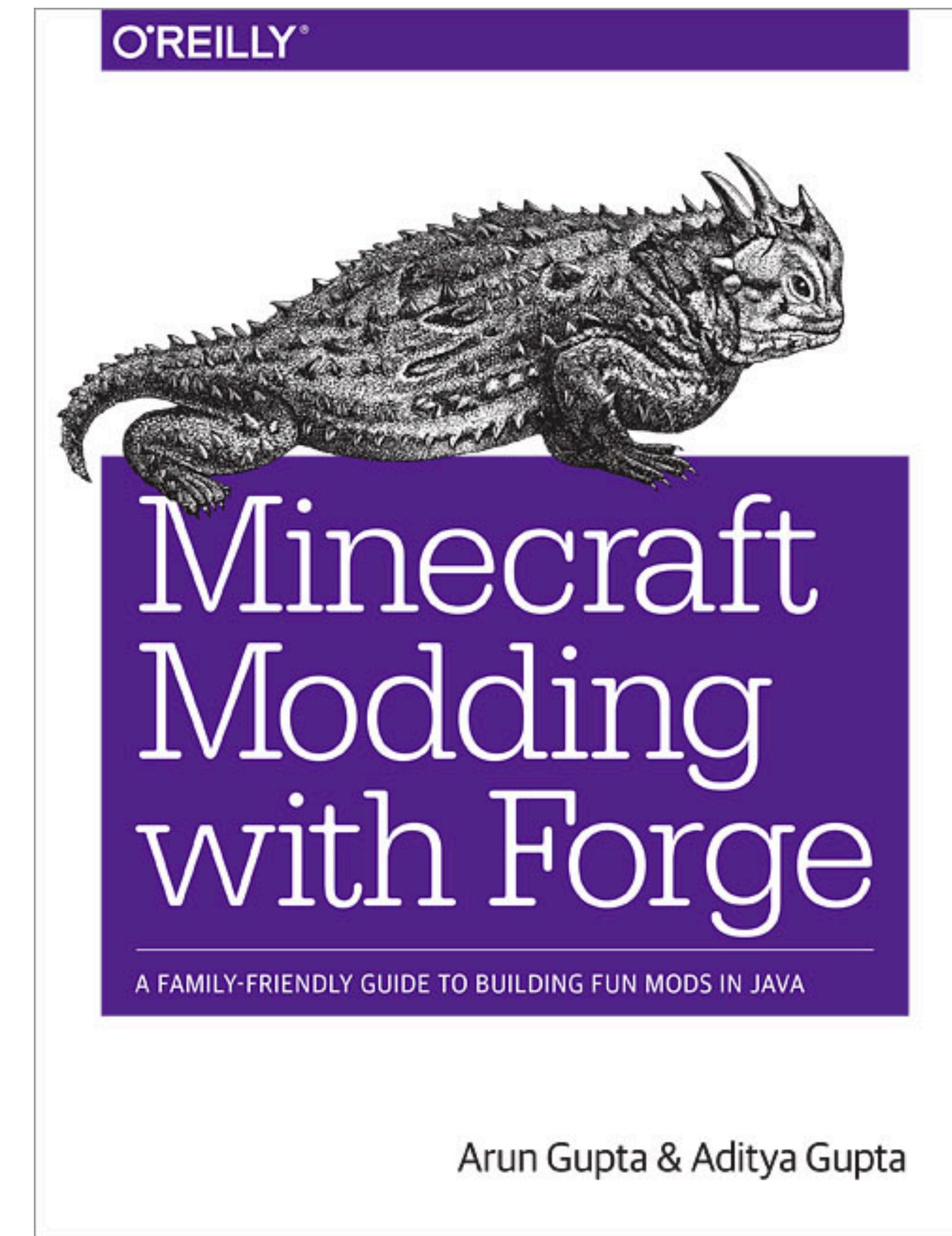


docker

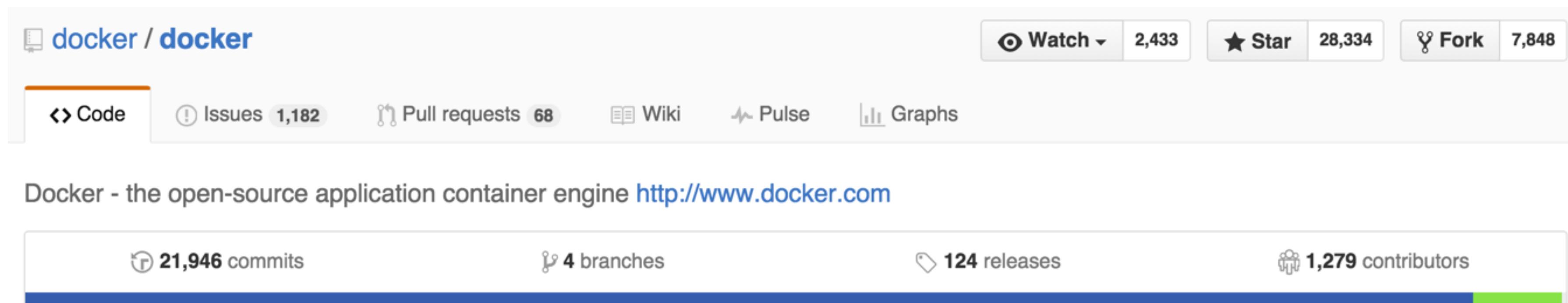
Getting Started with Docker

Arun Gupta, @arungupta
VP Developer Advocacy, Couchbase

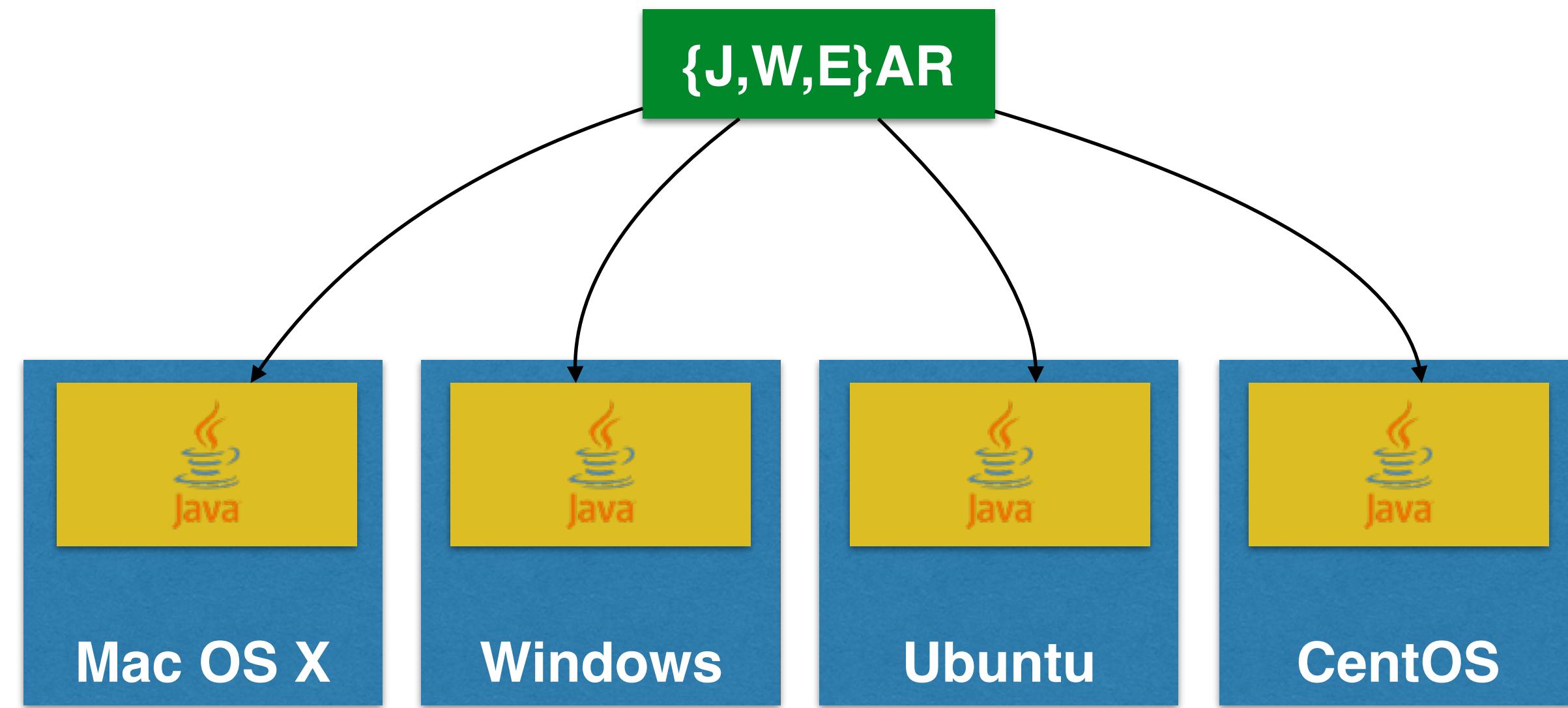


What is Docker?

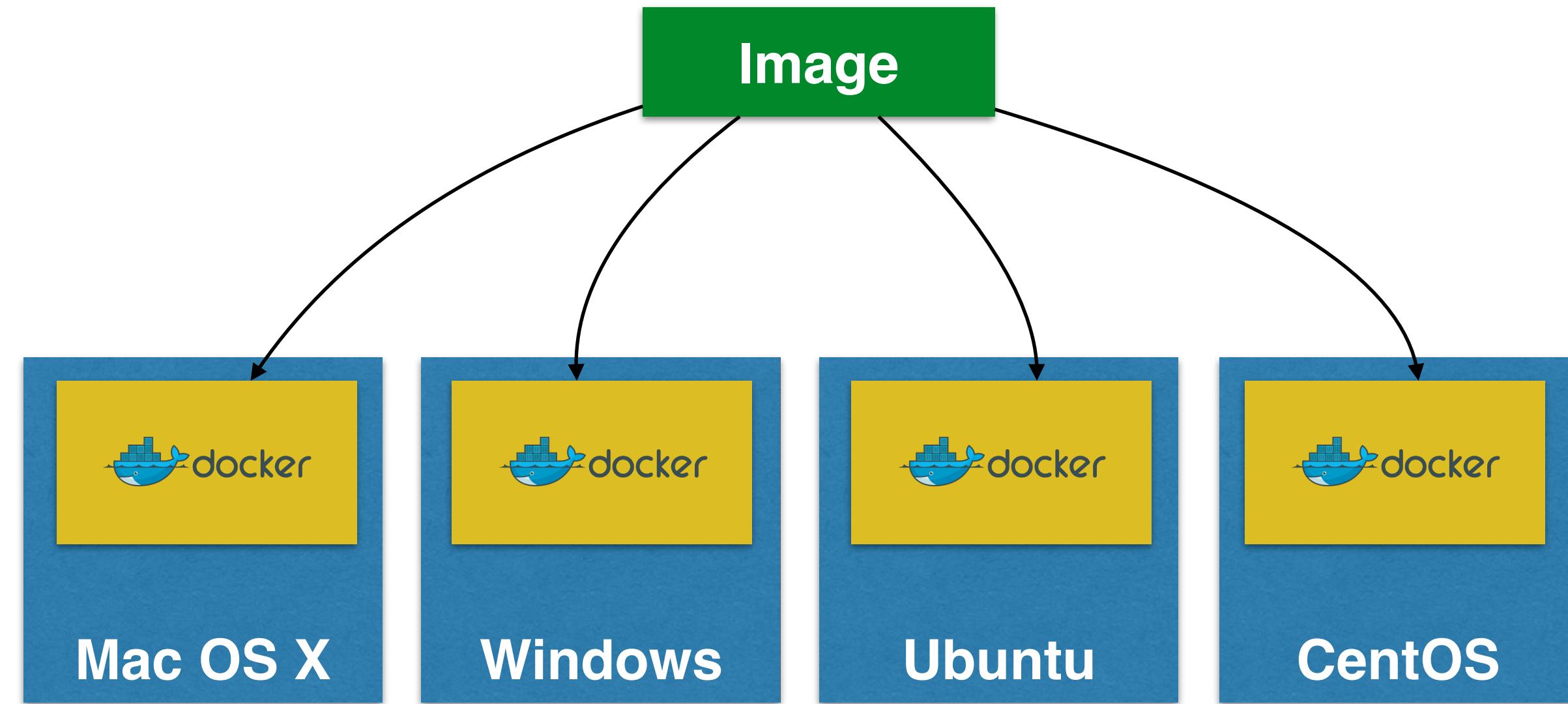
- Open source project and company



- Used to create containers for software applications
- Package Once Deploy Anywhere (PODA)

