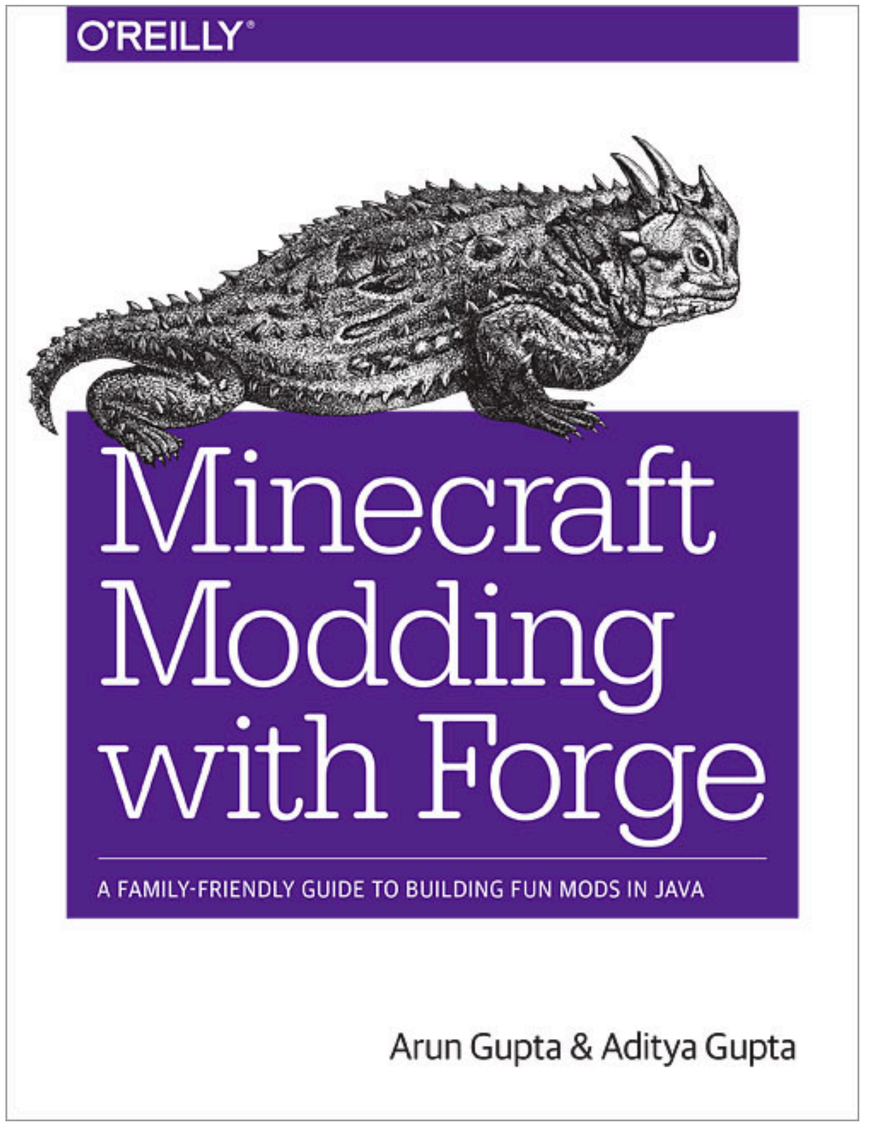
Getting Started with

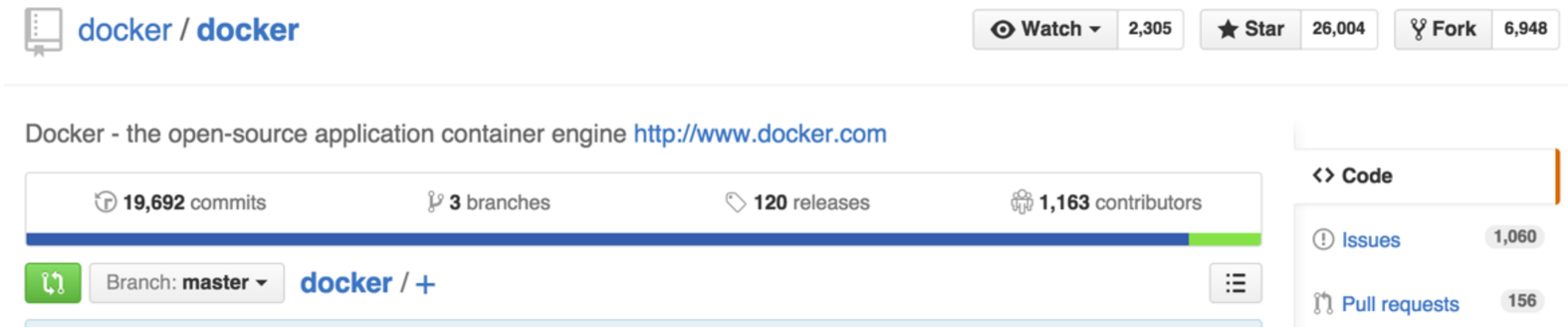
Arun Gupta
Vice President, Developer Advocacy
Couchbase
@arungupta, blog.arungupta.me
arun@couchbase.com

