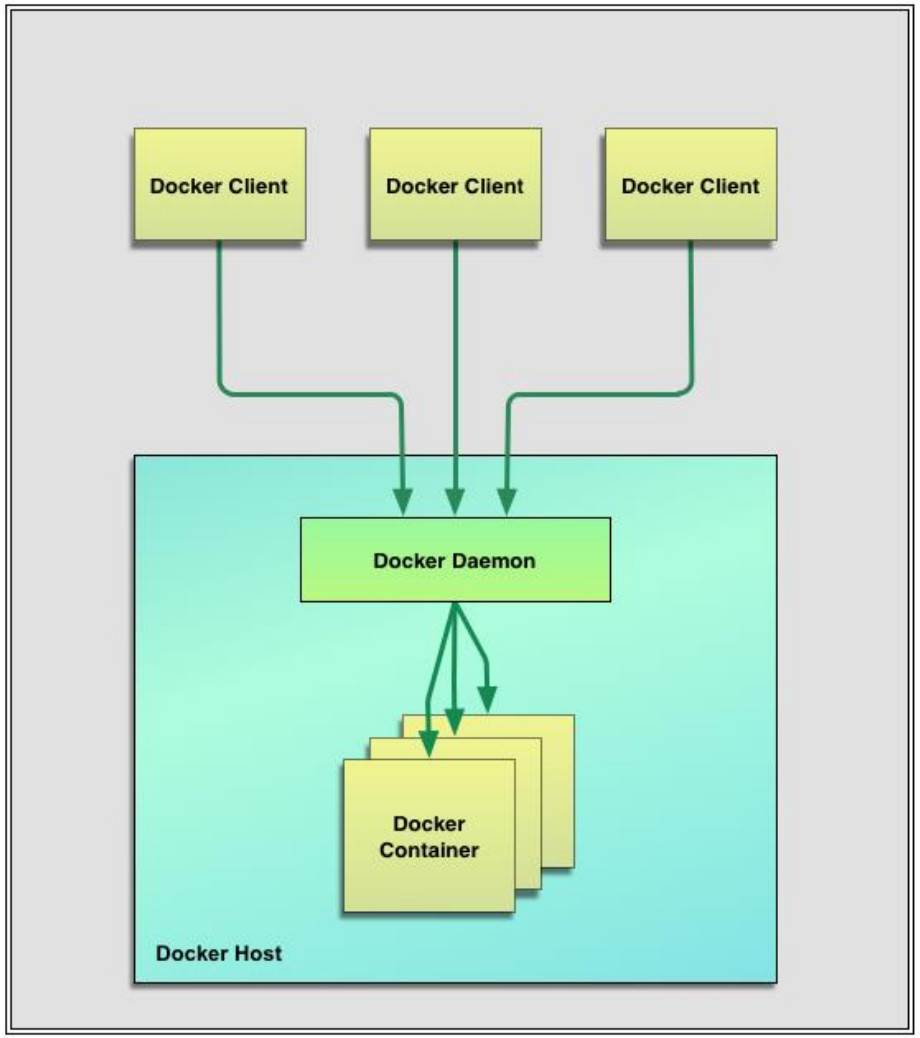
swarm

compose

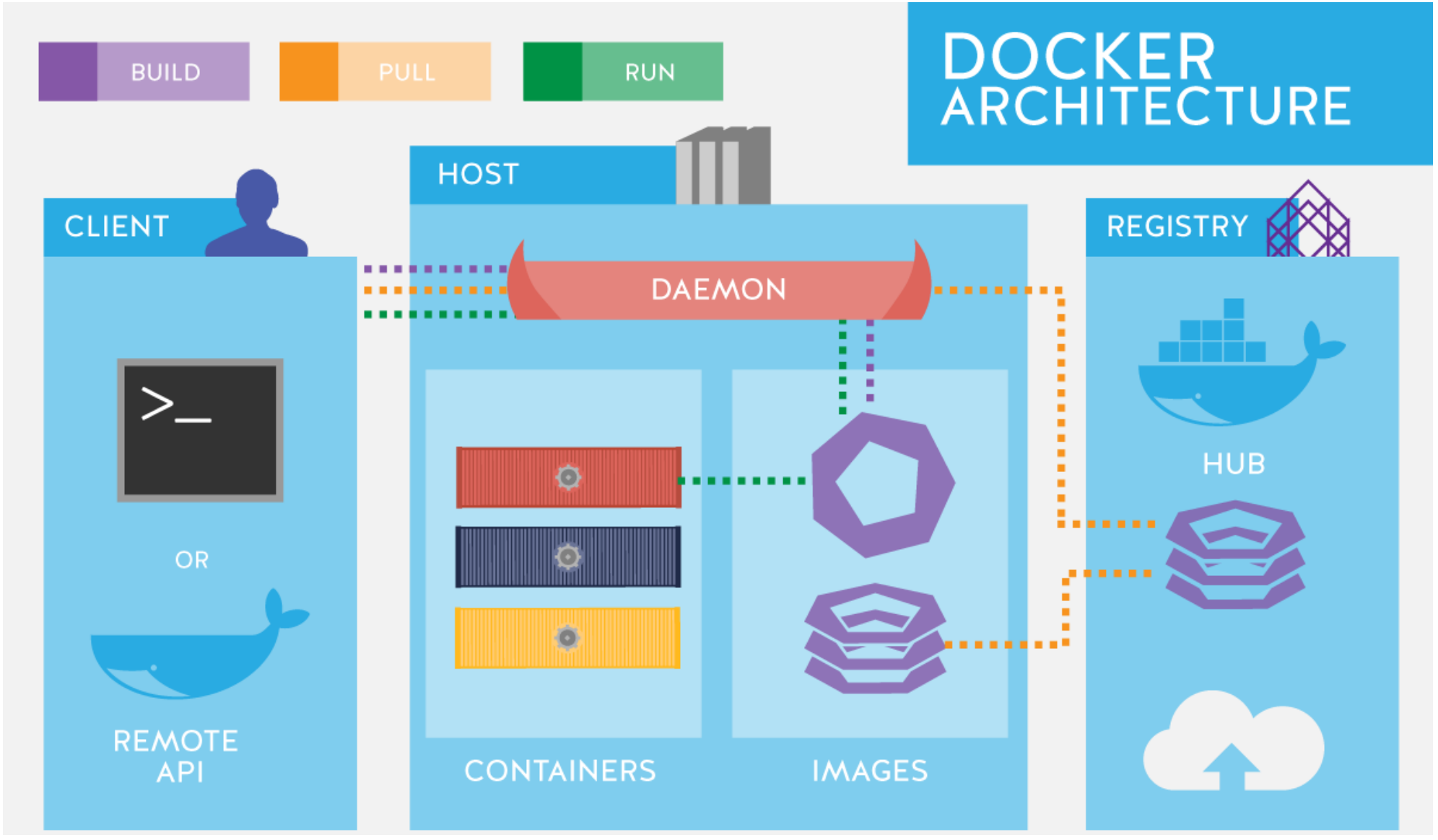
Docker basics



Basic Workflow



Better depiction of Workflow



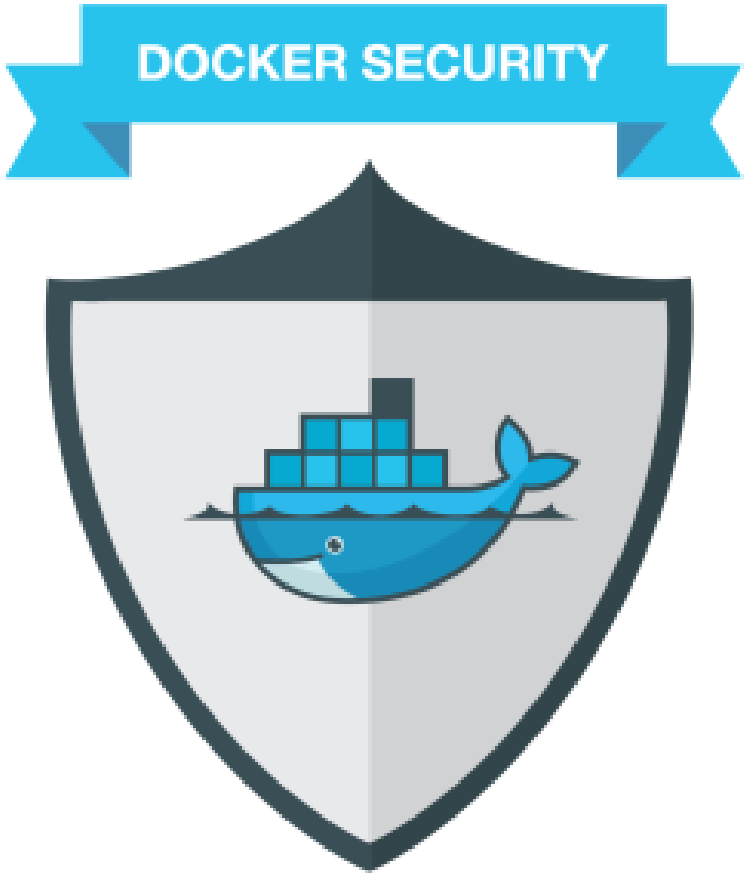
Docker Storage options

- Docker works on various storage options:
- Aufs
- Overlay
- ZFS
- VFS
- Device Mapper
- btrfs

When Adopting
Docker
Storage Matters!

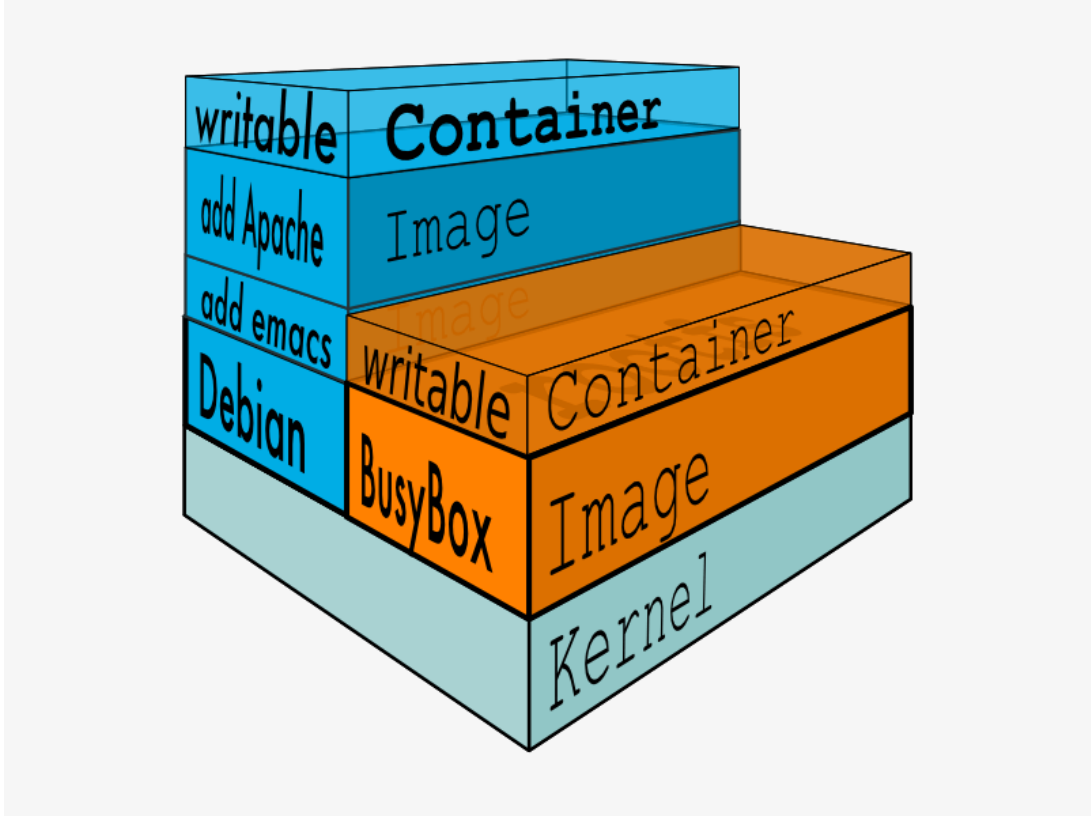
Docker Security

- Ideally Containers are not considered as secured as compared to virtualized environments.
- Some of the apps that can work for you in this domain are:
- Apparmor
- GRSecurity
- Seccomp Profiles



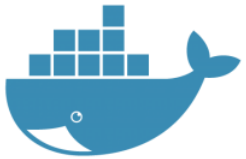
Docker Image

- Images are used to create Docker containers.
- Docker images are the build component of Docker.
- Docker provides a simple way to build new images or update existing images



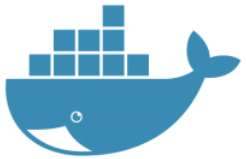
Docker commands

- sudo usermod -aG docker $(whoami) - After docker installation
- Docker info – To display information
- Docker images – To display images
- Docker run – To run images
- Docker pull – To pull the registries from Docket Hub
- Docker ps – To show the containers
- Service docker restart – To restart docker
- Service docker stop – To stop docker
- And many more….