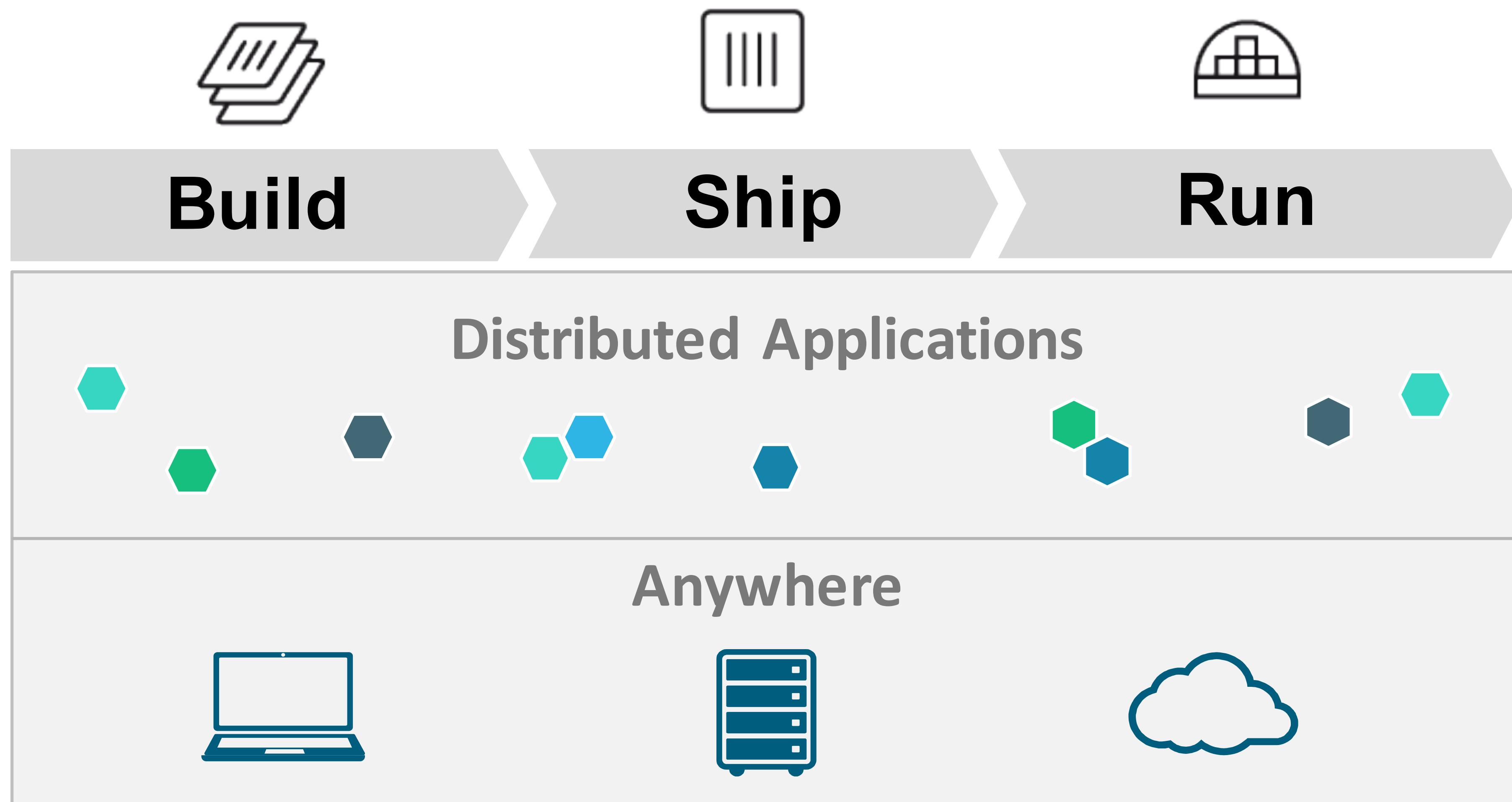


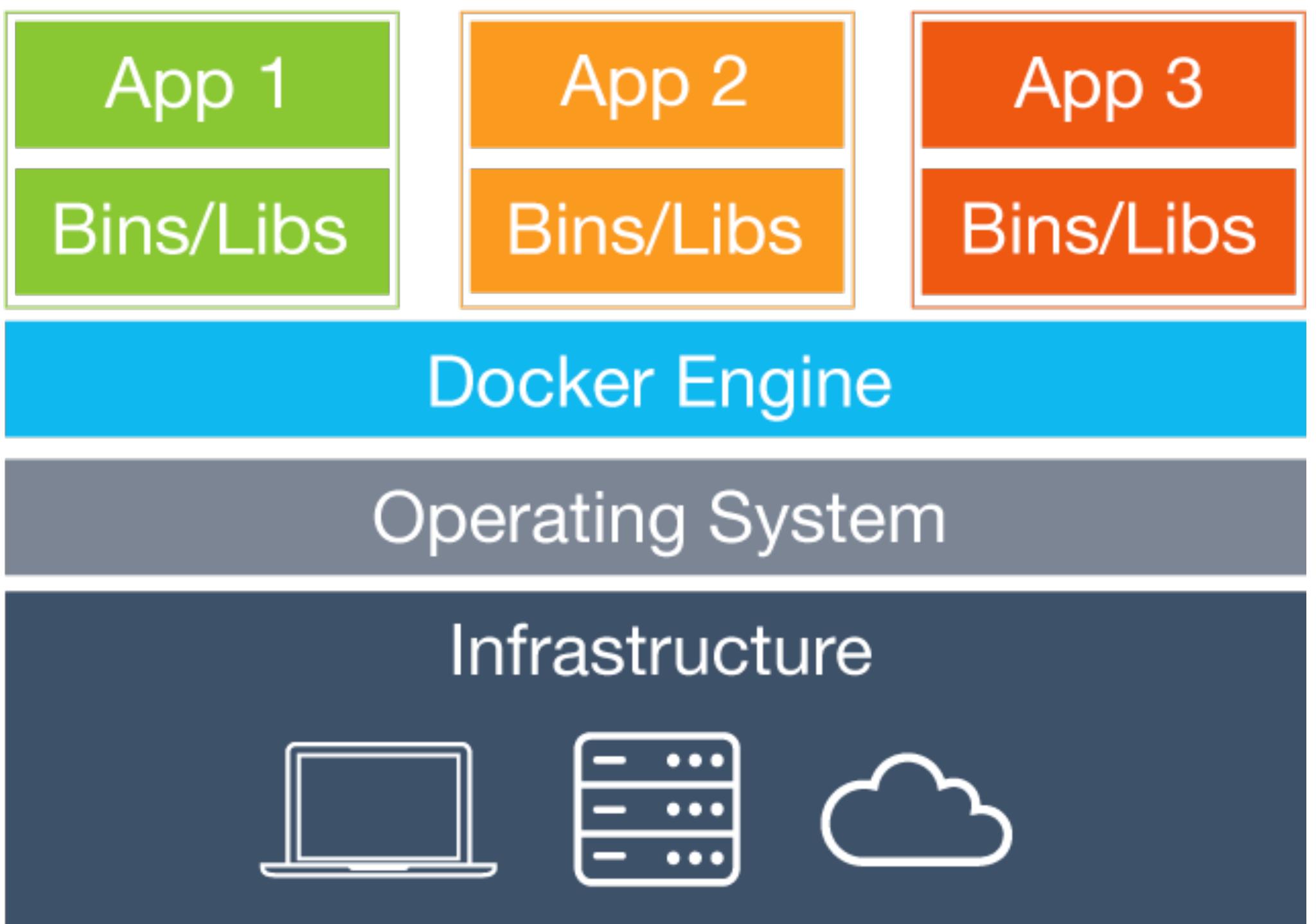
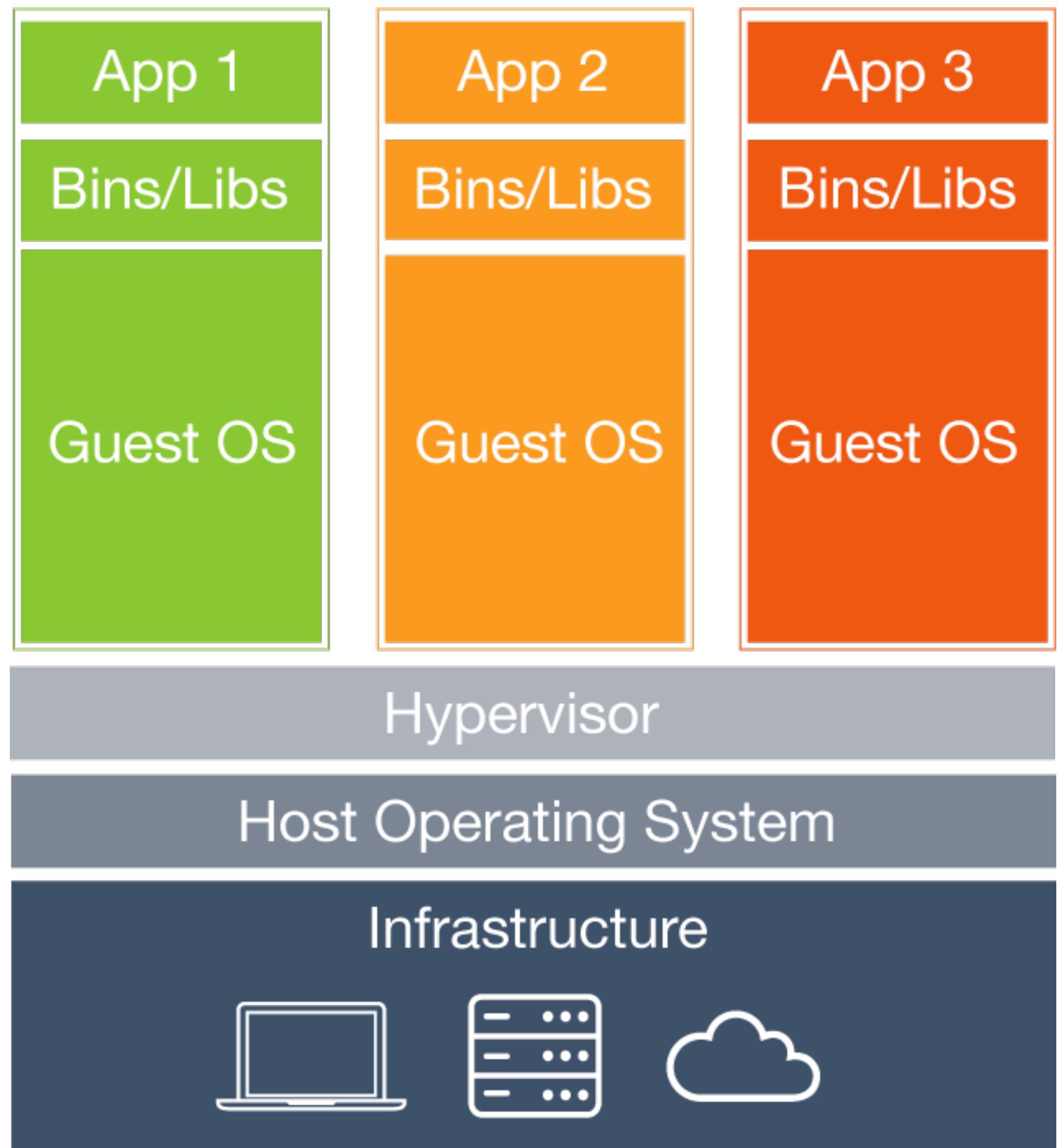
WORA = Write Once Run Anywhere



PODA = Package Once Deploy Anywhere

Docker Mission







Build

Develop an app using Docker containers with any language and any toolchain.

```
FROM ubuntu

CMD echo "Hello world"
```

```
FROM jboss/wildfly

RUN curl -L https://github.com/javaee-samples/javaee7-hol/raw/master/solution/movieplex7-1.0-SNAPSHOT.war -o /opt/jboss/wildfly/standalone/deployments/movieplex7-1.0-SNAPSHOT.war
```

Dockerfile reference

Usage

Format

 Environment replacement

 .dockerignore file

FROM

MAINTAINER

RUN

 Known issues (RUN)