DEVOXX™ 4KIDS USA

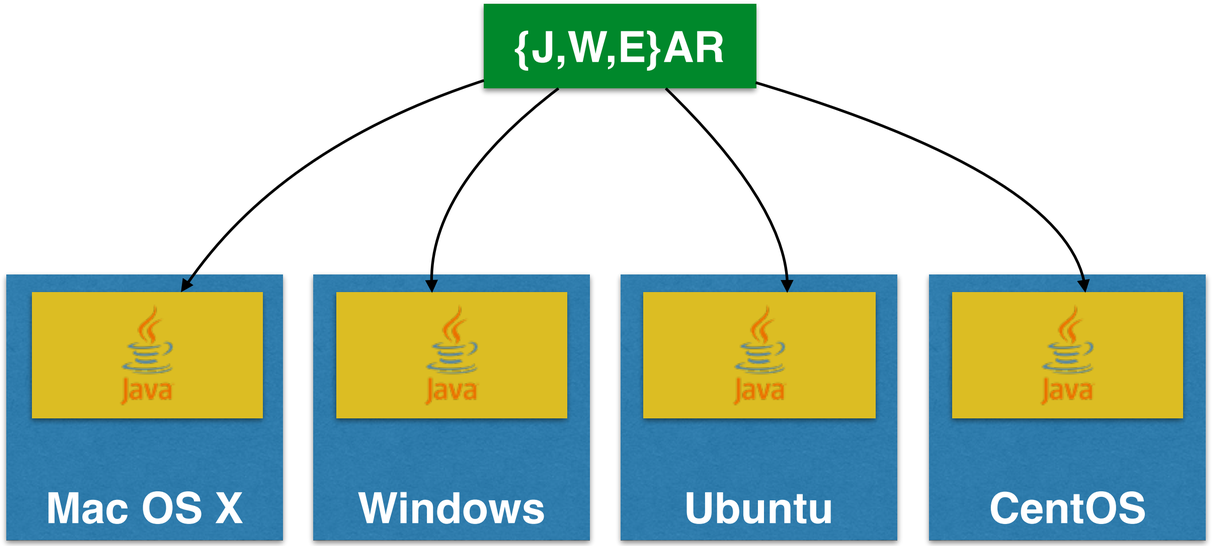


What is Docker?