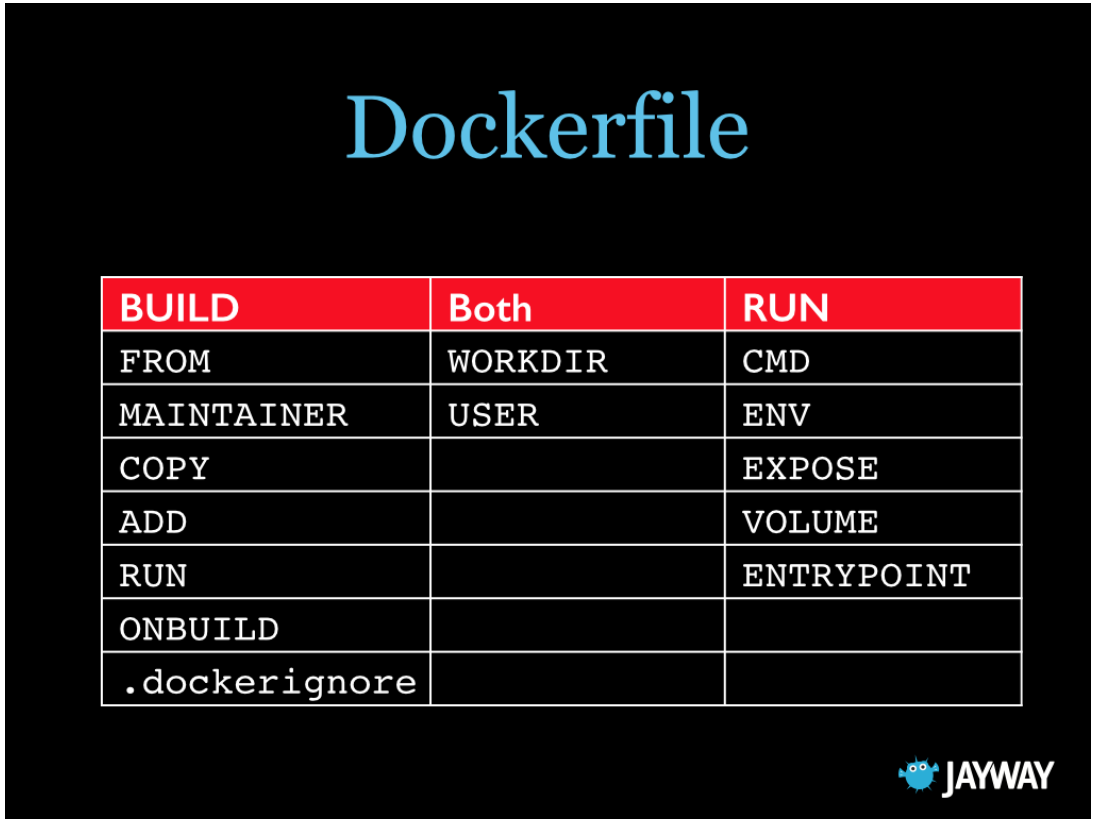
Docker flags

- --name – To name a container
- -d – To run the container in daemonized mode
- -f – To do forcefully
- -a – To attach
- -e – To set environment variables
- -m – Memory limit
- -p – To publish new ports
- And many more....



Dockerfile

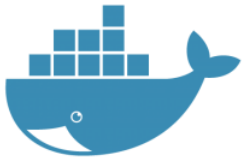
- Script i.e. composed of various commands and arguments listed successively to automatically perform actions on a base image in order to create a new one.
- Checkout some of my work:
- https://hub.docker.com/u/ramitsurana
- https://quay.io/user/ramitsurana05



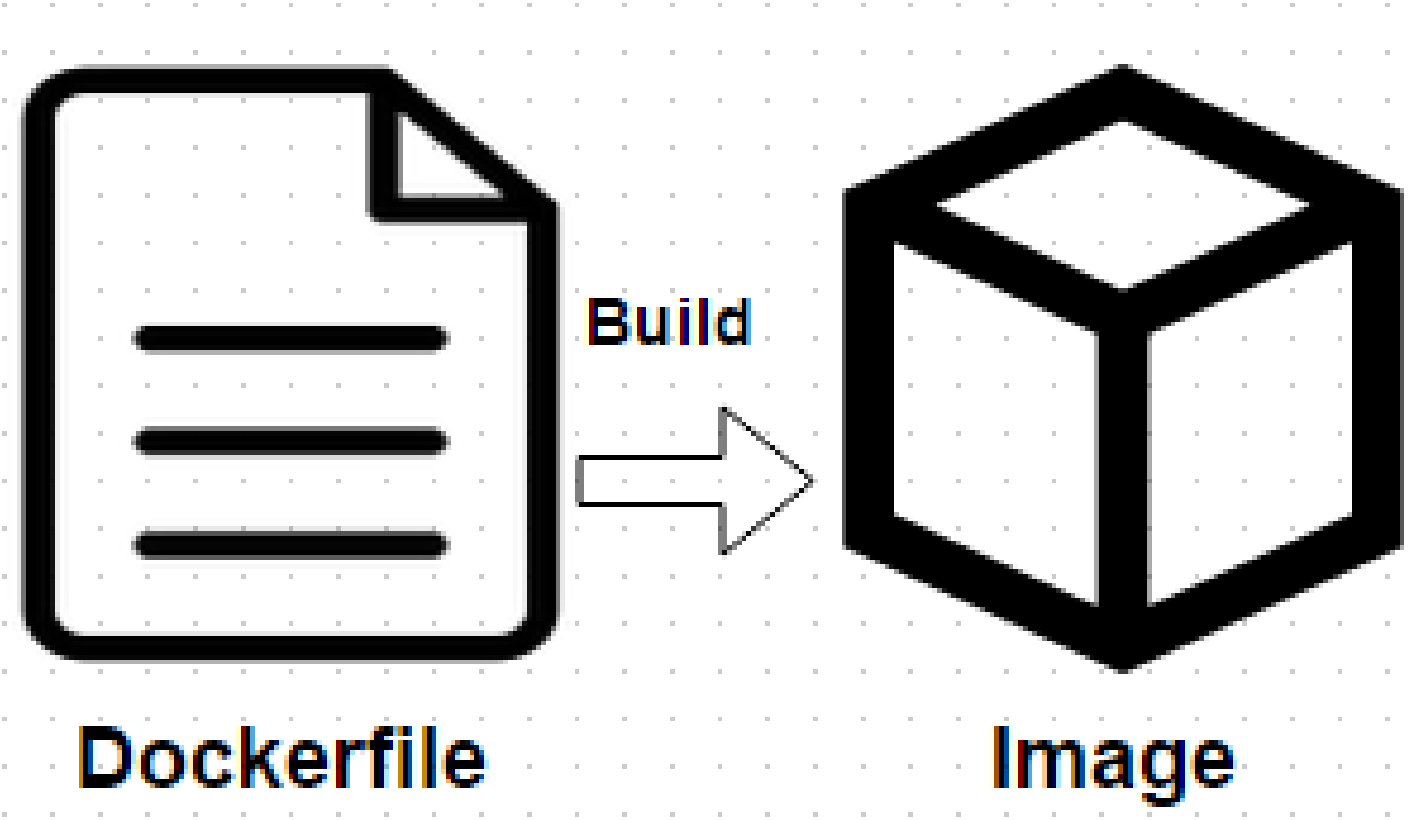
Dockerfile

- Sample of my work on Heroku-runtime:

```
FROM ubuntu

MAINTAINER Ramit Surana <ramitsurana@gmail.com>

#Installing basics
RUN apt-get update -qqy
RUN apt-get install -y ca-certificates curl wget git

#Installing Heroku Toolbelt
RUN echo >/etc/apt/sources.list.d/heroku.list \
deb http://toolbelt.heroku.com/ubuntu ./
RUN curl -sL https://toolbelt.heroku.com/apt/release.key | apt-key add -
RUN apt-get update && apt-get install -y heroku-toolbelt

#Cloning repository
RUN git clone https://github.com/ramitsurana/heroku-runtime

#Setting Workdir
WORKDIR ["/usr/local/heroku"]

~
```

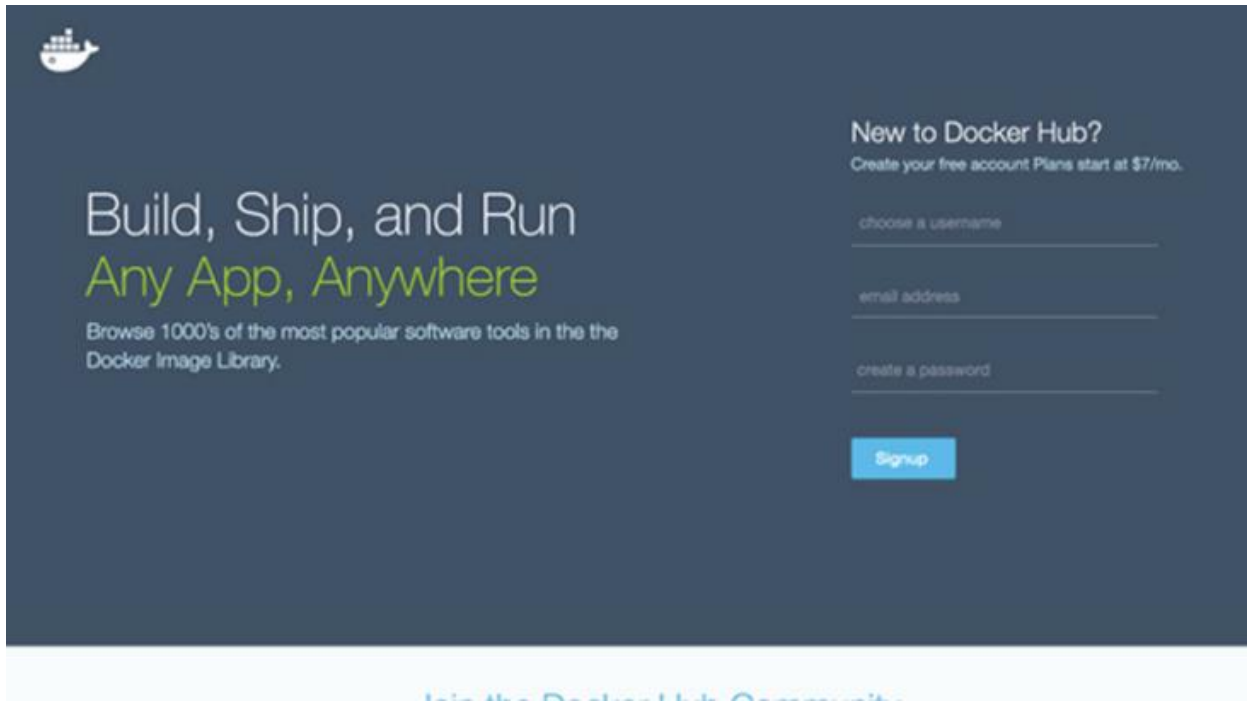


Dockerfile vs Docker Image



Docker Hub

- Collection of variety of different images from organizations & users.
- Now supports Vulnerability analysis
- Consists of 2 types of repositories:
- Official repos
- User based repos



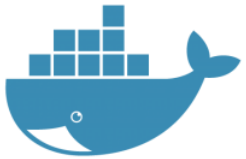
How to use it with docker ?

- Docker login on cli:

```
ramit@ramit-surana:~$ sudo docker login
Username (ramitsurana): ramitsurana
Password:
WARNING: login credentials saved in /home/ramit/.docker/config.json
Login Succeeded
```

- Pulling images from docker hub:

```
ramit@ramit-surana:~$ sudo docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world

4276590986f6: Already exists
a3ed95caeb02: Already exists
Digest: sha256:4f32210e234b4ad5cac92efacc0a3d602b02476c754f13d517e1ada048e5a8ba
Status: Downloaded newer image for hello-world:latest
```



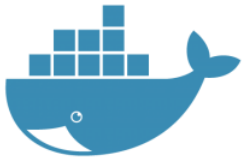
How to use it with docker ?

- Checking the image's presence :

```
ramit@ramit-surana:~$ sudo docker ps -a
CONTAINER ID        IMAGE                      COMMAND                  CREATED             STATUS                          PORTS
NAMES
cbb64a6d2170        quay.io/coreos/quay-docs   "/usr/local/bin/start"   About an hour ago   Exited (137) 59 minutes ago
silly_bartik
17aa534ee92f        quay.io/coreos/etcd        "/etcd"                  2 hours ago         Exited (0) About an hour ago
small_jones
b486799a9e25        hello-world                "/hello"                 2 weeks ago         Exited (0) 2 weeks ago
focused_dubinsky
```

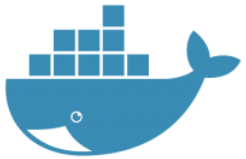
- Running the container :

```
ramit@ramit-surana:~$ sudo docker run -it hello-world
```



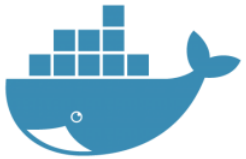
Let's Check out the output:

```
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.
 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 Hub account:
 https://hub.docker.com

For more examples and ideas, visit:
 https://docs.docker.com/engine/userguide/
```



Quay.io

- Hosted by CoreOS guys
- Supports rkt project
- Integrated with the amazing clair vulnerability service.
- Integrates with tectonic and other coreos projects



How to use it with docker ?