CMD

LABEL

EXPOSE

ENV

ADD

COPY

ENTRYPOINT

 Exec form ENTRYPOINT example

 Shell form ENTRYPOINT example

VOLUME

USER

WORKDIR

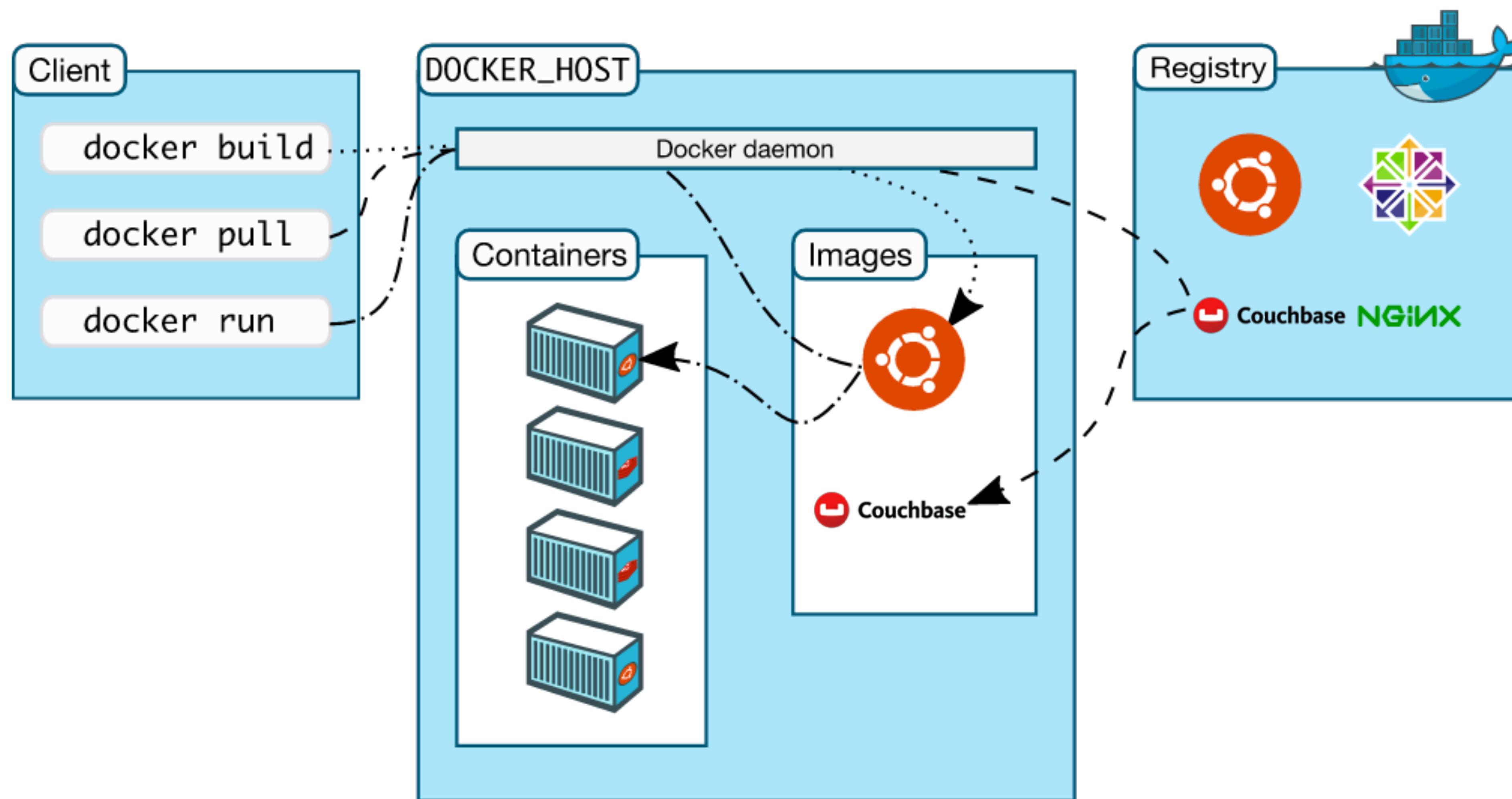
ARG

ONBUILD

STOPSIG

Dockerfile examples

Docker Workflow



Union File System

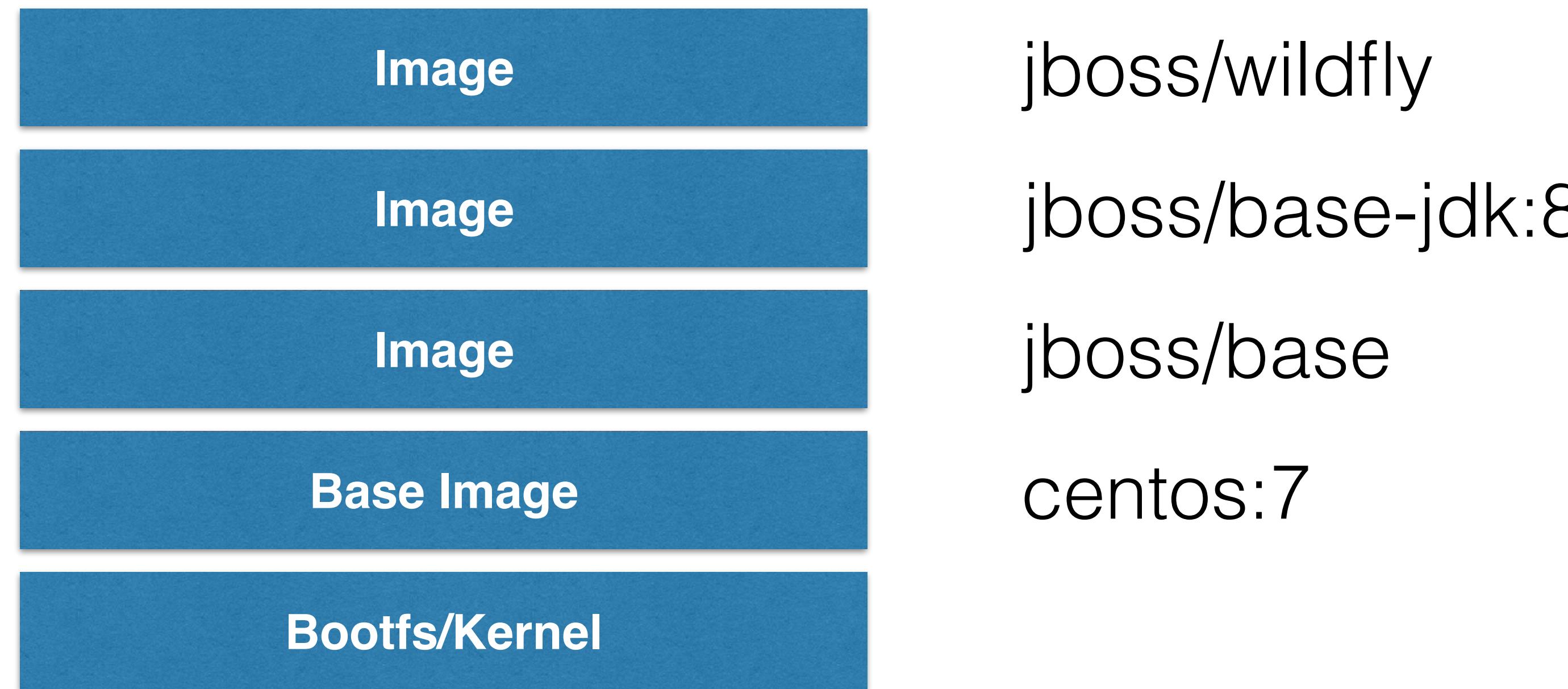


Image Layers - Couchbase

```
~ > docker images couchbase
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
couchbase	latest	45abdd57689a	3 weeks ago	372 MB

```
~ > docker history couchbase
```

IMAGE	CREATED	CREATED BY	SIZE
45abdd57689a	3 weeks ago	/bin/sh -c #(nop) VOLUME [/opt/couchbase/var]	0 B
dd8c5611343d	3 weeks ago	/bin/sh -c #(nop) EXPOSE 11207/tcp 11210/tcp	0 B
30852bbad62b	3 weeks ago	/bin/sh -c #(nop) CMD ["couchbase-server"]	0 B
5537747ea12f	3 weeks ago	/bin/sh -c #(nop) ENTRYPOINT &{ ["/entrypoint.	0 B
e8a83a5448df	3 weeks ago	/bin/sh -c #(nop) COPY file:cbb44c9c65b64a9dc	182 B
18165b90fef9	3 weeks ago	/bin/sh -c #(nop) COPY file:34e32c52f0895191f	389 B
5f37b8bdc5a6	3 weeks ago	/bin/sh -c wget -N \$CB_RELEASE_URL/\$CB_VERSI0	212.1 MB
1a8da511d01b	3 weeks ago	/bin/sh -c groupadd -g 1000 couchbase && user	328.7 kB
d9b2222c39b4	3 weeks ago	/bin/sh -c #(nop) ENV CB_VERSION=4.0.0 CB_REL	0 B
815f08b3c781	3 weeks ago	/bin/sh -c apt-get update && apt-get inst	23.57 MB
fc38f156c0ea	3 weeks ago	/bin/sh -c #(nop) MAINTAINER Couchbase Docker	0 B
2a7a952931ec	3 weeks ago	/bin/sh -c #(nop) CMD ["/bin/bash"]	0 B
10f1b5844a9c	3 weeks ago	/bin/sh -c sed -i 's/^#\s*/(deb.*universe\)\$/'	1.911 kB
23c388b926b6	3 weeks ago	/bin/sh -c echo '#!/bin/sh' > /usr/sbin/polic	156.2 kB
b45376f323f5	3 weeks ago	/bin/sh -c #(nop) ADD file:4a9e089e81d6581a54	135.9 MB

Image Layers - WildFly

~ > docker images jboss/wildfly		IMAGE ID	CREATED	VIRTUAL SIZE	
REPOSITORY	TAG	be97bf33ad55	2 weeks ago	584.5 MB	
jboss/wildfly	latest				
~ > docker history jboss/wildfly		CREATED BY		SIZE	COMMENT
IMAGE	CREATED	/bin/sh -c #(nop) CMD ["/opt/jboss/wildfly/bi	0 B		
be97bf33ad55	2 weeks ago	/bin/sh -c #(nop) EXPOSE 8080/tcp	0 B		
fd24a9f17c67	2 weeks ago	/bin/sh -c #(nop) ENV LAUNCH_JBOSS_IN_BACKGR0	0 B		
f5dea43156ba	2 weeks ago	/bin/sh -c cd \$HOME && curl -0 https://do	159.1 MB		
b17d8182afad	2 weeks ago	/bin/sh -c #(nop) ENV JBOSS_HOME=/opt/jboss/w	0 B		
00429dab510a	2 weeks ago	/bin/sh -c #(nop) ENV WILDFLY_SHA1=75738379f7	0 B		
ca2596a51c35	2 weeks ago	/bin/sh -c #(nop) ENV WILDFLY_VERSION=9.0.2.F	0 B		
62de294133f4	2 weeks ago	/bin/sh -c #(nop) ENV JAVA_HOME=/usr/lib/jvm/	0 B		
52940fa06773	3 months ago	/bin/sh -c #(nop) USER [jboss]	0 B		
bda081e220a9	3 months ago	/bin/sh -c yum -y install java-1.8.0-openjdk-	195.3 MB		
1681ffdbbe50	3 months ago	/bin/sh -c #(nop) USER [root]	0 B		
3b4e1d18fb41	3 months ago	/bin/sh -c #(nop) MAINTAINER Marek Goldmann <	0 B		
481c174e3d0d	3 months ago	/bin/sh -c #(nop) USER [jboss]	0 B		
722c8bdf4d33	5 months ago	/bin/sh -c #(nop) WORKDIR /opt/jboss	0 B		
88389be452f6	5 months ago	/bin/sh -c groupadd -r jboss -g 1000 && usera	294.8 kB		
e5351df2706d	5 months ago	/bin/sh -c yum update -y && yum -y install xm	57.58 MB		
e3ff94cddd8b	5 months ago	/bin/sh -c #(nop) MAINTAINER Marek Goldmann <	0 B		
92009d6c8aff	5 months ago	/bin/sh -c #(nop) CMD ["/bin/bash"]	0 B		
7ed0005a64f9	5 months ago	/bin/sh -c #(nop) ADD file:82835f82606420c764	172.2 MB		
4d1ec7e2b1dd	5 months ago	/bin/sh -c #(nop) MAINTAINER The CentOS Proje	0 B		
a2c33fe967de	7 months ago				



Docker Machine

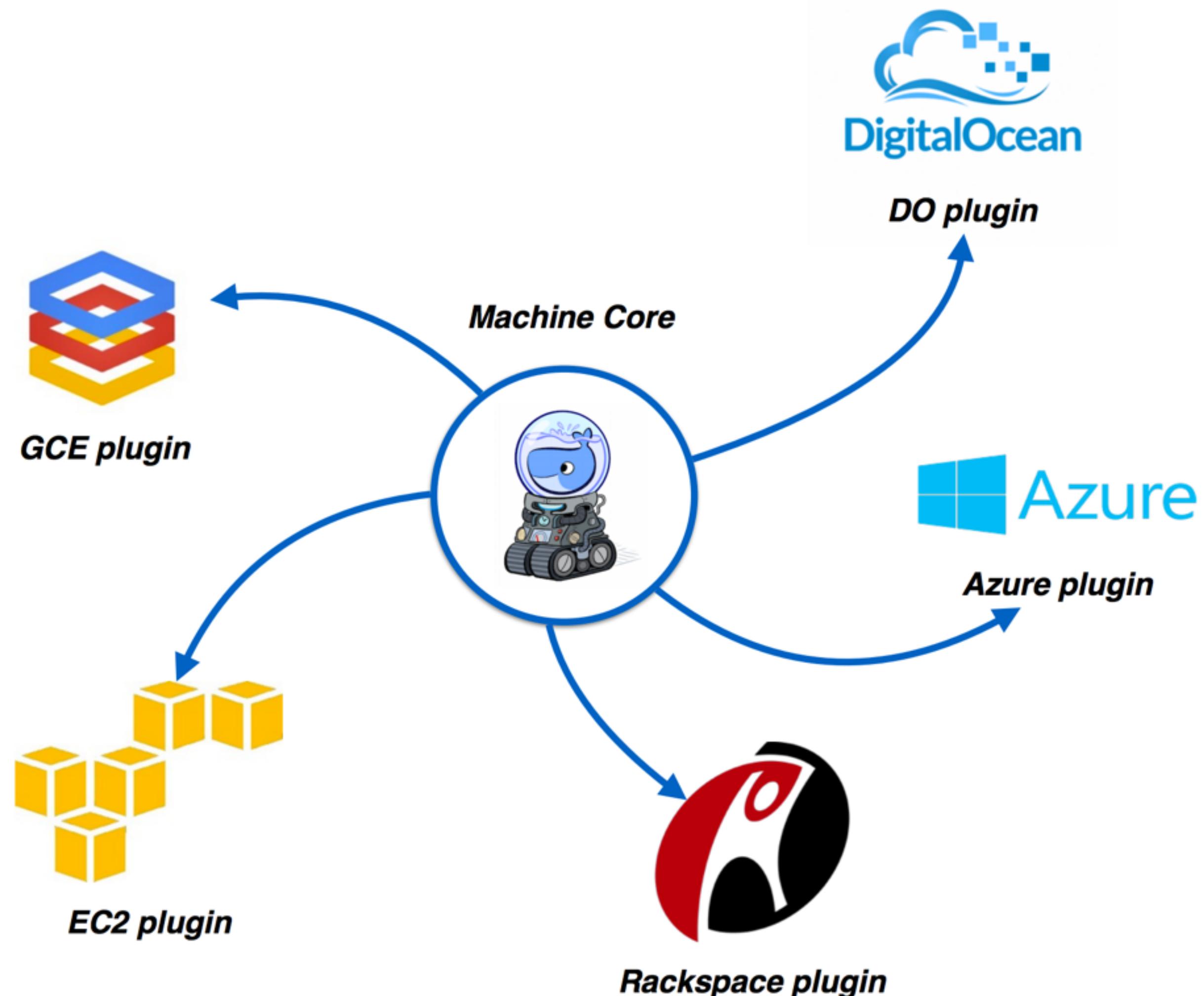
- Create Docker Host on computer or cloud provider