- Quay login with docker:

```
ramit@ramit-surana:~$ sudo docker login quay.io
Username (ramitsurana05): ramitsurana05
Password:
WARNING: login credentials saved in /home/ramit/.docker/config.json
Login Succeeded
ramit@ramit-surana:~$
```

- Pulling images from quay.io:

```
ramit@ramit-surana:~$ sudo docker pull quay.io/coreos/quay-docs
Using default tag: latest
latest: Pulling from coreos/quay-docs

a3ed95caeb02: Pull complete
b89e7f03cb68: Pull complete
035270191e61: Pull complete
070c9bbdef95: Downloading [==========================================>        ] 43.41 MB/49.55 MB
070c9bbdef95: Pull complete
801ae97e2880: Pull complete
e0d2be0d8d8d: Pull complete
05d57112a6b8: Pull complete
79fa7af75443: Pull complete
Digest: sha256:34e56b09dff122bcb97891e0731b394dc8a8948cfa0c9c7b4c648082601fe423
Status: Downloaded newer image for quay.io/coreos/quay-docs:latest
```

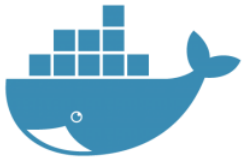
How to use it with docker ?

- Checking the image's presence :

```
ramit@ramit-surana:~$ sudo docker images
REPOSITORY                TAG                 IMAGE ID            CREATED             SIZE
quay.io/coreos/etcd       latest              019db9d72ee2        43 hours ago        32.29 MB
hello-world               latest              94df4f0ce8a4        2 weeks ago         967 B
quay.io/coreos/quay-docs  latest              515321ada123        7 weeks ago         198.9 MB
```

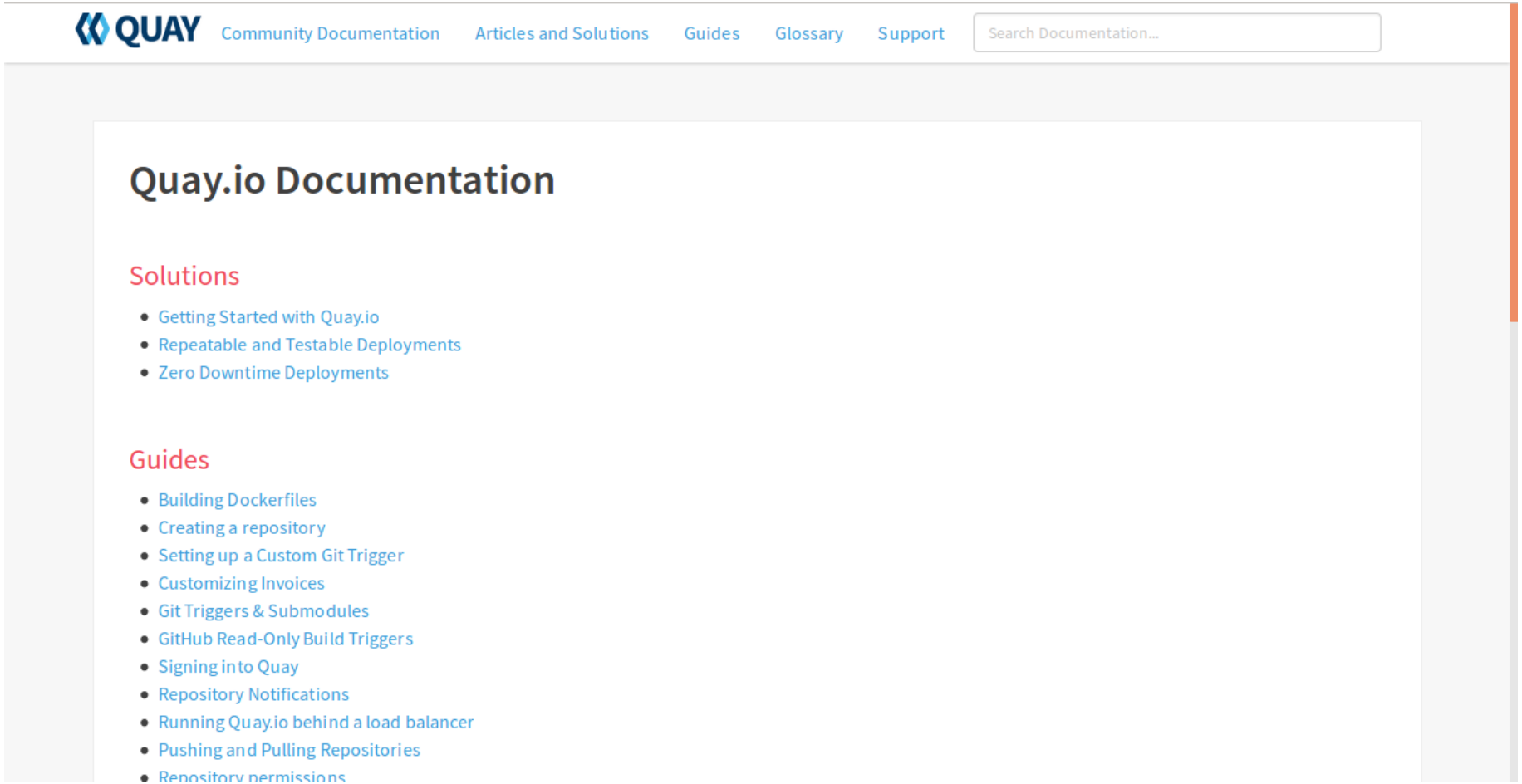
- Running the container :

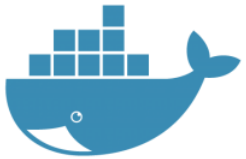
```
ramit@ramit-surana:~$ sudo docker run -p 4000:4000 -e QUAY_HOST=https://my.registry quay.io/coreos/quay-docs
Configuration file: /srv/jekyll/_config.yml
            Source: /srv/jekyll
       Destination: /srv/jekyll/_site
 Incremental build: disabled. Enable with --incremental
      Generating...
                    done in 1.909 seconds.
 Auto-regeneration: enabled for '/srv/jekyll'
Configuration file: /srv/jekyll/_config.yml
    Server address: http://0.0.0.0:4000/
  Server running... press ctrl-c to stop.
```



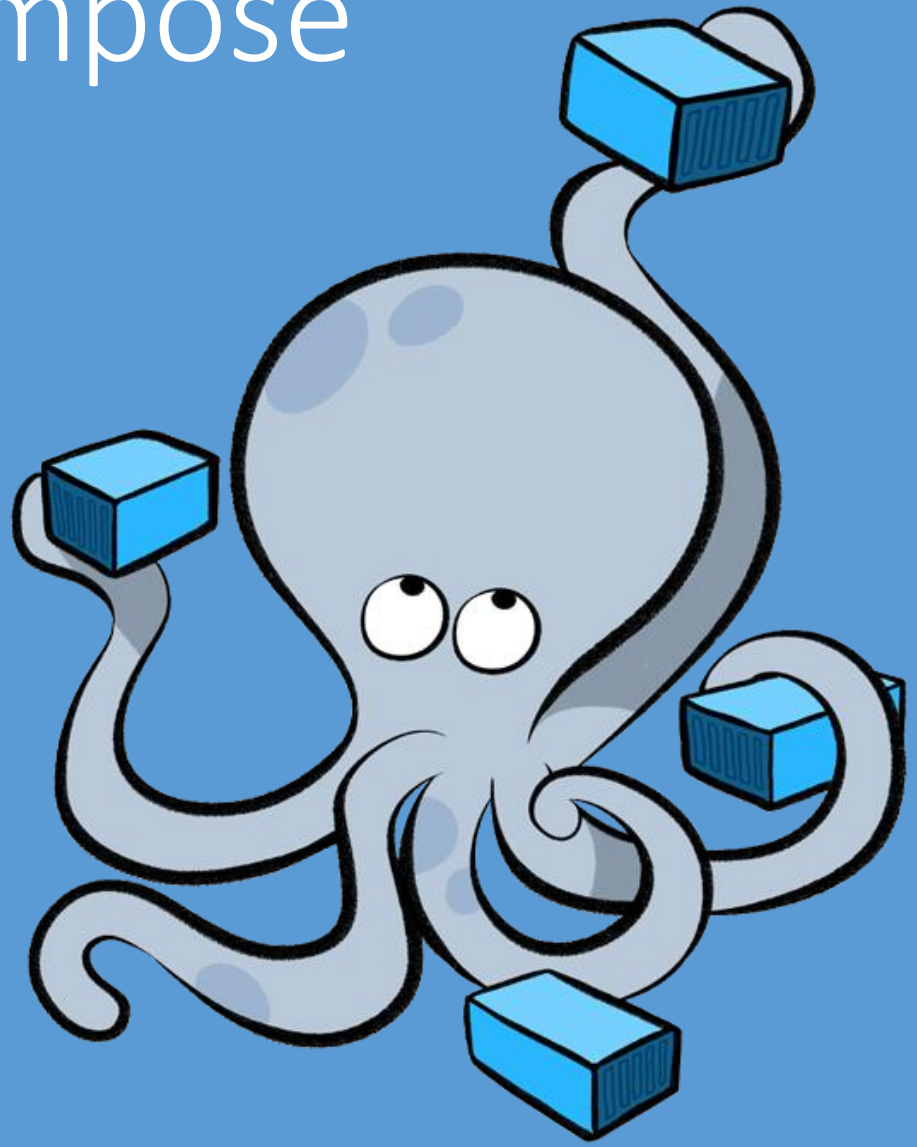
How to use it with docker ?

- Check out http://localhost:4000. Bingo !!



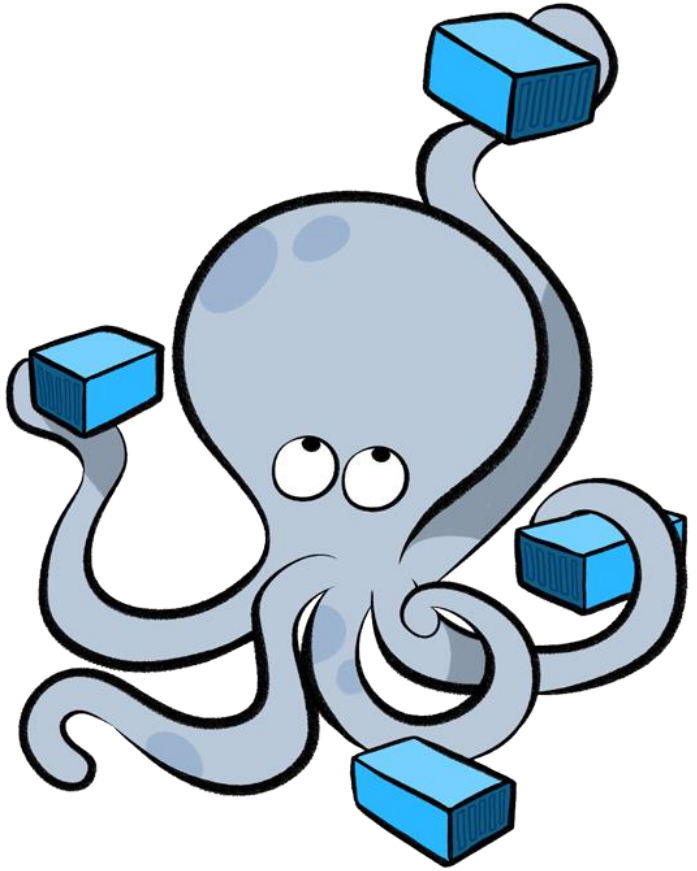


Docker Compose



Docker Compose

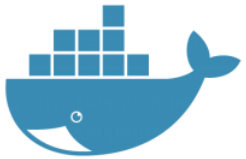
- Simple way to bring up multiple containers with .yaml file.
- Previously known as Fig.
- Bought by docker in 2014.



How to use Docker Compose ?