```
docker-machine create --driver=virtualbox myhost
```

- Configure Docker client to talk to host
- Create and pull images
- Start, stop, restart containers
- Upgrade Docker

- Not recommended for production yet

Docker Machine Providers



boot2docker

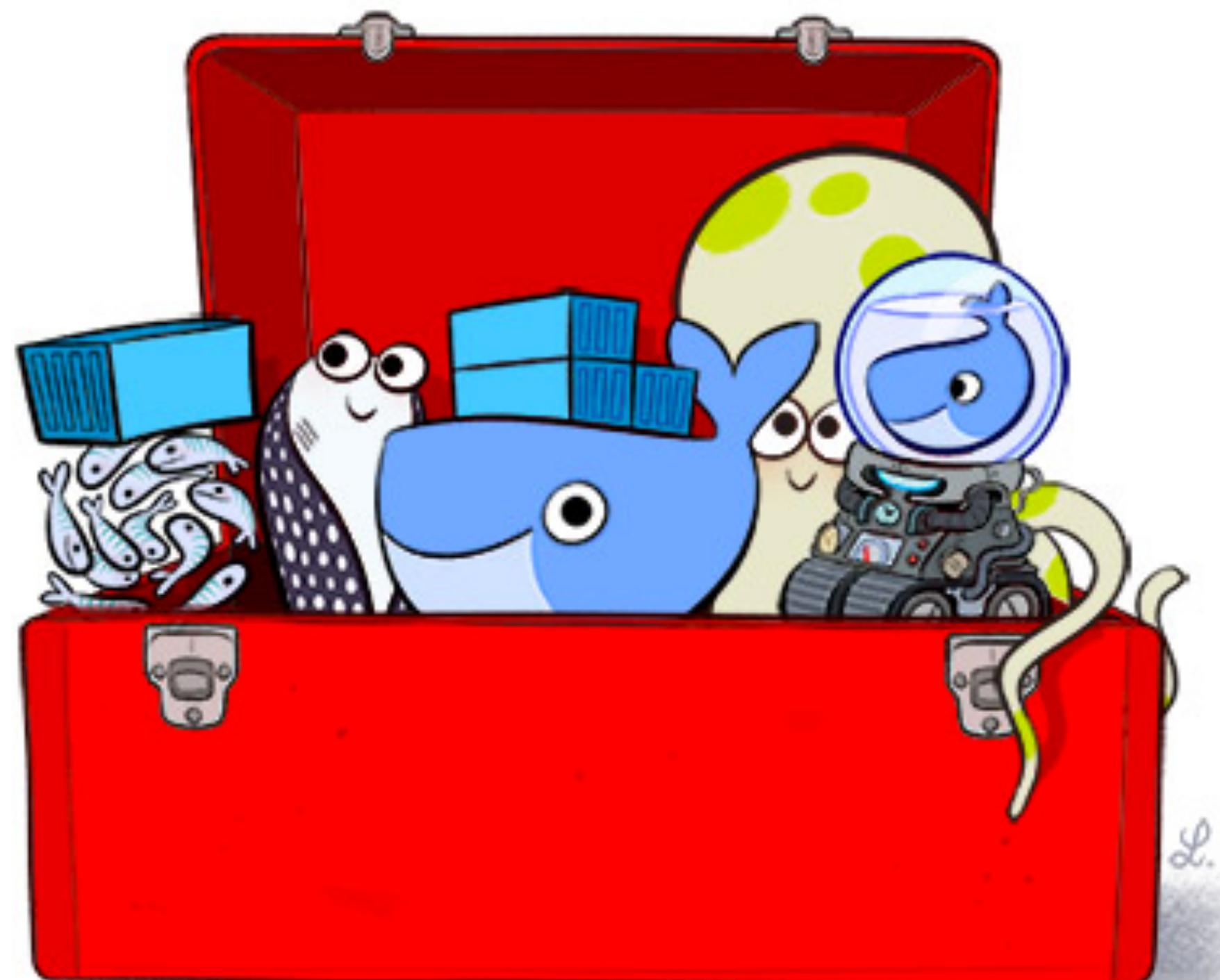
- Migrate to Docker Machine

```
docker-machine  
create -d virtualbox  
--virtualbox-import-boot2docker-vm  
boot2docker-vm docker-vm
```

boot2docker	docker-machine	docker-machine description
init	create	Creates a new docker host.
up	start	Starts a stopped machine.
ssh	ssh	Runs a command or interactive ssh session on the machine.
save	-	Not applicable.
down	stop	Stops a running machine.
poweroff	stop	Stops a running machine.
reset	restart	Restarts a running machine.
config	inspect	Prints machine configuration details.
status	ls	Lists all machines and their status.
info	inspect	Displays a machine's details.
ip	ip	Displays the machine's ip address.
shellinit	env	Displays shell commands needed to configure your shell to interact with a machine
delete	rm	Removes a machine.
download	-	Not applicable.
upgrade	upgrade	Upgrades a machine's Docker client to the latest stable release.

Docker Toolbox

- Docker Client 1.10.0
- Docker Machine 0.6.0
- Docker Compose 1.6.0
- Docker Kitematic 0.10.0
- Boot2Docker ISO 1.10.0
- Virtualbox 5.0.14







Docker Compose - One Service

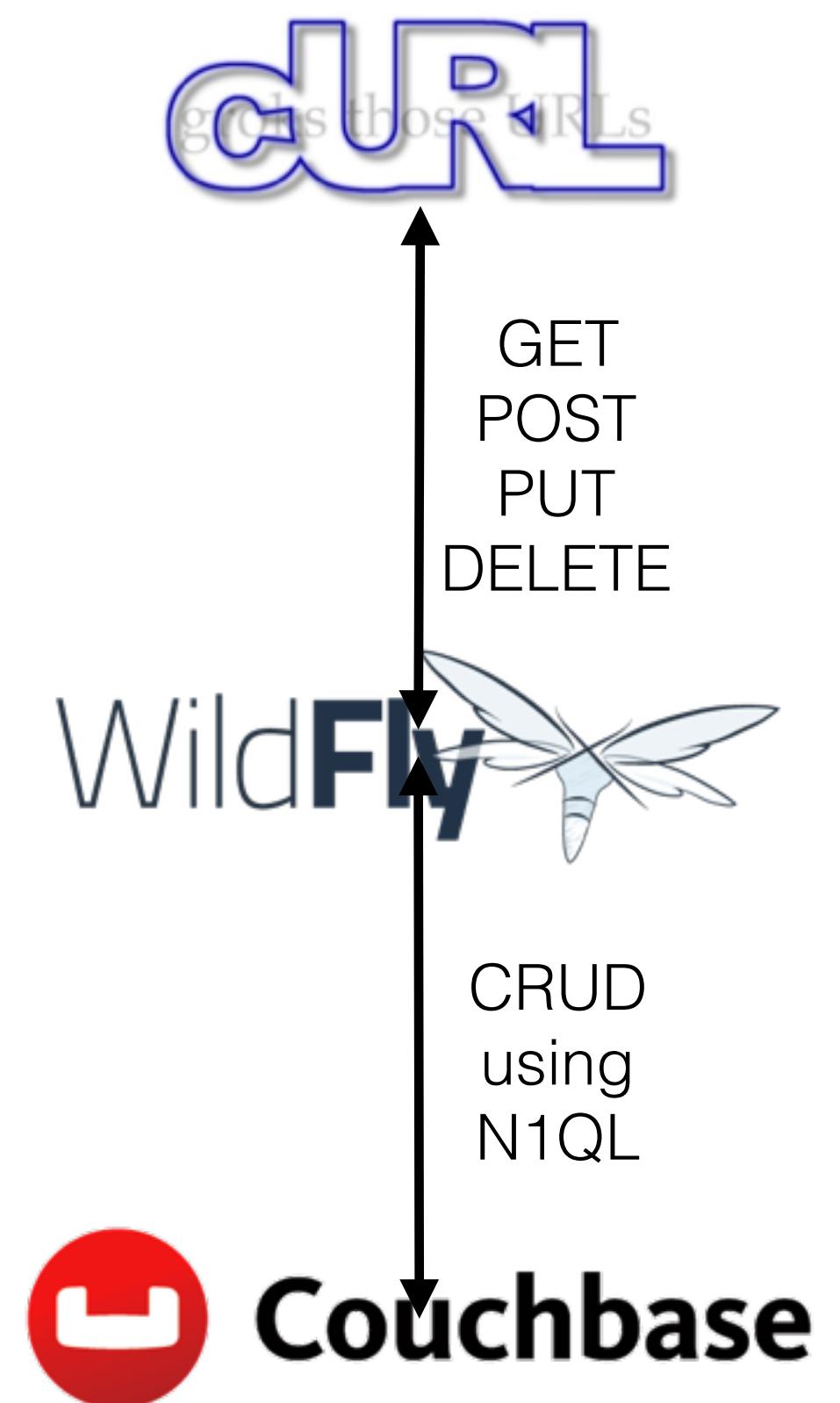
```
mycouchbase:  
  name: mycouchbase  
  image: couchbase/server  
  volumes:  
    - ~/couchbase:/opt/couchbase/var  
  ports:  
    - 8091:8091  
    - 8092:8092  
    - 8093:8093  
    - 11210:11210
```



```
docker run  
--name mycouchbase  
-p 8091:8091  
-p 8093:8093  
-p 11210:11210  
-v ~/couchbase:/opt/couchbase/var  
-d  
couchbase
```

```
docker run  
--name mywildfly  
--link mycouchbase:cb  
-p 8080:8080  
-d  
arungupta/wildfly-admin
```

Docker Compose - Two Services





Docker Compose

- Defining and running multi-container applications
- Configuration defined in one or more files
 - `docker-compose.yml` (default)
 - `docker-compose.override.yml` (default)
 - Multiple files specified using `-f`
 - All paths relative to base configuration file
- Great for dev, staging, and CI

Docker Compose - Two Services

```
mycouchbase:  
  image: couchbase/server  
  ports:  
    - 8091:8091  
    - 8092:8092  
    - 8093:8093  
    - 11210:11210  
  
mywildfly:  
  image: arungupta/wildfly-admin  
  links:  
    - mycouchbase:cb  
  ports:  
    - 8080:8080  
    - 9990:9990
```



Docker Compose - Links

```
CouchbaseCluster.create(System.getenv("DB_PORT_8091_TCP_ADDR"));
```

Docker Compose - Couchbase Cluster

```
couchbase1:  
  image: couchbase/server  
  volumes:  
    - ~/couchbase/node1:/opt/couchbase/var  
couchbase2:  
  image: couchbase/server  
  volumes:  
    - ~/couchbase/node2:/opt/couchbase/var  
couchbase3:  
  image: couchbase/server  
  volumes:  
    - ~/couchbase/node3:/opt/couchbase/var  
  ports:  
    - 8091:8091  
    - 8092:8092  
    - 8093:8093  
    - 11210:11210
```

Docker Machine



Couchbase



Couchbase



Couchbase

Overriding Services in Docker Compose

```
mywildfly:  
  image: jboss/wildfly  
  ports:  
    - 8080:8080
```

docker-compose.yml

```
mywildfly:  
  ports:  
    - 9080:8080
```

docker-compose.override.yml

docker-compose up -d

Dev/Prod with Compose

```
mycouchbase:  
  container_name: "db-dev"  
  image: couchbase/server  
  ports:  
    - . . .  
mywildfly:  
  image: arungupta/wildfly-couchbase-javee7  
  environment:  
    - COUCHBASE_URI=db-dev:8093  
  ports:  
    - 8080:8080
```

docker-compose.yml

docker-compose up -d