- Installation:

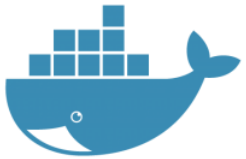
```
ramit@ramit-ramitsurana:~$ sudo apt-get -y install python-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  gir1.2-rb-3.0 gir1.2-secret-1 libdmapsharing-3.0-2 libgmime-2.6-0
  libgrilo-0.2-1 libquvi-scripts libquvi7 librhythmbox-core9
  libtotem-plparser-common libtotem-plparser18 media-player-info python3-mako
  rhythmbox-data
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libexpat1-dev libpython-all-dev libpython-dev libpython2.7-dev python-all
  python-all-dev python-dev python-pip-whl python-setuptools python-wheel
  python2.7-dev
Suggested packages:
  python-setuptools-doc
The following NEW packages will be installed:
  libexpat1-dev libpython-all-dev libpython-dev libpython2.7-dev python-all
  python-all-dev python-dev python-pip python-pip-whl python-setuptools
  python-wheel python2.7-dev
0 upgraded, 12 newly installed, 0 to remove and 55 not upgraded.
Need to get 29.5 MB/29.6 MB of archives.
After this operation, 44.6 MB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 libpython2.7-dev amd64 2.7.11-7ubuntu1 [27.8 MB]
Get:2 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 libpython-dev amd64 2.7.11-1 [7,728 B]
Get:3 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 libpython-all-dev amd64 2.7.11-1 [992 B]
Get:4 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 python-all amd64 2.7.11-1 [978 B]
Get:5 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 python2.7-dev amd64 2.7.11-7ubuntu1 [280 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 python-dev amd64 2.7.11-1 [1,160 B]
Get:7 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 python-all-dev amd64 2.7.11-1 [1,000 B]
Get:8 http://in.archive.ubuntu.com/ubuntu xenial/universe amd64 python-pip-whl all 8.1.1-2 [1,074 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu xenial/universe amd64 python-pip all 8.1.1-2 [144 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 python-setuptools all 20.7.0-1 [169 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu xenial/universe amd64 python-wheel all 0.29.0-1 [48.0 kB]
Fetched 22.9 MB in 1min 29s (256 kB/s)
Selecting previously unselected package libexpat1-dev:amd64.
(Reading database ... 219618 files and directories currently installed.)
Preparing to unpack .../libexpat1-dev_2.1.0-7ubuntu0.16.04.1_amd64.deb ...
Unpacking libexpat1-dev:amd64 (2.1.0-7ubuntu0.16.04.1) ...
Selecting previously unselected package libpython2.7-dev:amd64.
Preparing to unpack .../libpython2.7-dev_2.7.11-7ubuntu1_amd64.deb ...
Unpacking libpython2.7-dev:amd64 (2.7.11-7ubuntu1) ...
Selecting previously unselected package libpython-dev:amd64.
```



How to use Docker Compose ?

- Installation:

```
ramit@ramit-ramitsurana:~$ pip install docker-compose
Collecting docker-compose
  Downloading docker-compose-1.7.1.tar.gz (141kB)
    100% |████████████████████████████████| 143kB 270kB/s
Collecting cached-property<2,>=1.2.0 (from docker-compose)
  Downloading cached_property-1.3.0-py2.py3-none-any.whl
Collecting docopt<0.7,>=0.6.1 (from docker-compose)
  Downloading docopt-0.6.2.tar.gz
Collecting PyYAML<4,>=3.10 (from docker-compose)
  Downloading PyYAML-3.11.zip (371kB)
    100% |████████████████████████████████| 378kB 257kB/s
Collecting requests<2.8,>=2.6.1 (from docker-compose)
  Downloading requests-2.7.0-py2.py3-none-any.whl (470kB)
    100% |████████████████████████████████| 471kB 250kB/s
Collecting texttable<0.9,>=0.8.1 (from docker-compose)
  Downloading texttable-0.8.4.tar.gz
Collecting websocket-client<1.0,>=0.32.0 (from docker-compose)
  Downloading websocket_client-0.37.0.tar.gz (194kB)
    100% |████████████████████████████████| 194kB 263kB/s
Collecting docker-py<2,>=1.8.1 (from docker-compose)
  Downloading docker_py-1.8.1-py2.py3-none-any.whl (41kB)
    100% |████████████████████████████████| 51kB 336kB/s
Collecting dockerpty<0.5,>=0.4.1 (from docker-compose)
  Downloading dockerpty-0.4.1.tar.gz
Collecting six<2,>=1.3.0 (from docker-compose)
  Downloading six-1.10.0-py2.py3-none-any.whl
Collecting jsonschema<3,>=2.5.1 (from docker-compose)
  Downloading jsonschema-2.5.1-py2.py3-none-any.whl
Collecting enum34<2,>=1.0.4 (from docker-compose)
  Downloading enum34-1.1.6-py2-none-any.whl
Collecting ipaddress>=1.0.16; python_version < "3.3" (from docker-py<2,>=1.8.1->docker-compose)
  Downloading ipaddress-1.0.16-py27-none-any.whl
Collecting backports.ssl-match-hostname>=3.5; python_version < "3.5" (from docker-py<2,>=1.8.1->docker-compose)
  Downloading backports.ssl_match_hostname-3.5.0.1.tar.gz
Collecting functools32; python_version == "2.7" (from jsonschema<3,>=2.5.1->docker-compose)
  Downloading functools32-3.2.3-2.zip
Building wheels for collected packages: docker-compose, docopt, PyYAML, texttable, websocket-client, dockerpty, backports.ssl-match-hostname, f
unctools32
  Running setup.py bdist_wheel for docker-compose ... done
  Stored in directory: /home/ramit/.cache/pip/wheels/f6/89/0d/67b12aa5eac653598b5e24a9407b11043c0c53b80860bcf883
  Running setup.py bdist_wheel for docopt ... done
  Stored in directory: /home/ramit/.cache/pip/wheels/b2/16/5f/c33a2bb5f2dce71205f8e65cbfd05647d79d441282be31fd82
  Running setup.py bdist_wheel for PyYAML ... done
```



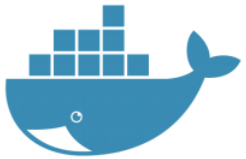
How to use it with Docker Compose ?

- Create a dir and .yml file inside it:

```
ramit@ramit-ramitsurana:~$ mkdir hello
ramit@ramit-ramitsurana:~$ cd hello/

ramit@ramit-ramitsurana:~/hello$ vim docker-compose.yml
```

- My .yml file:

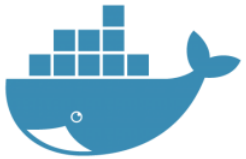
```
my-test:
  image: hello-world
~
~
~
~
~
~
~
```



How to use it with Docker Compose ?

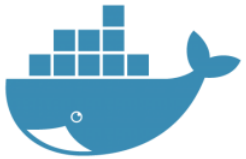
- The Output:

```
ramit@ramit-ramitsurana:~/hello$ sudo docker-compose up
Creating hello_my-test_1
Attaching to hello_my-test_1
my-test_1 |
my-test_1 | Hello from Docker.
my-test_1 | This message shows that your installation appears to be working correctly.
my-test_1 |
my-test_1 | To generate this message, Docker took the following steps:
my-test_1 |  1. The Docker client contacted the Docker daemon.
my-test_1 |  2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
my-test_1 |  3. The Docker daemon created a new container from that image which runs the
my-test_1 |     executable that produces the output you are currently reading.
my-test_1 |  4. The Docker daemon streamed that output to the Docker client, which sent it
my-test_1 |     to your terminal.
my-test_1 |
my-test_1 | To try something more ambitious, you can run an Ubuntu container with:
my-test_1 |  $ docker run -it ubuntu bash
my-test_1 |
my-test_1 | Share images, automate workflows, and more with a free Docker Hub account:
my-test_1 |  https://hub.docker.com
my-test_1 |
my-test_1 | For more examples and ideas, visit:
my-test_1 |  https://docs.docker.com/engine/userguide/
my-test_1 |
hello_my-test_1 exited with code 0
```

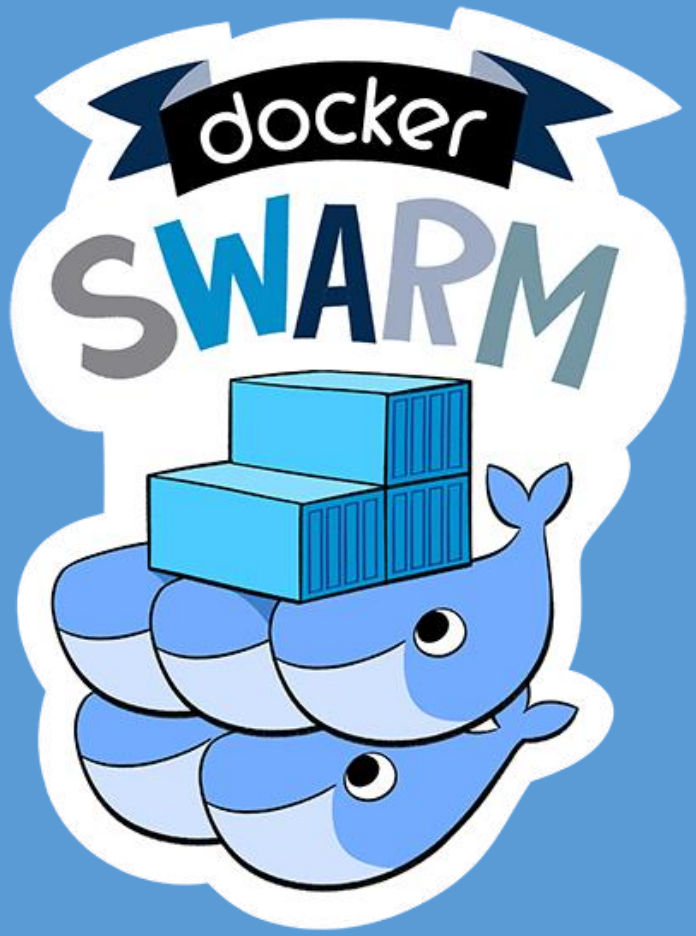


How to use it with Docker Compose ?

- Find more cool examples at http://github.com/ramitsurana/dcompose
- Some simple commands for docker compose:
- docker-compose up – To bring up containers
- docker-compose build - Build or rebuild services
- docker-compose kill - Kill containers
- docker-compose logs - View output from containers
- docker-compose port - Print the public port for a port binding
- docker-compose ps - List containers
- docker-compose pull - Pulls service images
- docker-compose rm - Remove stopped containers
- docker-compose scale - Set number of containers for a service