```
mywildfly:  
  environment:  
    - COUCHBASE_URI=db-prod:8093  
  ports:  
    - 80:8080  
mycouchbase:  
  container_name: "db-prod"
```

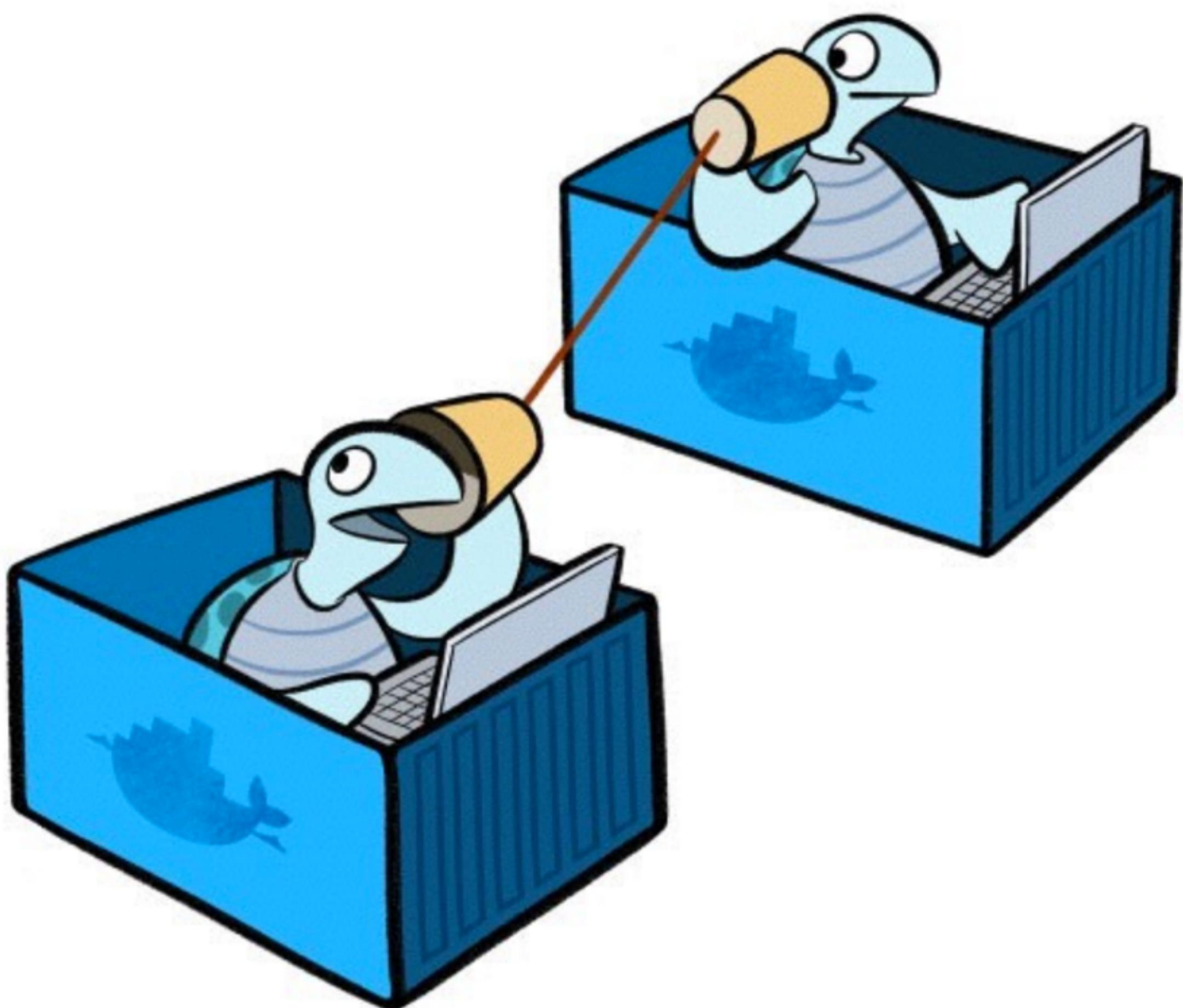
production.yml

docker-compose up
-f docker-compose.yml
-f production.yml
-d

Compose in Production

- Primarily aimed at development and testing environments
- May be for smaller deployments
- Not suitable for larger deployments

Multi-host Networking



Default Networks

```
docker network ls
```

NETWORK ID	NAME	DRIVER
8cf651cafbef	bridge	bridge
14e63204639e	none	null
96901337c96f	host	host

Network Name	Purpose
bridge	Default network that containers connect to
none	Container-specific networking stack
host	Adds container on hosts networking stack

Multi-host Networking

- Create virtual networks and attach containers
- **Bridge** network spans single host
- **Overlay** network spans multiple hosts
 - Uses `libnetwork` (built-in VXLAN-based overlay network driver) and Docker's `libkv`
- Works with Swarm and Compose
 - ~~Experimental in Compose~~
- Pluggable: Calico, Cisco, Weave, . . .

Networking vs Links

- Connect containers to each other
- Across different physical or virtual hosts
- Containers can be easily stopped, started and restarted
 - Connection to other containers not disturbed
- Can be created in any order

```
mycouchbase:  
  image: couchbase/server  
  ports:  
    - 8091:8091  
    - 8092:8092  
    - 8093:8093  
    - 11210:11210  
  
mywildfly:  
  image: arungupta/wildfly-admin  
  links:  
    - mycouchbase:cb  
  ports:  
    - 8080:8080  
    - 9990:9990
```

```
mycouchbase:  
  container_name: "db"  
  image: couchbase/server  
  ports:  
    - 8091:8091  
    - 8092:8092  
    - 8093:8093  
    - 11210:11210  
  
mywildfly:  
  image: arungupta/wildfly-admin  
  environment:  
    - COUCHBASE_URI=db  
  ports:  
    - 8080:8080  
    - 9990:9990
```

Links

Networking

Networking - Application Code

```
CouchbaseCluster.create(System.getenv("DB_PORT_8091_TCP_ADDR"));
```

Links

```
CouchbaseCluster.create(System.getenv("COUCHBASE_URI"));
```

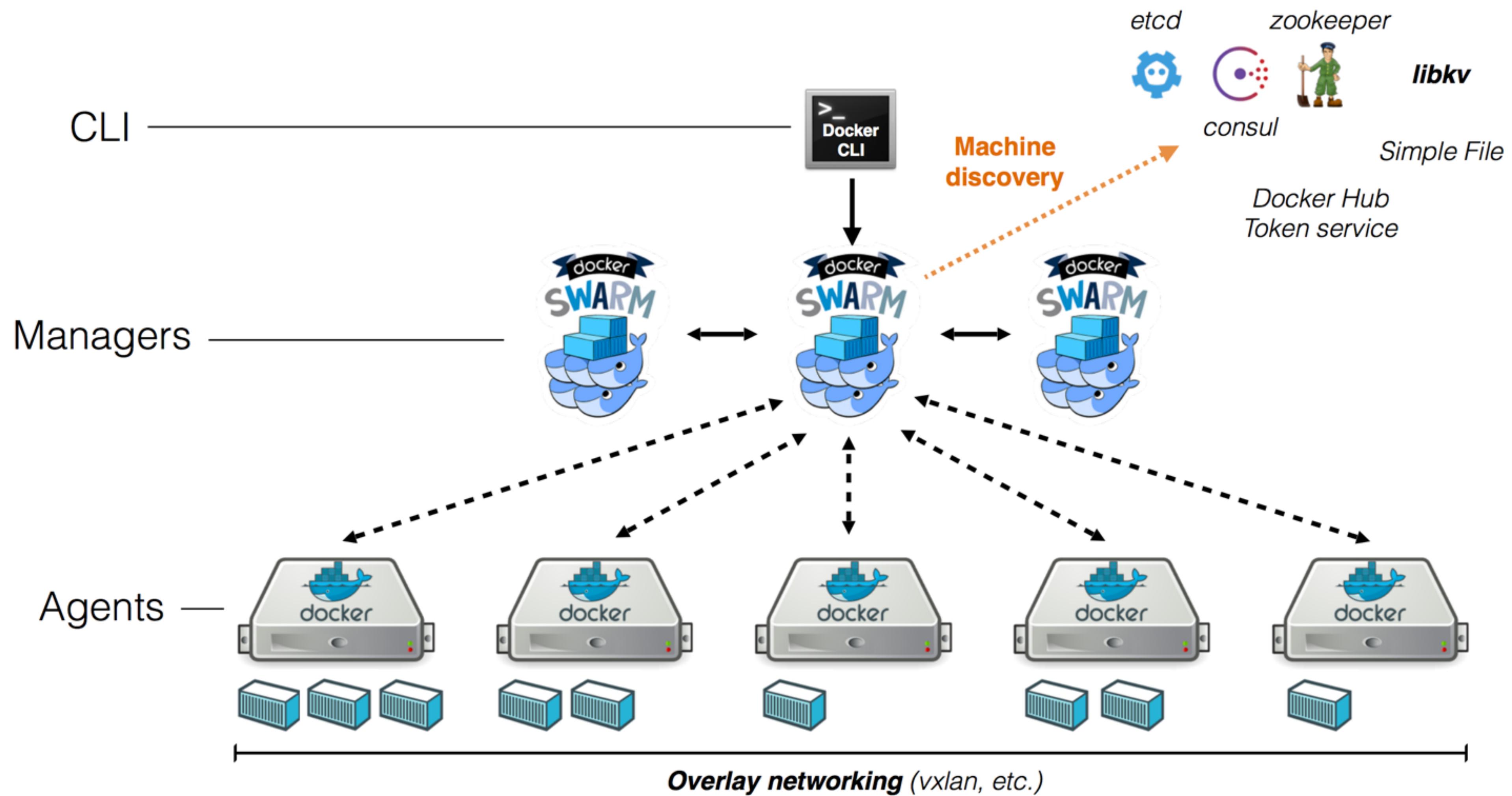
Networking



Docker Swarm

- Native clustering for Docker
- Provides a unified interface to a pool of Docker hosts
- Fully integrated with Machine and Compose
- Serves the standard Docker API
- 1.1 - Ready for production
 - Reschedule containers when a node fails (experimental)
 - Better node management

Stress tested on 1000 EC2 nodes, ~30k containers





Master High Availability

- Handle the failover of a manager instance
- **Primary** manager, multiple **replica** instances
- Requests to replica are proxied to primary
- `docker run swarm manage`
 - `--replication`: Enable Swarm manager replication
 - `--replication-ttl "30s"`: Leader lock release time on failure
 - `--advertise`, `--addr`: Address of the swarm manager joining the cluster



Scheduling Backends

- Based on CPU (**-c**), RAM (**-m**), number of containers
- **docker machine create --strategy <value>**
 - **spread** (default): node with least number of running containers
 - **binpack**: node with most number of running containers
 - **random**: mostly for debugging
- API for pluggable backends (e.g. Mesos) coming

Limiting CPU resources

```
docker run --help | grep cpu  
--cpu-shares=0  
--cpu-period=0  
--cpu-quota=0  
--cpuset-cpus=  
--cpuset-mems=
```

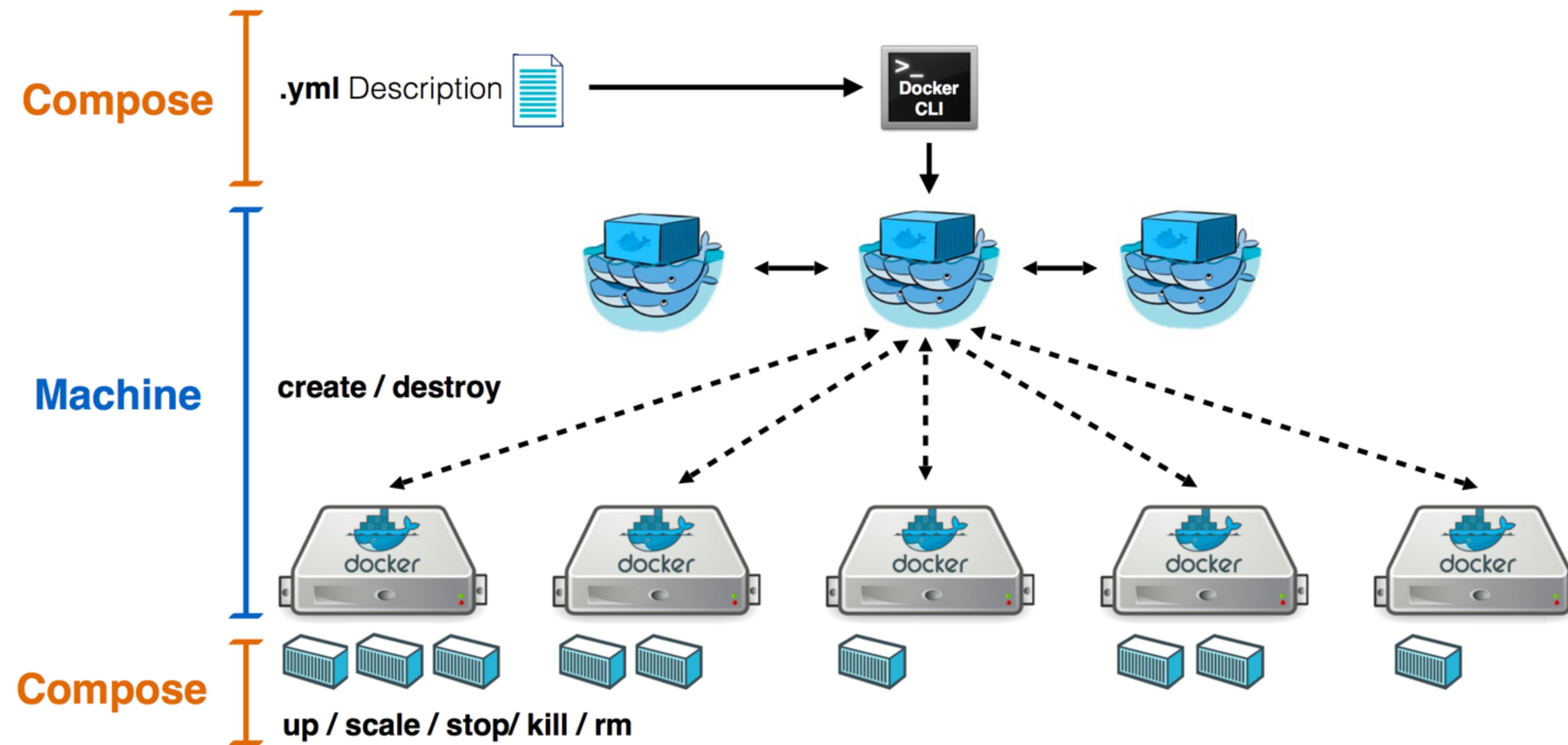
CPU shares (relative weight)
Limit CPU CFS (Completely Fair Scheduler) period
Limit CPU CFS (Completely Fair Scheduler) quota
CPUs in which to allow execution (0-3, 0,1)
MEMs in which to allow execution (0-3, 0,1)

-c=1024

-c=512

-c=512

Machine + Swarm + Compose



Persistent Storage

- Data volumes - used to persist data independent of container's lifecycle
- Multiple plugins: Flocker, Ceph, . . .