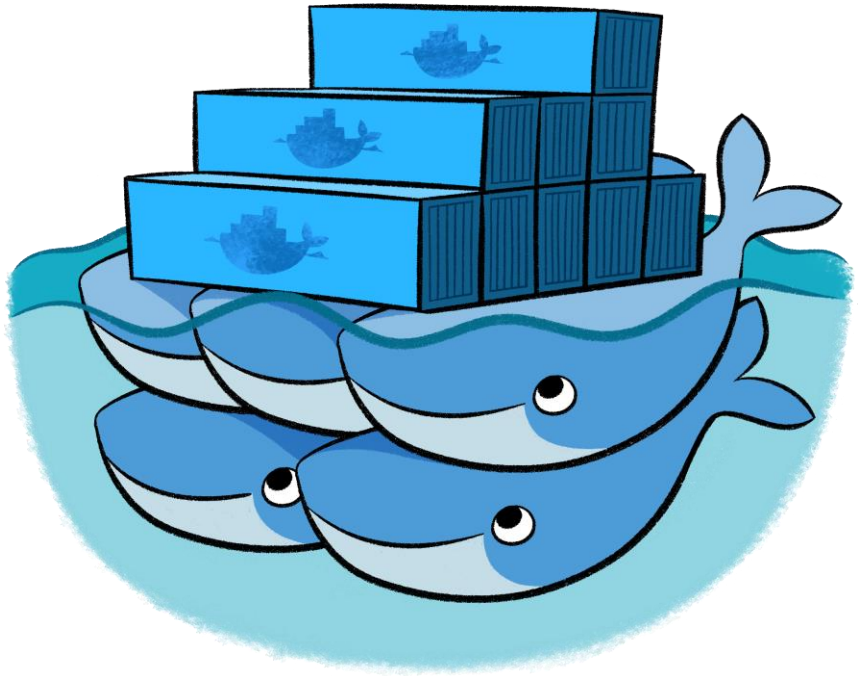


Docker Swarm

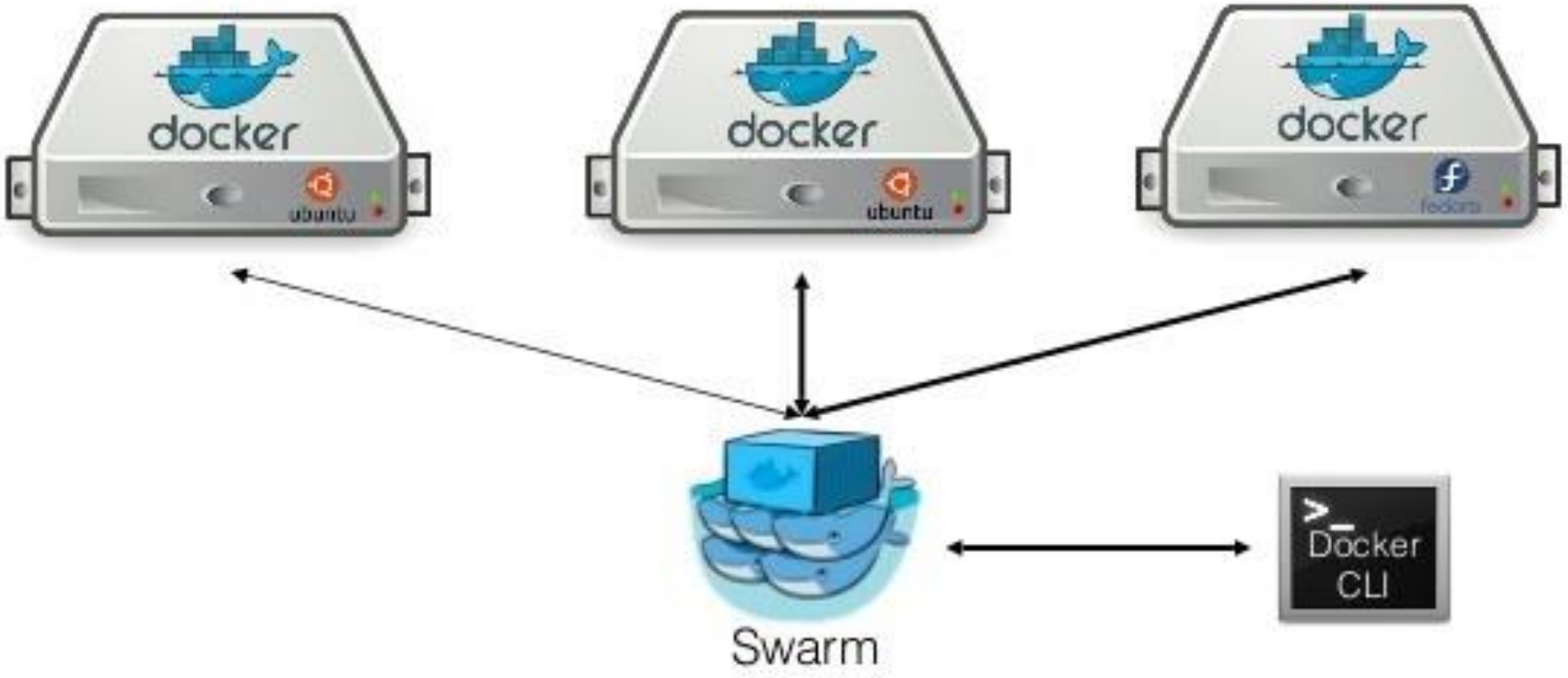


Docker Swarm

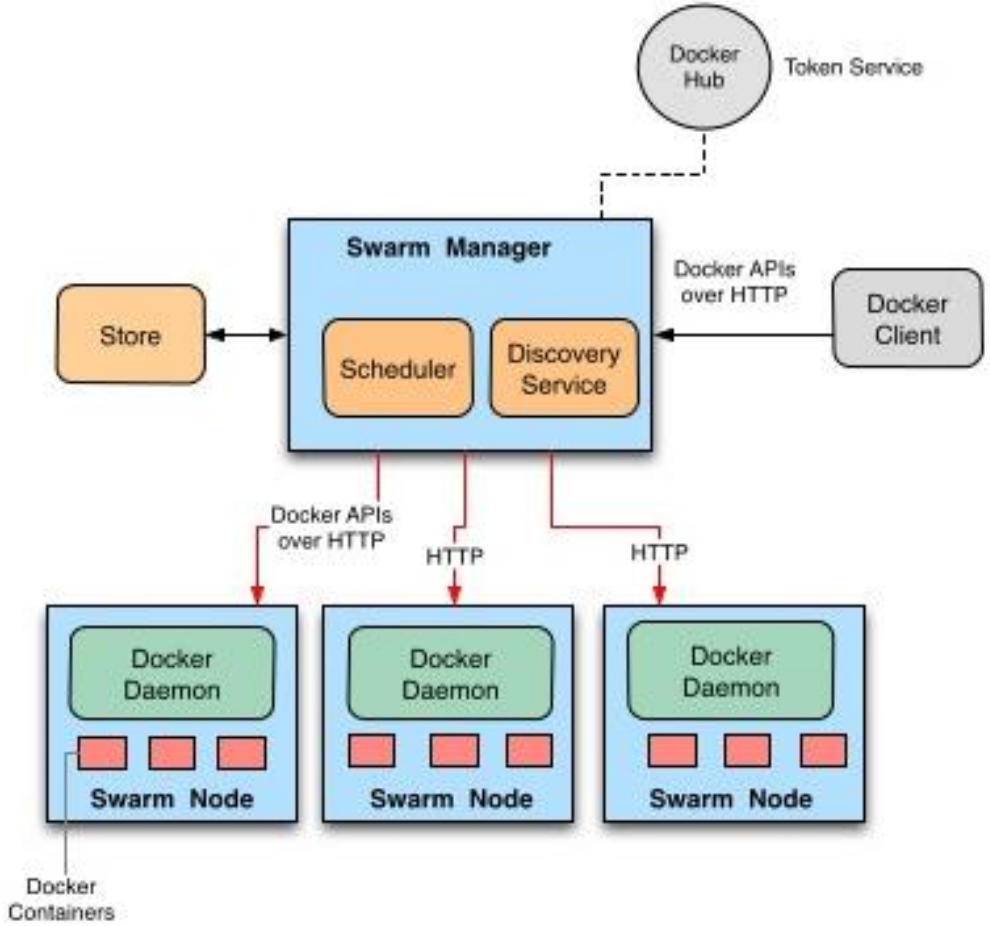
- Swarm is an orchestration tool to orchestrate your docker containers.
- Follows"swap, plug and play" principle.
- Allows you to create and access to a pool of Docker Hosts.
- For more info: http://www.slideshare.net/ramitsurana/introducing-docker-swarm-the-orchestration-tool-by-docker

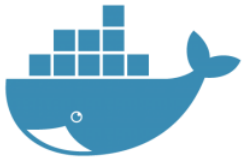


Docker Swarm working



Docker Swarm high level architecture





How to use Docker Swarm ?

- Pulling the image:

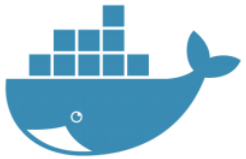
```
ramit@ramit-ramitsurana:~$ docker pull swarm
Using default tag: latest
latest: Pulling from library/swarm

eada7ab697d2: Pull complete
afaf40cb2366: Pull complete
7495da266907: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:12e3f7bdb86682733adf5351543487f581e1ccede5d85e1d5e0a7a62dcc88116
Status: Downloaded newer image for swarm:latest
```

- Creating new cluster using,

docker run –rm swarm create

```
ramit@ramit-ramitsurana:~$ docker run --rm swarm create
2f1a32011e21c853884c29e7ff8e3fe3
```

Cluster ID



How to use Docker Swarm ?

- Starting swarm agent,
- docker run -d swarm join --addr=<node_ip:2375> token://<cluster_id>

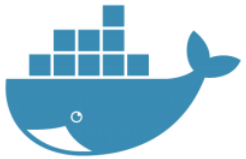
Default Port of Docker

```
ramit@ramit-ramitsurana:~$ docker run -d swarm join --addr=192.168.1.7:2375 token://2f1a32011e21c853884c29e7ff8e3fe3
5769c05edd699c3ce0a817197ff25c0631ee9967590fd76862ff153cc98e457e
```

- Setting up swarm on the other nodes:

```
ramit@ramit-ramitsurana:~$ docker run -d swarm join --addr=192.168.1.104:2375 token://2f1a32011e21c853884c29e7ff8e3f3
275d6fa96444ae171bf7b8538711da6e86a0954fdc29ed27fa510a4473306a70
```

```
ramit@ramit-ramitsurana:~$ docker run -d swarm join --addr=192.168.1.107:2375 token://2f1a32011e21c853884c29e7ff8e3f3
e5aa21c95fe6b36991c3a1e18c293a2f43e08d09d36e0a3269e2c01abdc7906d
```



How to use Docker Swarm ?

- To view a list of containers on port,

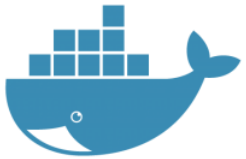
docker -H tcp://<ip-addr>:<port> ps

```
ramit@ramit-ramitsurana:~$ docker -H tcp://192.168.1.7:2375 ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS              PORTS
                    NAMES
e5aa21c95fe6        swarm                          "/swarm join --addr=1"   2 minutes ago       Up 2 minutes        2375/tcp
                    adoring_elion
275d6fa96444        swarm                          "/swarm join --addr=1"   6 minutes ago       Up 6 minutes        2375/tcp
                    compassionate_mcnulty
```

- To run a container on swarm,

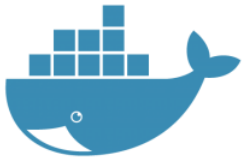
docker –H tcp://<ip-addr>:<port> run –d –name www –p <port>:<port> <image>

```
ramit@ramit-ramitsurana:~$ docker -H tcp://192.168.1.7:2375 run -d --name www -p 80:80 nginx
```



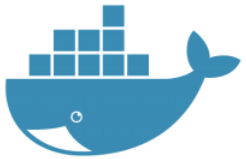
Docker Swarm Filters

- Tell Docker Swarm scheduler which nodes to use when creating and running a container.
- Consists of 5 categories:
- constraint
- health
- affinity
- dependency
- port



Some Docker Monitoring Tools

cAdvisor



cAdvisor

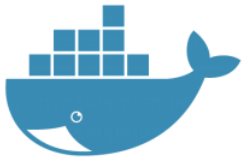
- Provided as open source by google.
- Easy to use
- Integrates well with kubernetes.
- Lightweight by nature

How to use cAdvisor ?

- Pulling docker image:

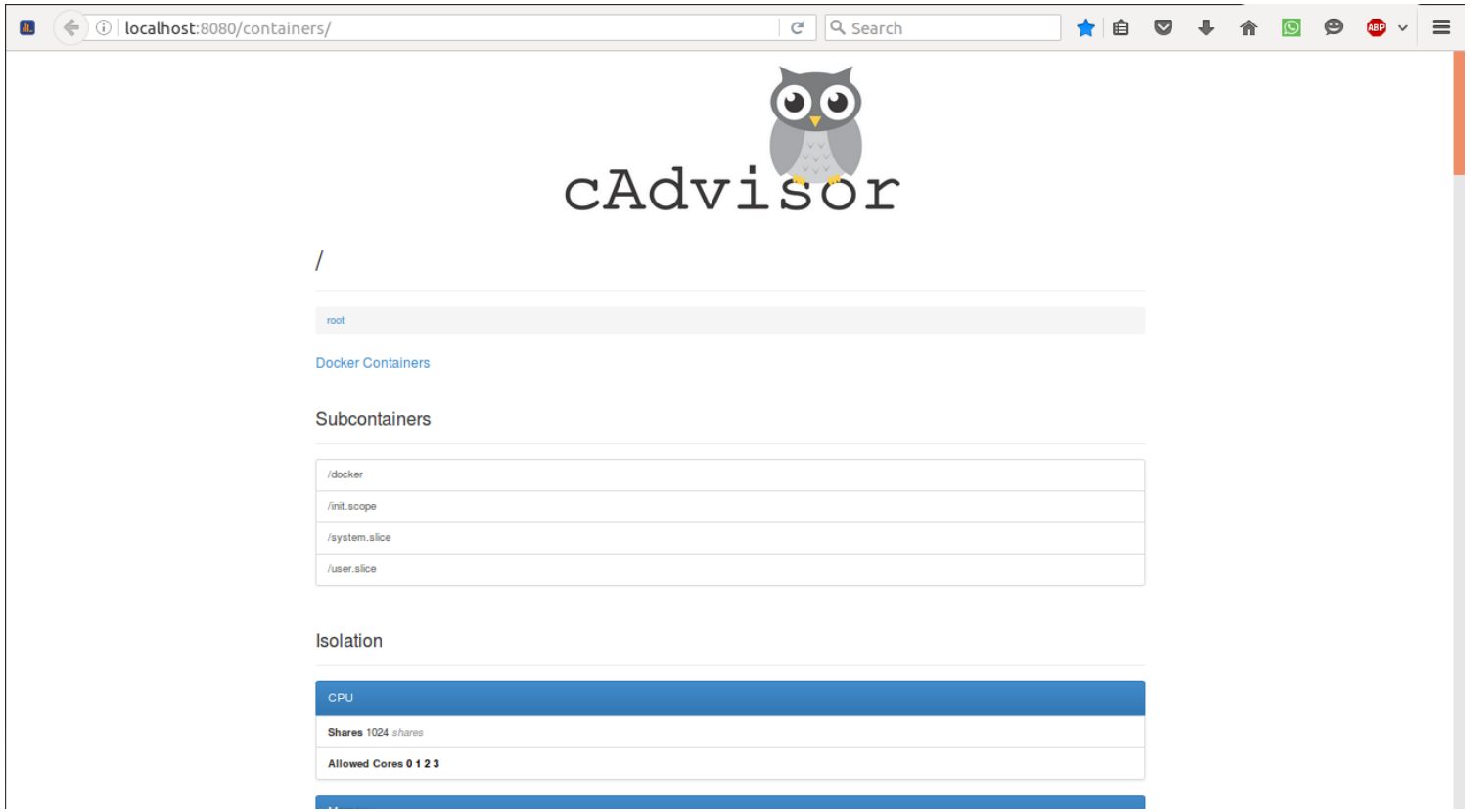
```
ramit@ramit-ramitsurana:~$ sudo docker run \
>    --volume=/:/rootfs:ro \
>    --volume=/var/run:/var/run:rw \
>    --volume=/sys:/sys:ro \
>    --volume=/var/lib/docker/:/var/lib/docker:ro \
>    --publish=8080:8080 \
>    --detach=true \
>    --name=cadvisor \
>   google/cadvisor:latest
Unable to find image 'google/cadvisor:latest' locally
latest: Pulling from google/cadvisor

09d0220f4043: Pull complete
151807d34af9: Pull complete
14cd28dce332: Pull complete
Digest: sha256:8364c7ab7f56a087b757a304f9376c3527c8c60c848f82b66dd728980222bd2f
Status: Downloaded newer image for google/cadvisor:latest
133f4d545faa9f11c1f2cae0443bcf7d5c7c1a8653af2303f07bd6396ac9e032
```

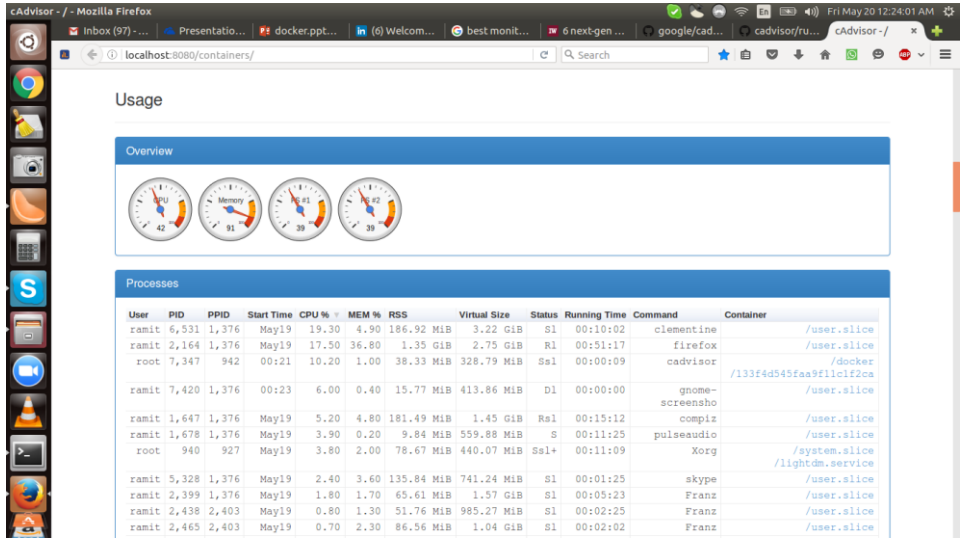
- Try http://localhost:8080 on your browser

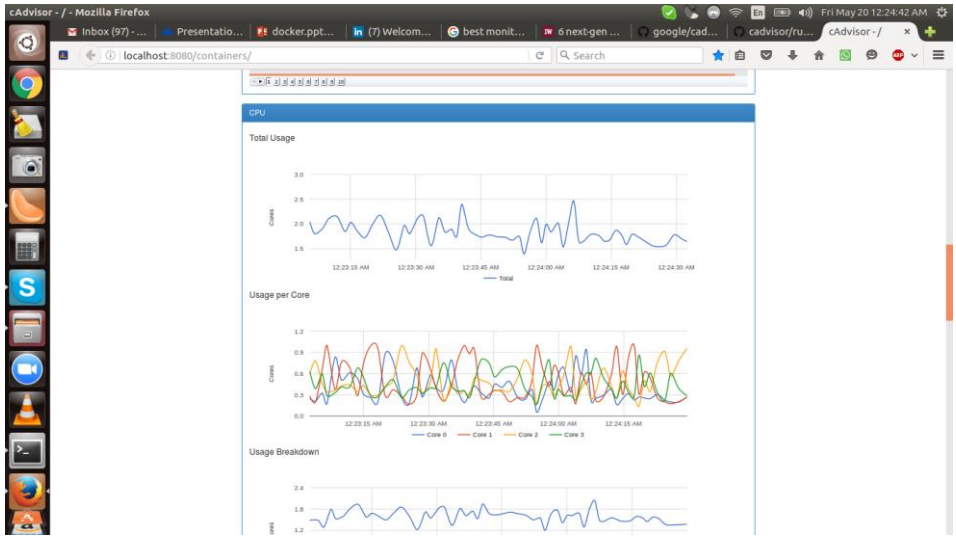


How to use cAdvisor ?

- My Results :







Shipyard

Shipyard

- Built by Evan Hazlett.
- Built on Docker Swarm
- Gives you the ability to manage Docker resources including containers, images, private registries and more.
- Requires installation of :

- Rethinkdb
- Etcd
- docker-proxy
- Shipyard image
- Swarm manager, agent & controller

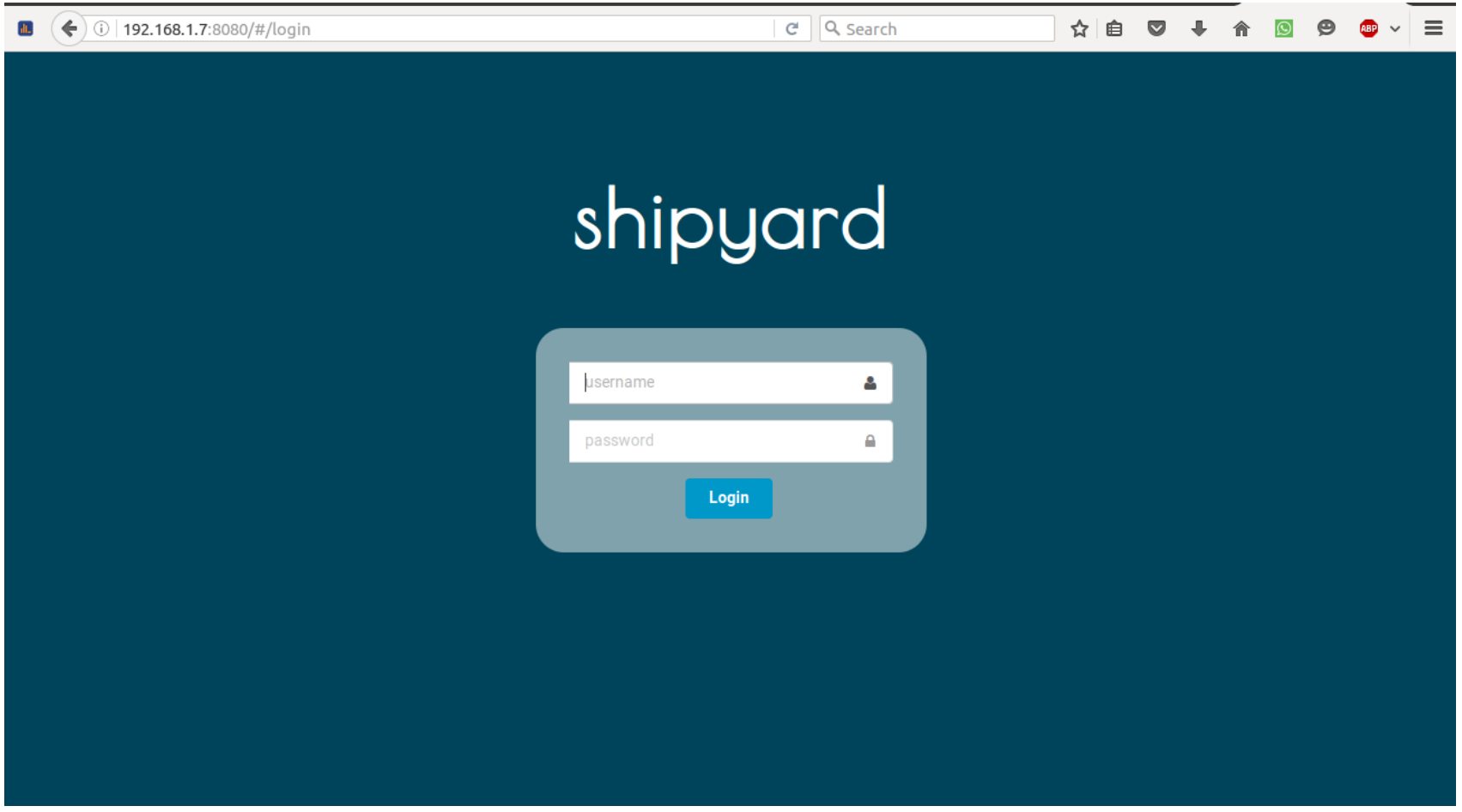
How to use Shipyard ?

- Using Shell script:

```
ramit@ramit-ramitsurana:~$ curl -s https://shipyard-project.com/deploy | bash -s
Deploying Shipyard
 -> Starting Database
Unable to find image 'rethinkdb:latest' locally
latest: Pulling from library/rethinkdb
8b87079b7a06: Pulling fs layer
a3ed95caeb02: Pulling fs layer
bbc284a5d4bf: Pulling fs layer
7de707c60810: Pulling fs layer
96104b6c9bd4: Pulling fs layer
bbc284a5d4bf: Waiting
7de707c60810: Waiting
96104b6c9bd4: Waiting
bbc284a5d4bf: Download complete
a3ed95caeb02: Download complete
7de707c60810: Verifying Checksum
7de707c60810: Download complete
96104b6c9bd4: Verifying Checksum
96104b6c9bd4: Download complete
8b87079b7a06: Download complete
8b87079b7a06: Pull complete
8b87079b7a06: Pull complete
a3ed95caeb02: Pull complete
a3ed95caeb02: Pull complete
bbc284a5d4bf: Pull complete
bbc284a5d4bf: Pull complete
7de707c60810: Pull complete
7de707c60810: Pull complete
96104b6c9bd4: Pull complete
96104b6c9bd4: Pull complete
Digest: sha256:241a11bfd3fccaa114cdf8e527944f2bb5c91fa728ac751b54d3ff56dbb8ce21
Status: Downloaded newer image for rethinkdb:latest
 -> Starting Discovery
Unable to find image 'microbox/etcd:latest' locally
latest: Pulling from microbox/etcd
8ded6e8ab3fd: Pulling fs layer
bf8f85223d7a: Pulling fs layer
a3ed95caeb02: Pulling fs layer
a3ed95caeb02: Download complete
8ded6e8ab3fd: Verifying Checksum
8ded6e8ab3fd: Download complete
8ded6e8ab3fd: Pull complete
8ded6e8ab3fd: Pull complete
```

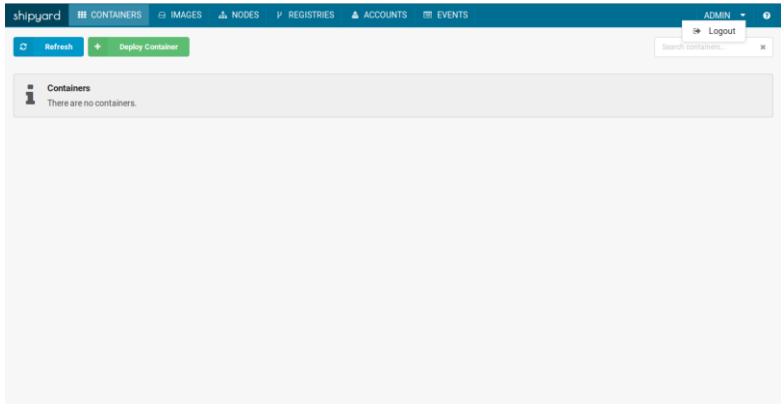
How to use Shipyard?