```
docker volume --help

Usage: docker volume [OPTIONS] [COMMAND]

Manage Docker volumes

Commands:
  create           Create a volume
  inspect          Return low-level information on a volume
  ls               List volumes
  rm              Remove a volume
```

Persistent Storage

Create a volume

```
docker volume create --name=data data
```

Run a container with the volume

```
docker run -it -v data:/opt/couchbase/var couchbase
```

docker network

```
docker network --help
```

Usage: docker network [OPTIONS] COMMAND [OPTIONS]

Commands:

disconnect

Disconnect container from a network

inspect

Display detailed network information

ls

List all networks

rm

Remove a network

create

Create a network

connect

Connect container to a network

Docker Compose and Networking

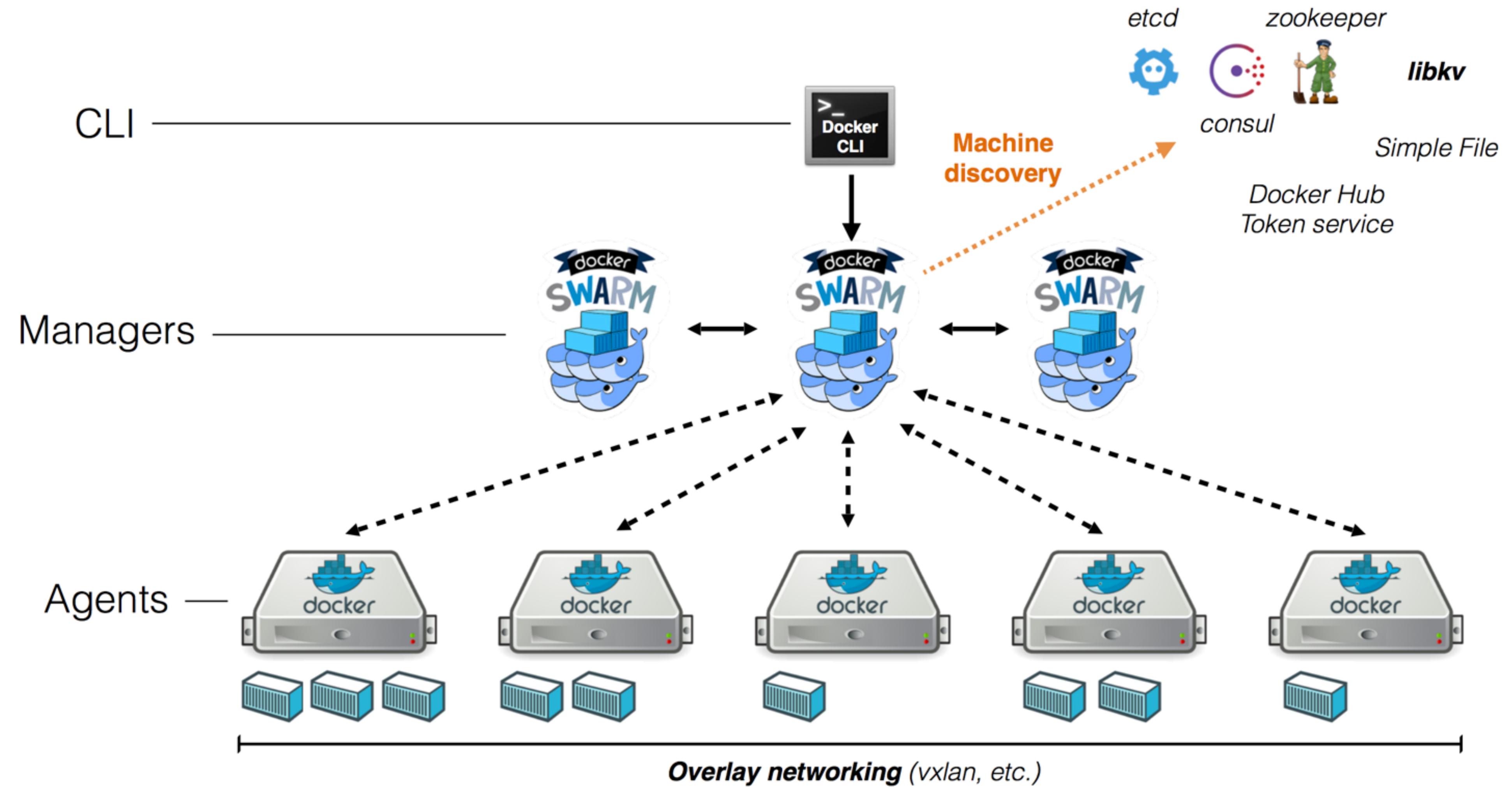
- `docker network ls`

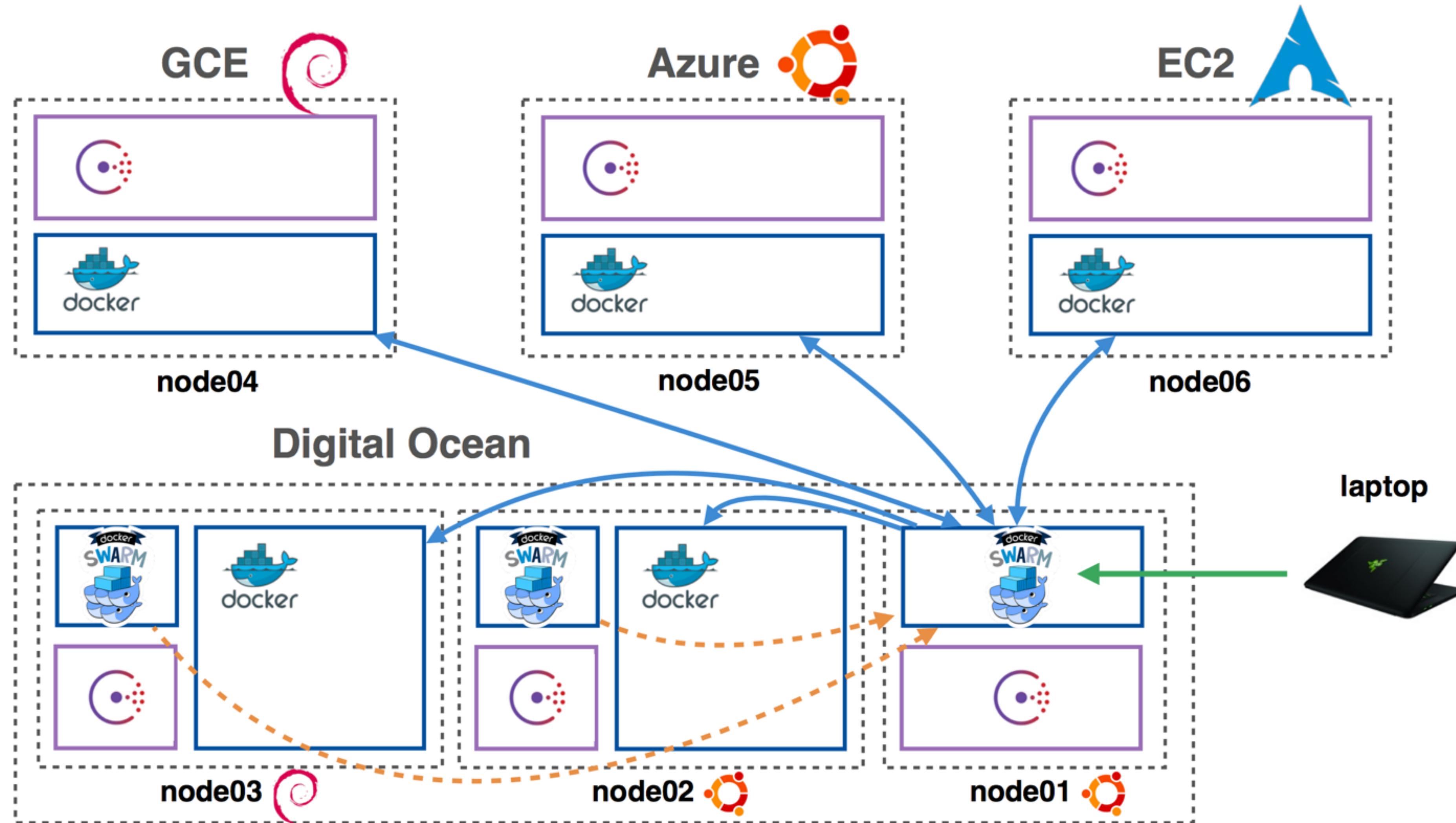
NETWORK ID	NAME	DRIVER
1d4d4152b99a	bridge	bridge
8396171e6bbc	none	null
8b50dda5fe61	host	host

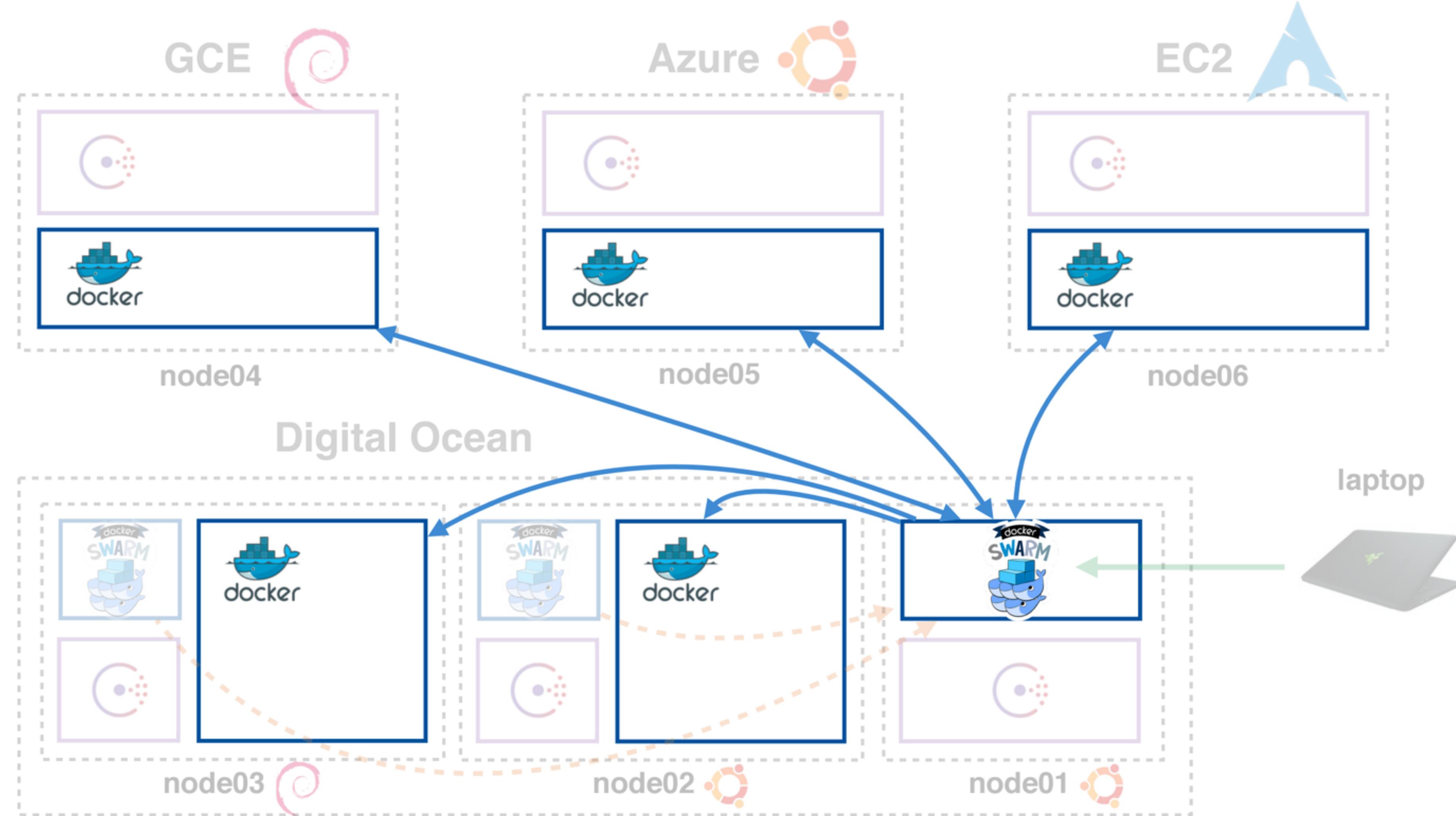
- `docker-compose up -d`

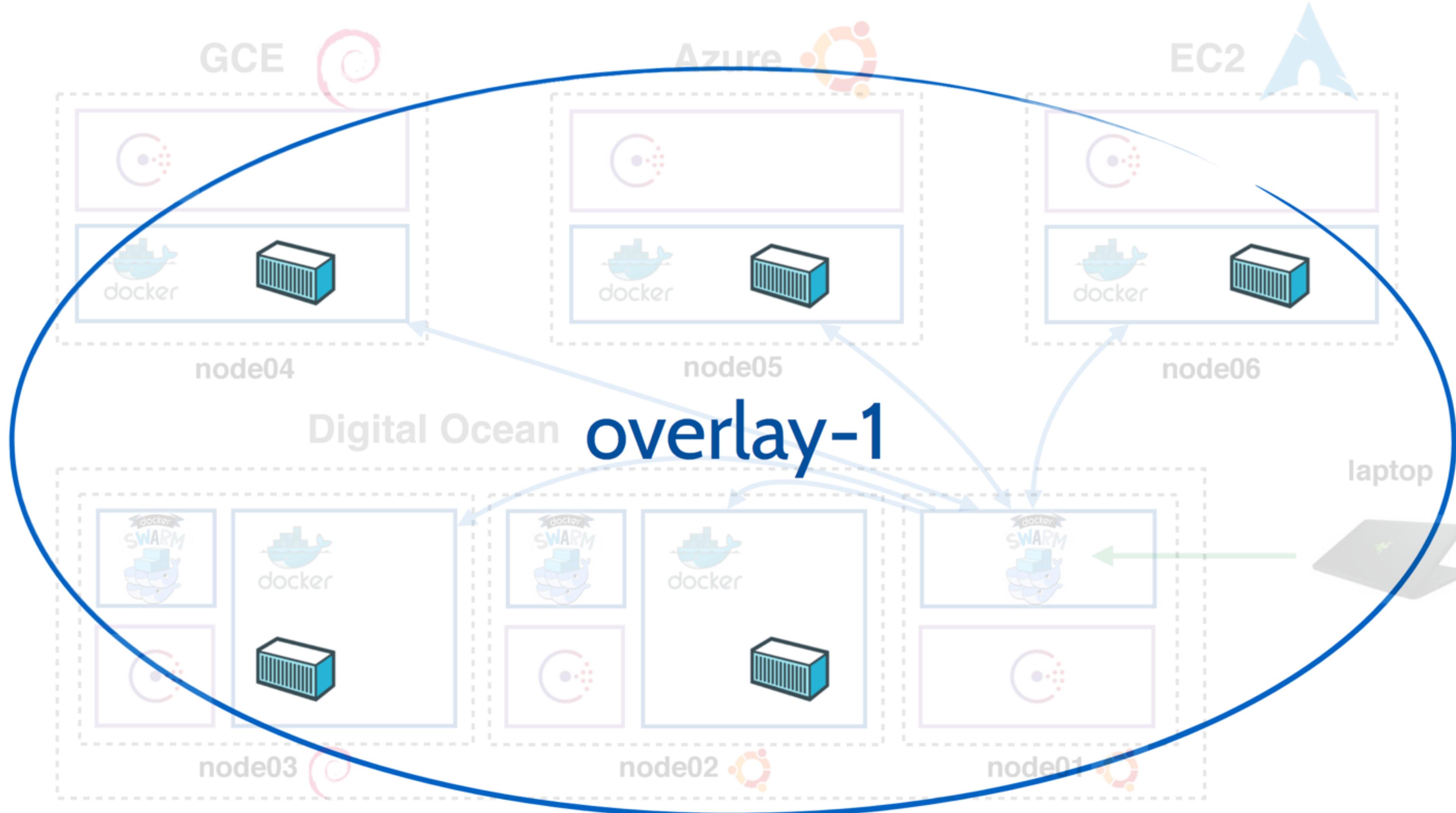
- `docker network ls`

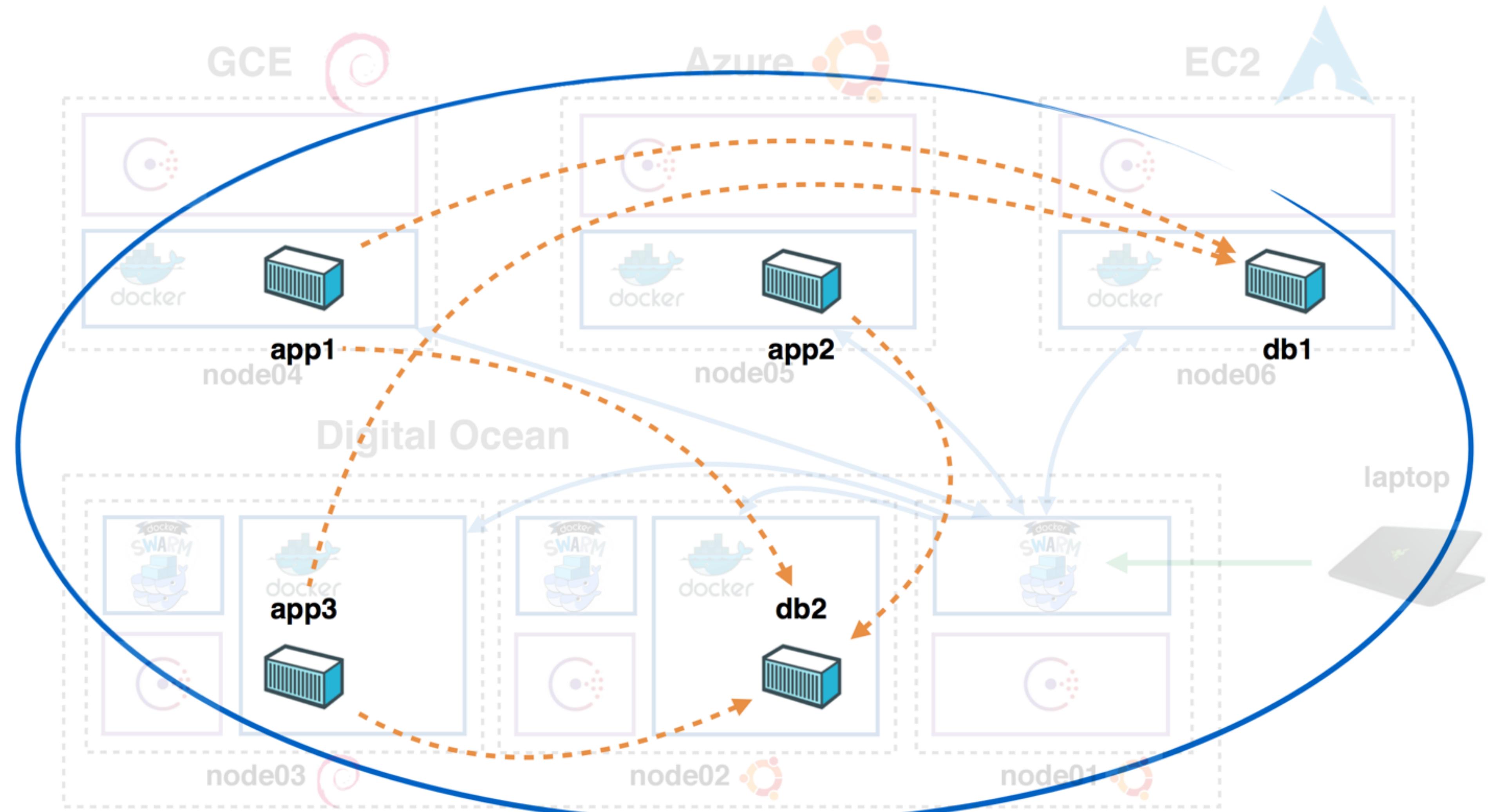
NETWORK ID	NAME	DRIVER
1d4d4152b99a	bridge	bridge
8396171e6bbc	none	null
8b50dda5fe61	host	host
b1770955e5aa	wildflycouchbasejavaee7	bridge

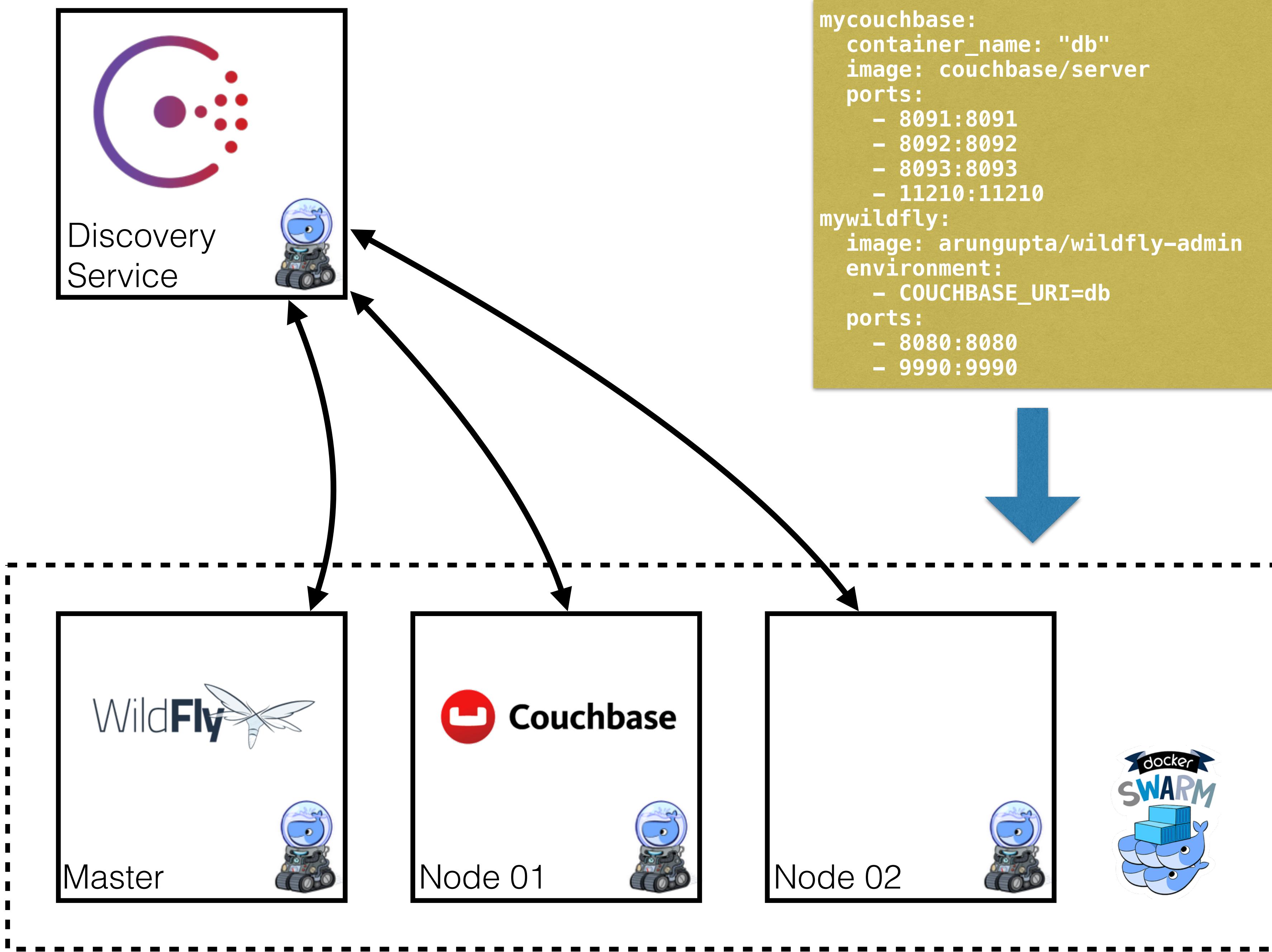












Docker TUTUM

- SaaS
- Build your images
- Deploy and manage across different clouds
 - Amazon, Digital Ocean, Microsoft, Azure, IBM SoftLayer
 - BYON



Docker Tutum



Build

Containerize your applications and accelerate development.



Deploy

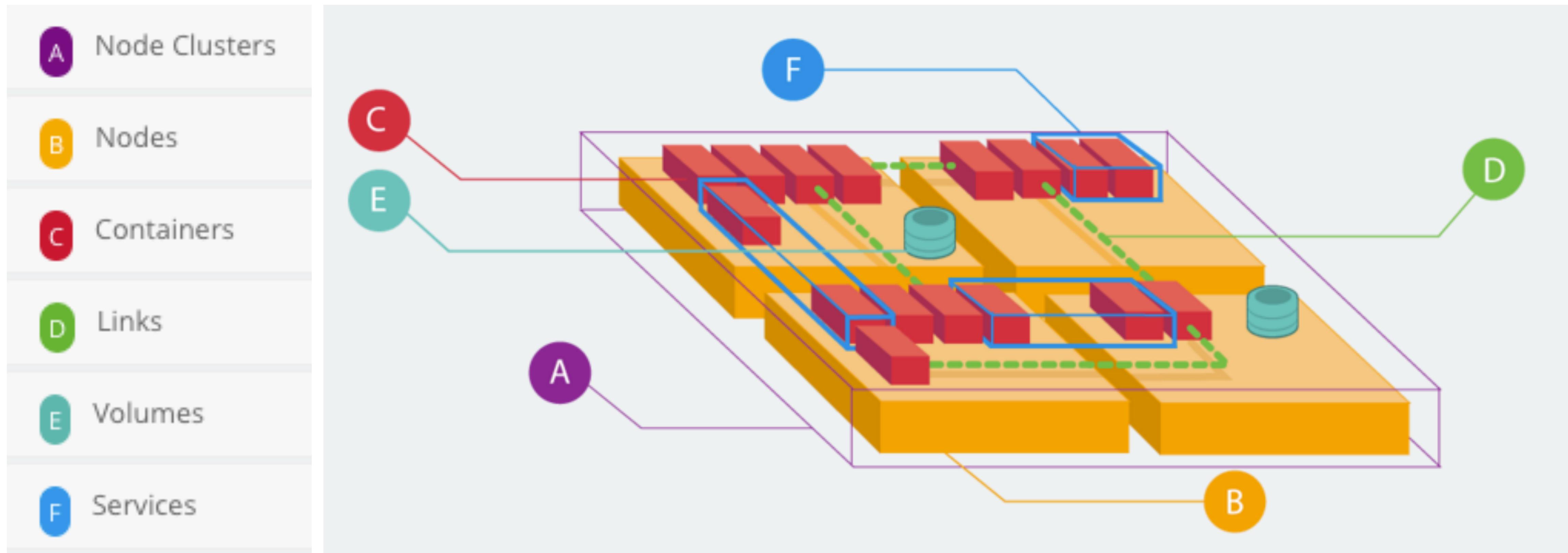
All frameworks and technologies welcomed.
Scale with ease on any Cloud.



Manage

Simplify operations, focus on your code, and forget about managing servers.

Docker Tutmum Architecture





The screenshot shows the Docker Compose interface for creating a service named 'my-couchbase-node-cluster'. The service configuration includes:

- Deploy tags:** couchbase
- Provider:** Amazon Web Services
- Region:** US West (N. California)
- VPC:** Auto
- Subnet:** Auto

Image selection: arungupta/couchbase-node (latest)

Service configuration: couchbase-node-fac95d8b

Environment variables: Vol

Logs: The logs show the container starting and listening on ports 8091, 8092, and 8093, which are mapped to node ports 8091, 8092, and 8093 respectively.