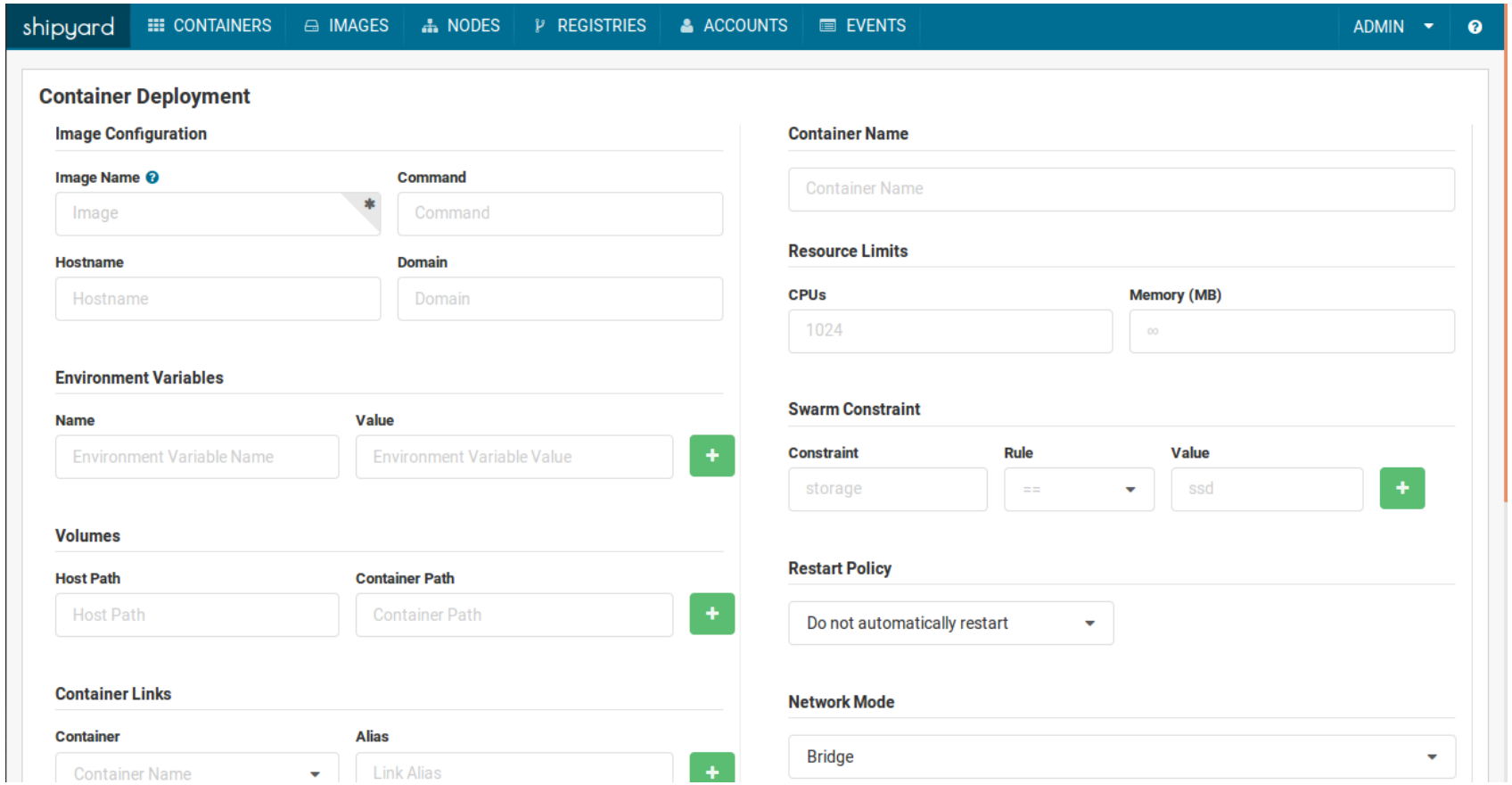
- Open up http://:<your local ip>:8080

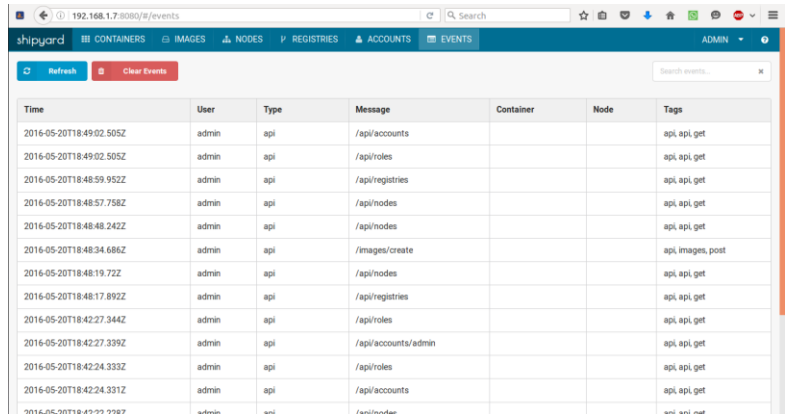


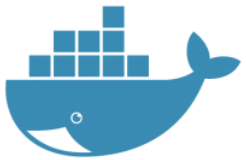
How to use Shipyard?

- Use admin for username & shipyard for password







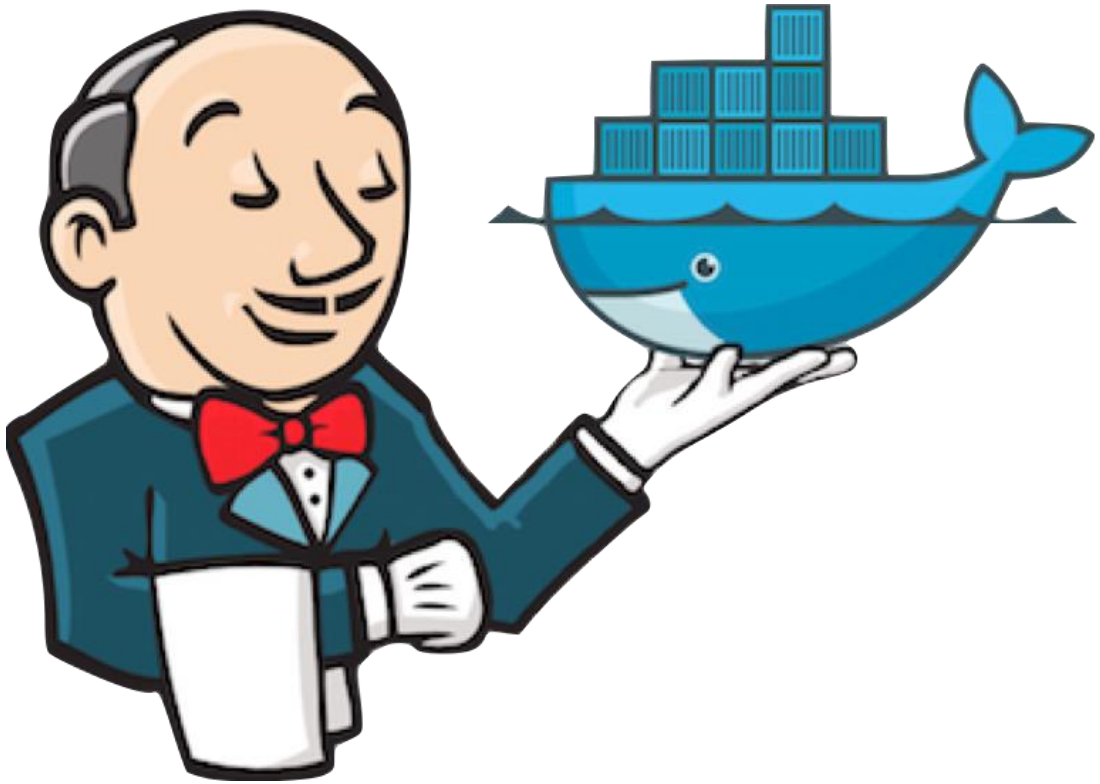


CI/CD the Docker Way

Developing CI/CD

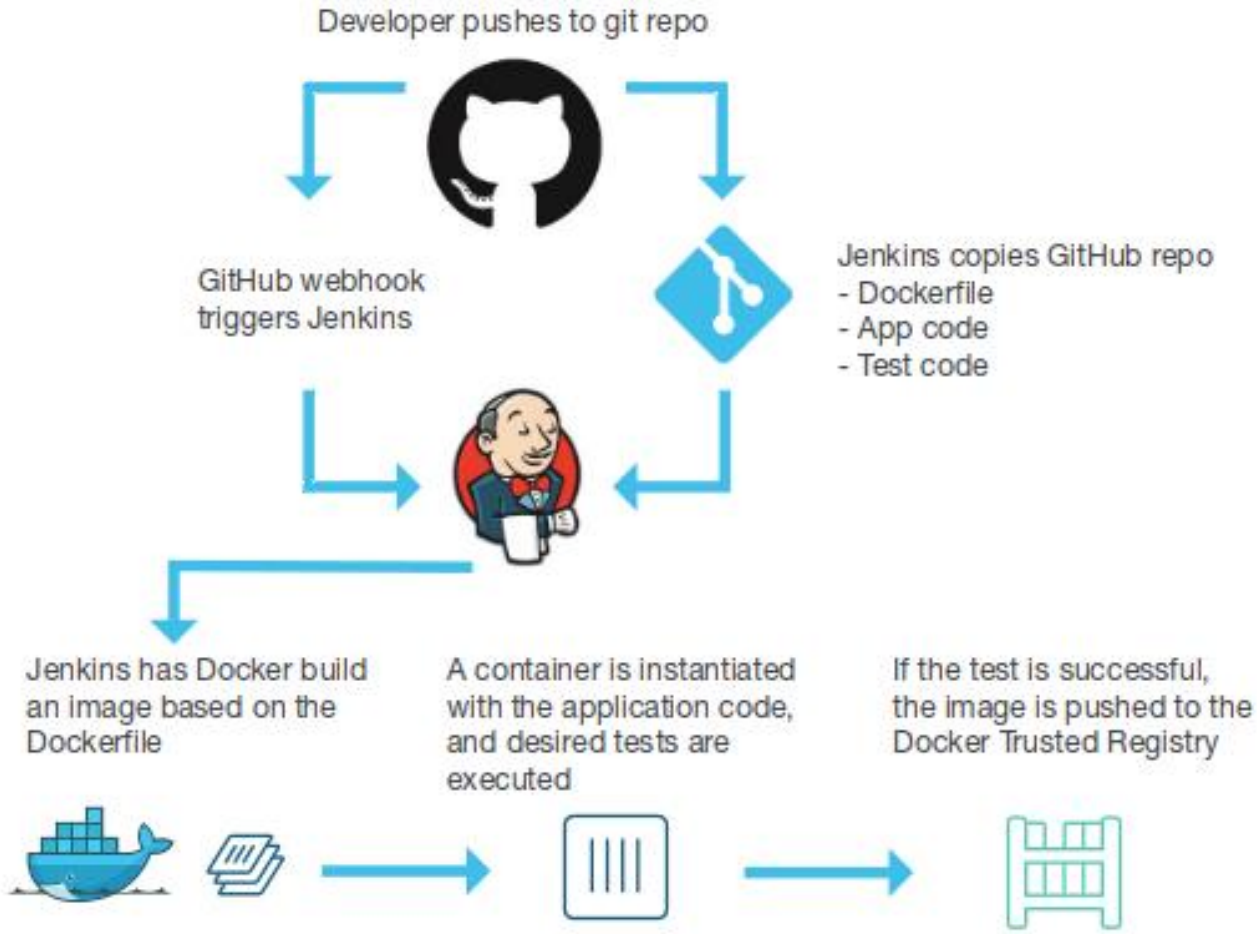
- Docker version 2.0 introduces new UI.
- Jenkinsfile introduced with this new version.
- It's a container for your pipeline which details what specific steps are needed to perform a job for which you want to use Jenkins.
- For better info:

http://theremotelab.com /blog/jenkins2.0-screencast-series/

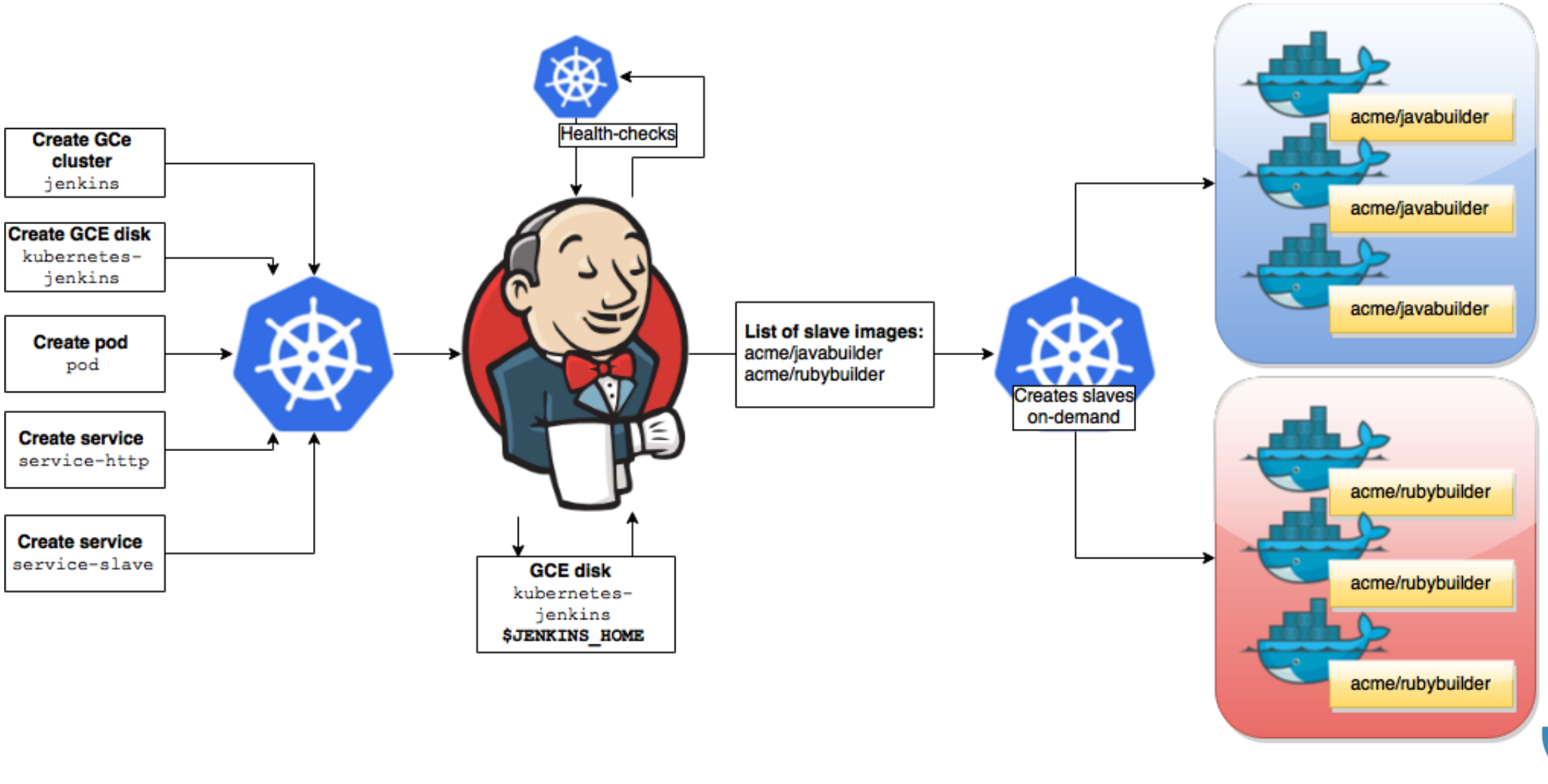


Sample Workflow model

Workflow

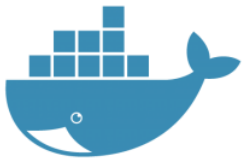


Better workflow model (using kubernetes)