Container port	Protocol	Published	Node port
8091	tcp	✓	8091
8092	tcp	✓	8092
8093	tcp	✓	8093
11207	tcp	✗	--
11210	tcp	✓	11210



Docker Registry

- Store and distribute Docker images
 - Control where images are stored
 - Own image distribution pipeline
 - Integrate image storage/distribution in dev workflow
- Docker Hub
 - Free-to-use and hosted
- Docker Trusted Registry
 - Commercially supported
 - RBAC, LDAP/AD integration, updates, etc

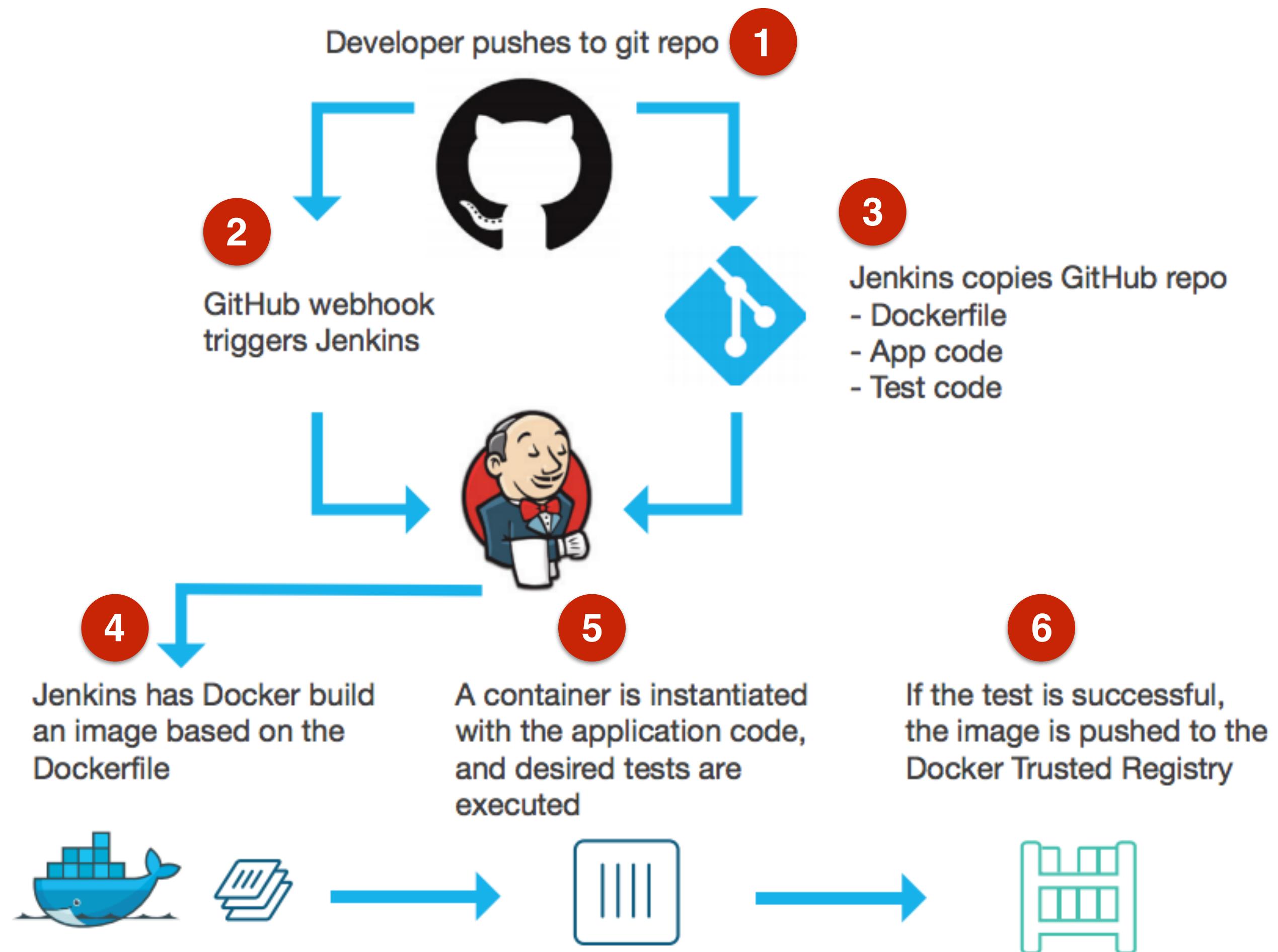


Registry Primary Usage Scenario

- CI/CD with Docker
 - Centrally located base images
 - Store individual build images
 - Pull tested images to production

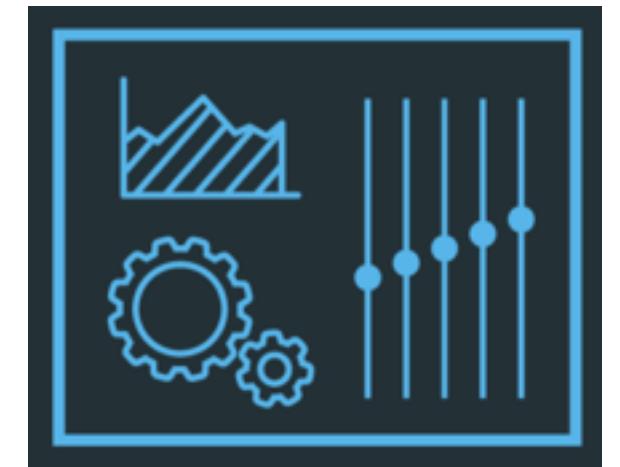
CI/CD with Docker + Jenkins

1. Push a commit to GitHub
2. GitHub webhook
3. Jenkins copies artifacts
4. Jenkins builds Docker image
5. Runs test on Docker container
6. Pushes to DTR



Monitoring Docker Containers

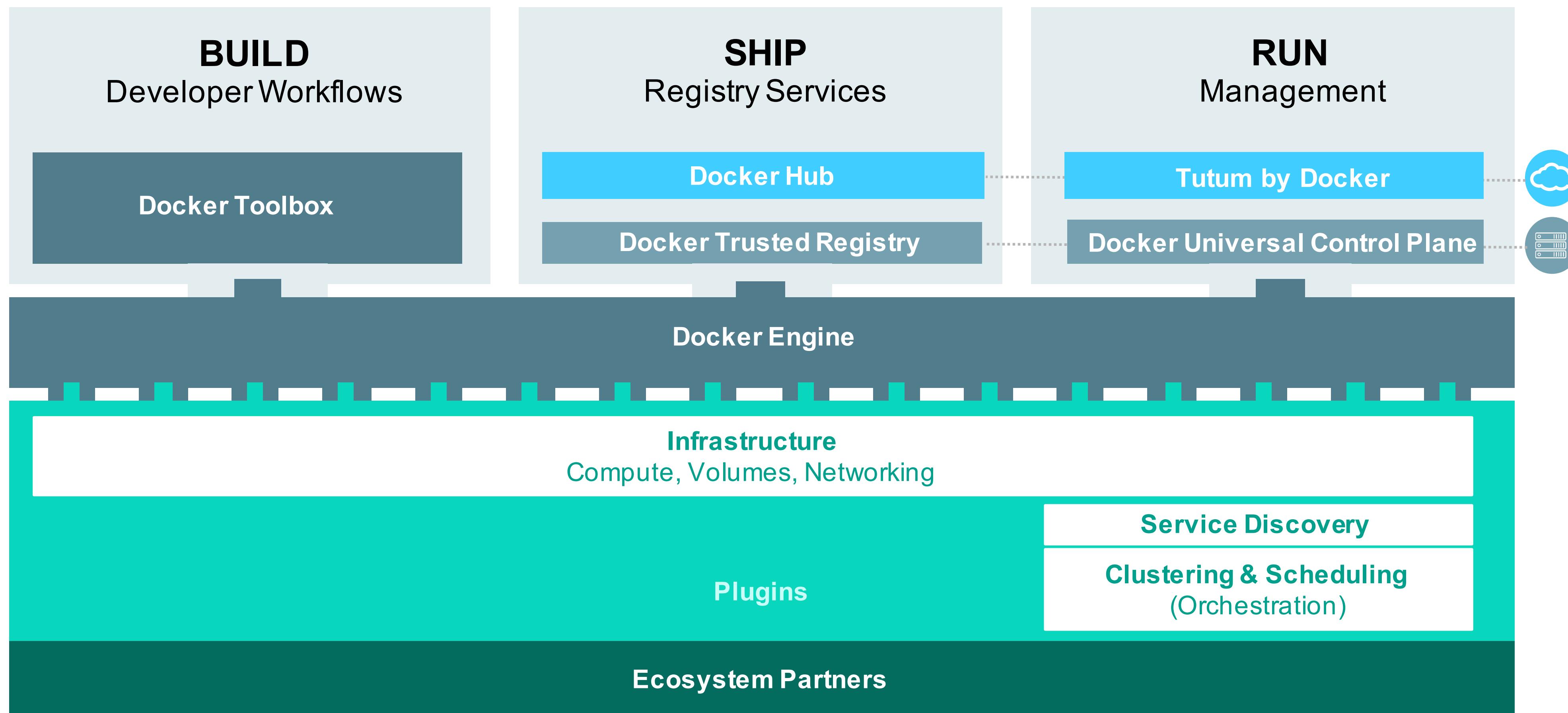
- `docker stats` command
 - LogEntries
- Docker Remote API: `/container/{container-name|cid}/stats`
- Docker Universal Control Plane
- cAdvisor
 - Prometheus
 - InfluxDB



cAdvisor



Docker Mission



References

- github.com/javaee-samples/docker-java
- docs.docker.com