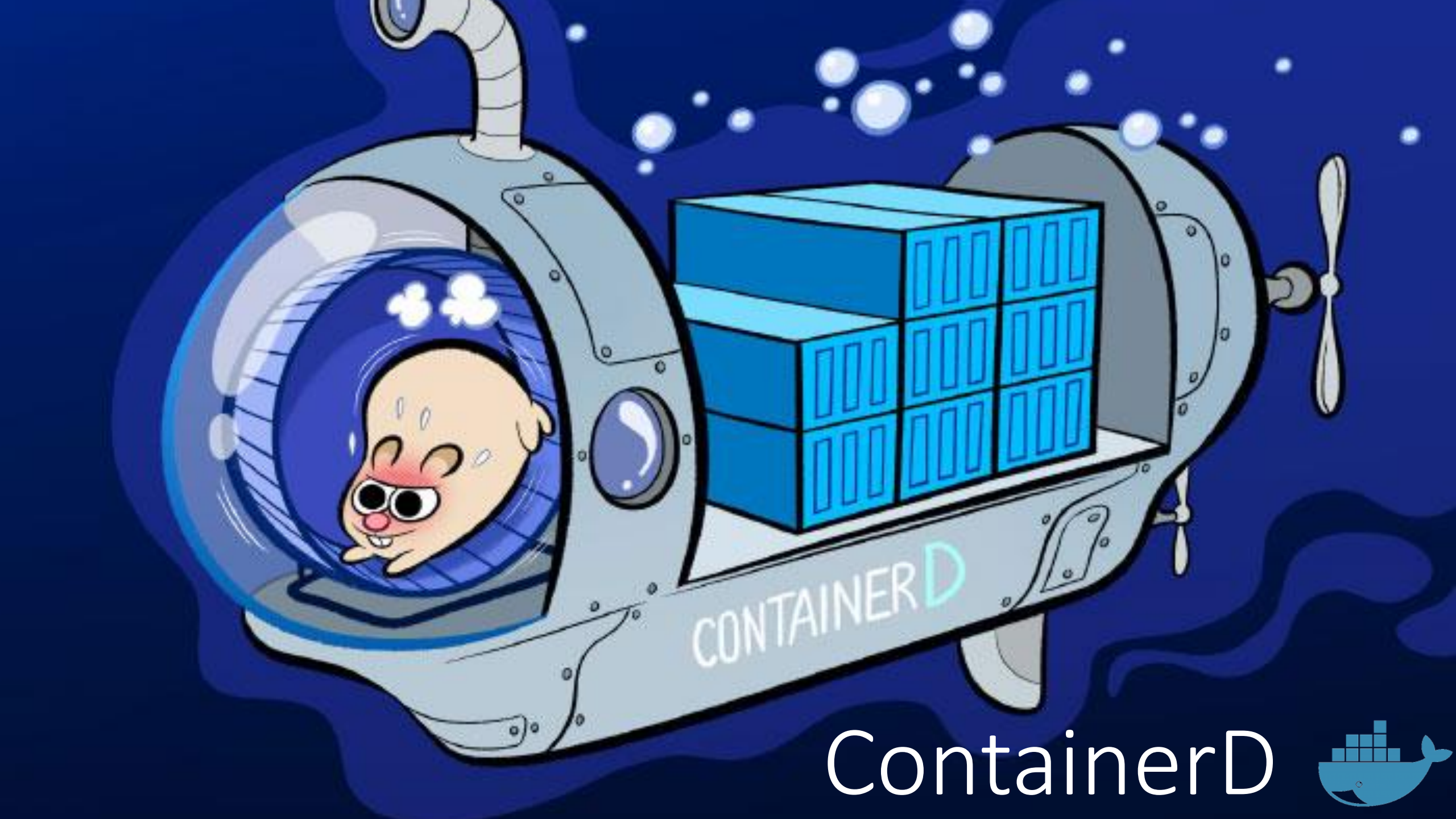
Dockerv1.1 introduces major updates

- From Dockerv1.1 new things have come into actions.
- Some major changes are
- RunC
- Containerd
- OCI Standards



runC

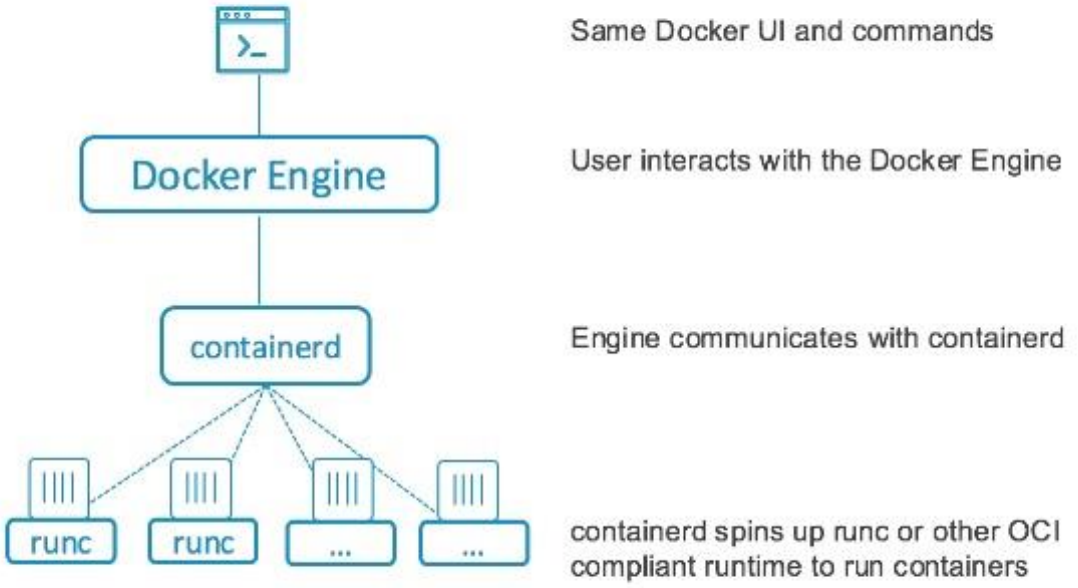
- CLI tool for spawning and running containers.
- runC does not create a daemon, so it integrates well with systemd.
- Currently in alpha version



ContainerD

Containerd

- Daemon with an API and a command line client, to manage containers on one machine.
- Uses runC to run containers.
- Has advanced features such as seccomp and user namespace support as well as checkpoint and restore for cloning and live migration of containers.

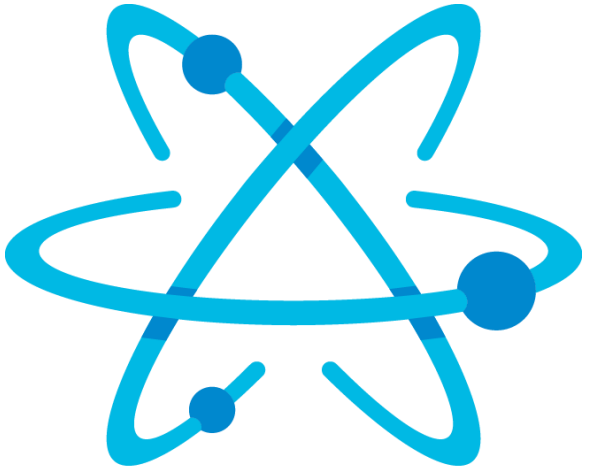


Some more services from docker

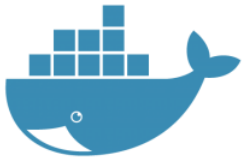
- Docker Datacenter
- Kitematic
- Docker Cloud a.k.a. Tumtum cloud
- Docker Toolbox
- And many more....

Some interesting new operating systems for running docker

- CoreOS
- RancherOS
- Atomic OS
- Alpine

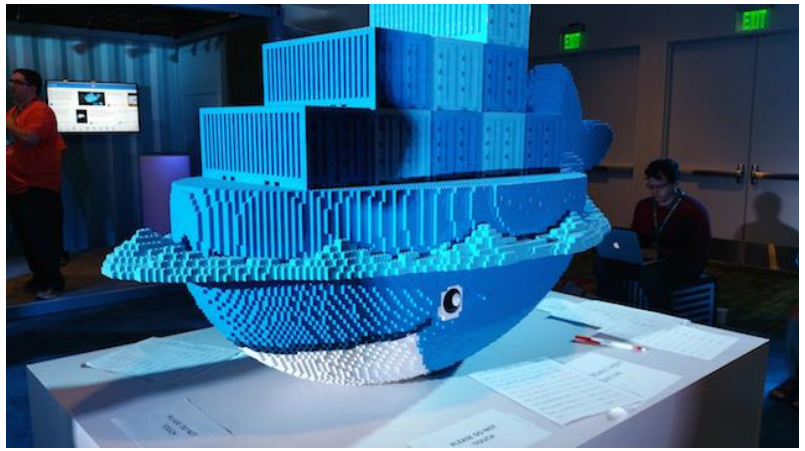


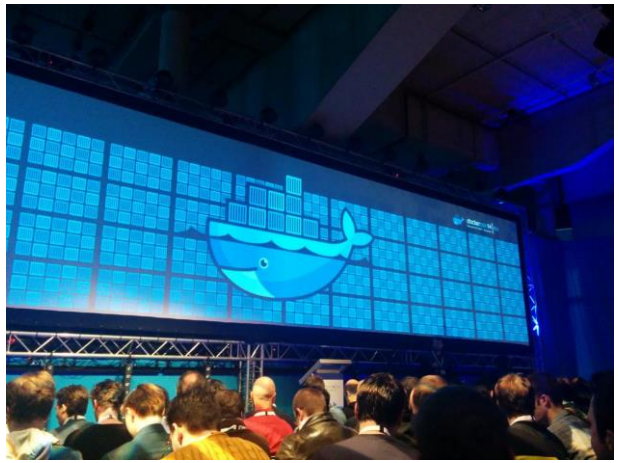
RED HAT ATOMIC



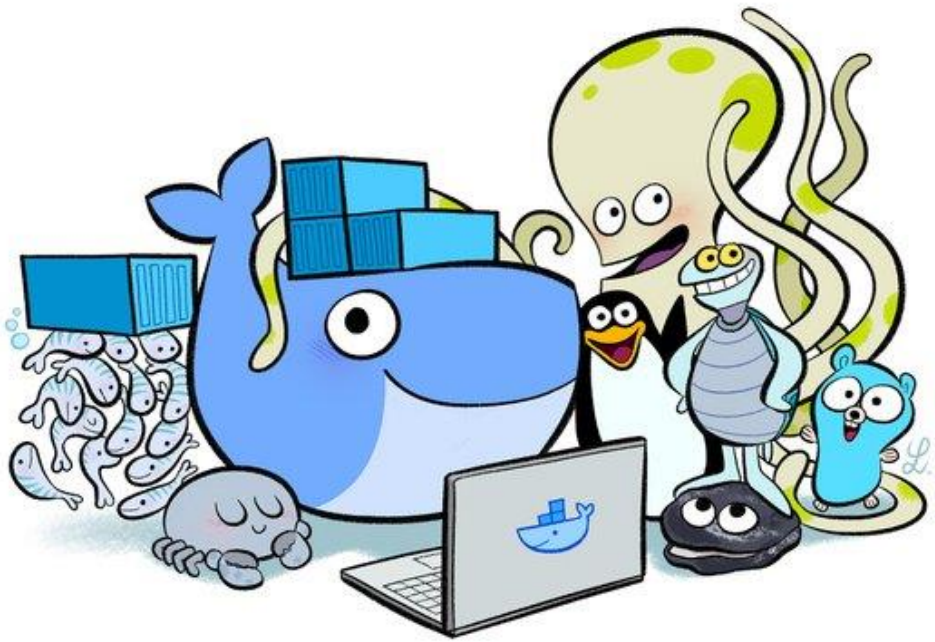
Docker Questions ?

Docker Celebrations & conference





Thank you



May you have a Dockerized day ahead !