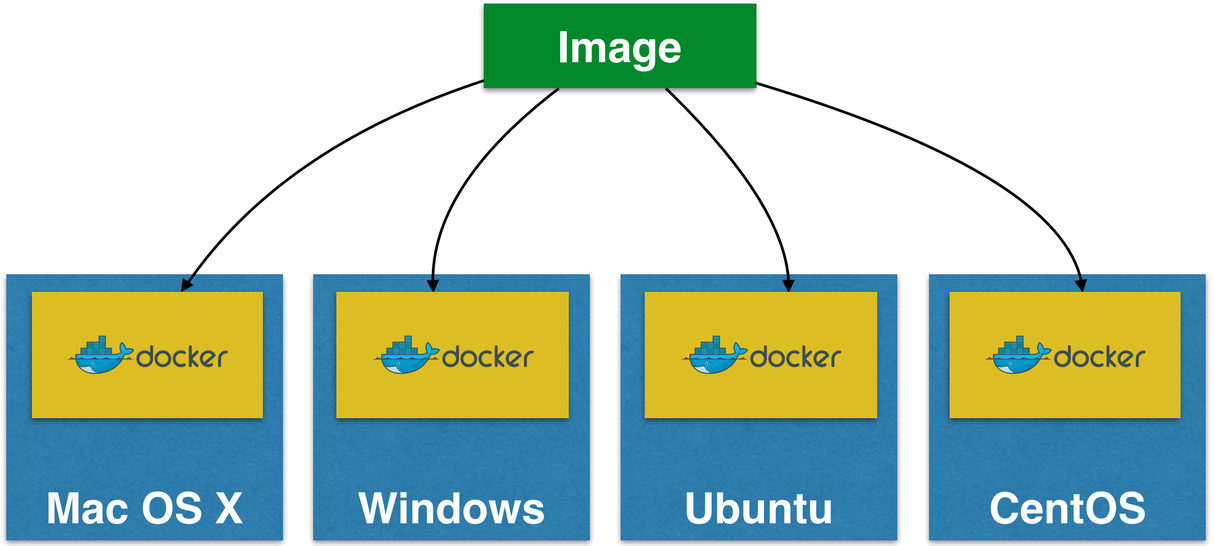
- Open source project and company



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



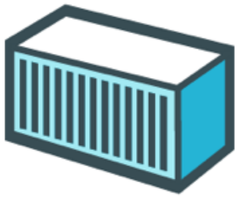
WORA = Write Once Run Anywhere



PODA = Package Once Deploy Anywhere

Build

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

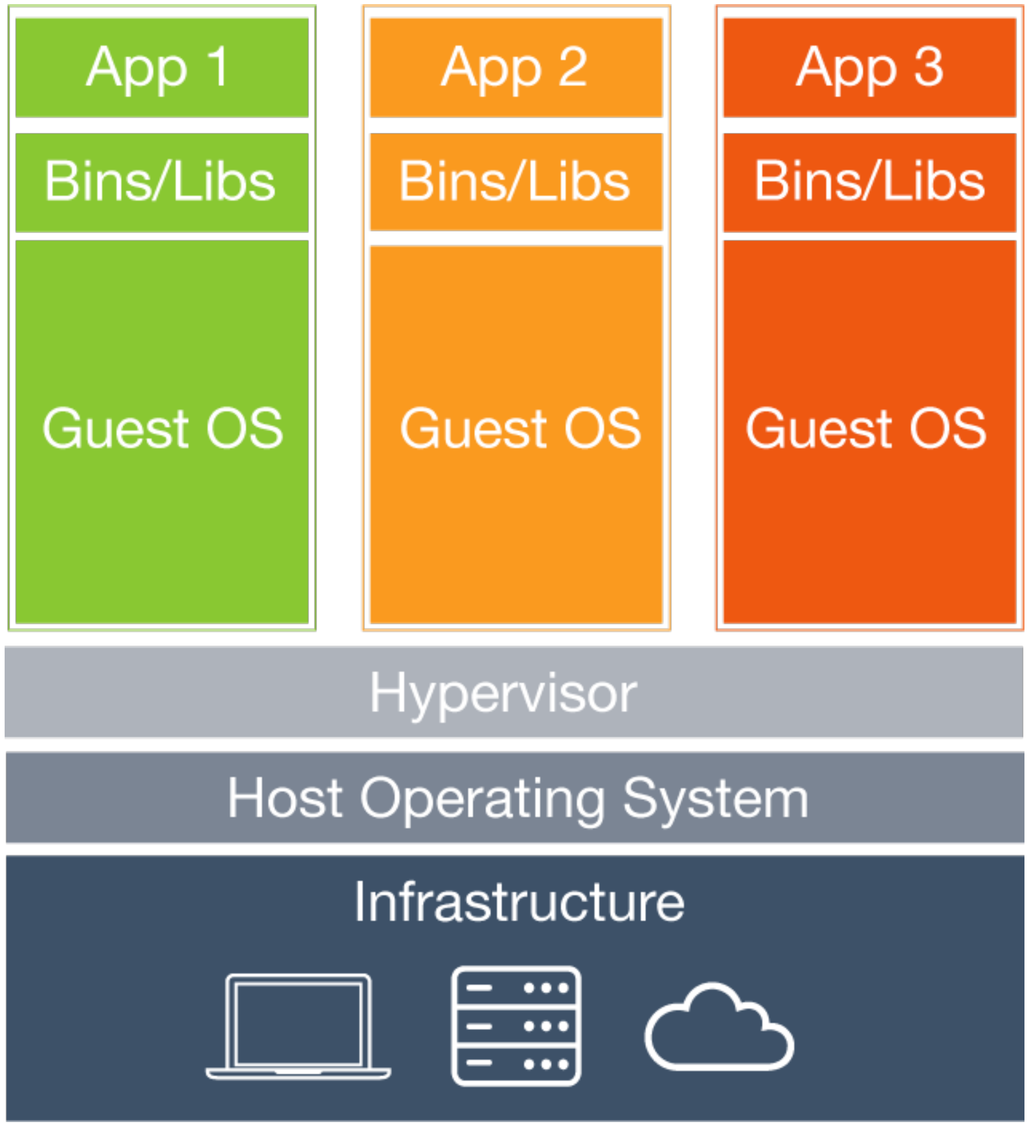


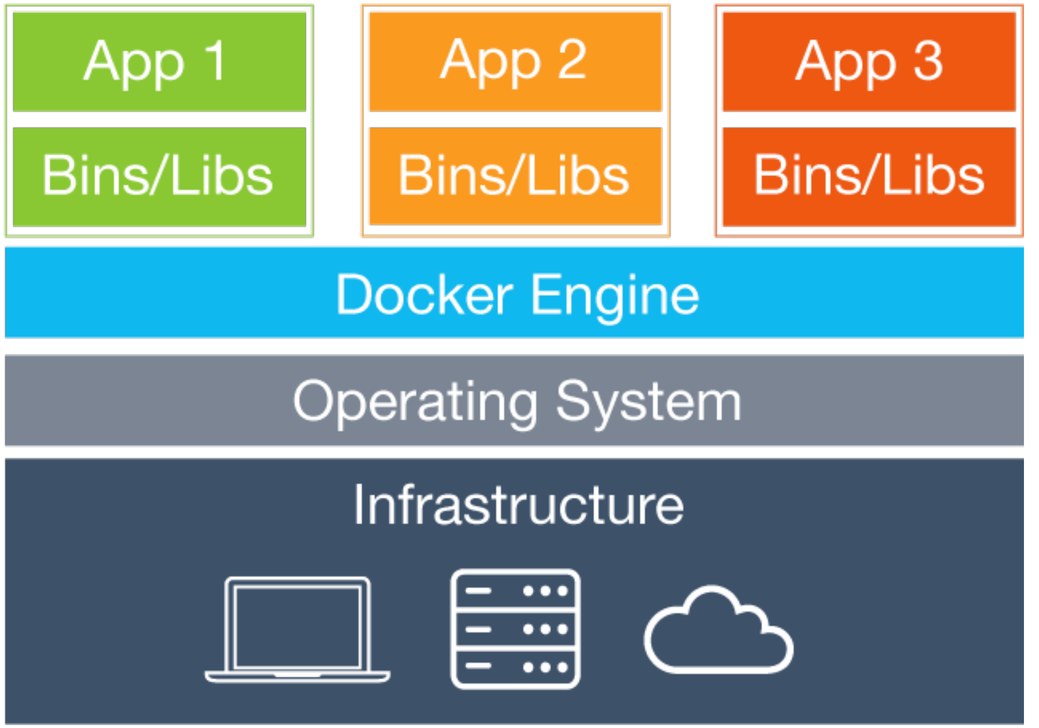
Ship

Ship the “Dockerized” app and dependencies anywhere - to QA, teammates, or the cloud - without breaking anything.

Run

Scale to 1000s of nodes, move between data centers and clouds, update with zero downtime and more.


App 1
Bins/Libs
Guest OS
App 2
Bins/Libs
Guest OS
App 3
Bins/Libs
Guest OS
Hypervisor
Host Operating System
Infrastructure


App 1
Bins/Libs
App 2
Bins/Libs
App 3
Bins/Libs
Docker Engine
Operating System
Infrastructure

Build

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

- Image defined in text-based **Dockerfile**
- List of commands to build the image

```
FROM fedora:latest

CMD echo “Hello world”
```

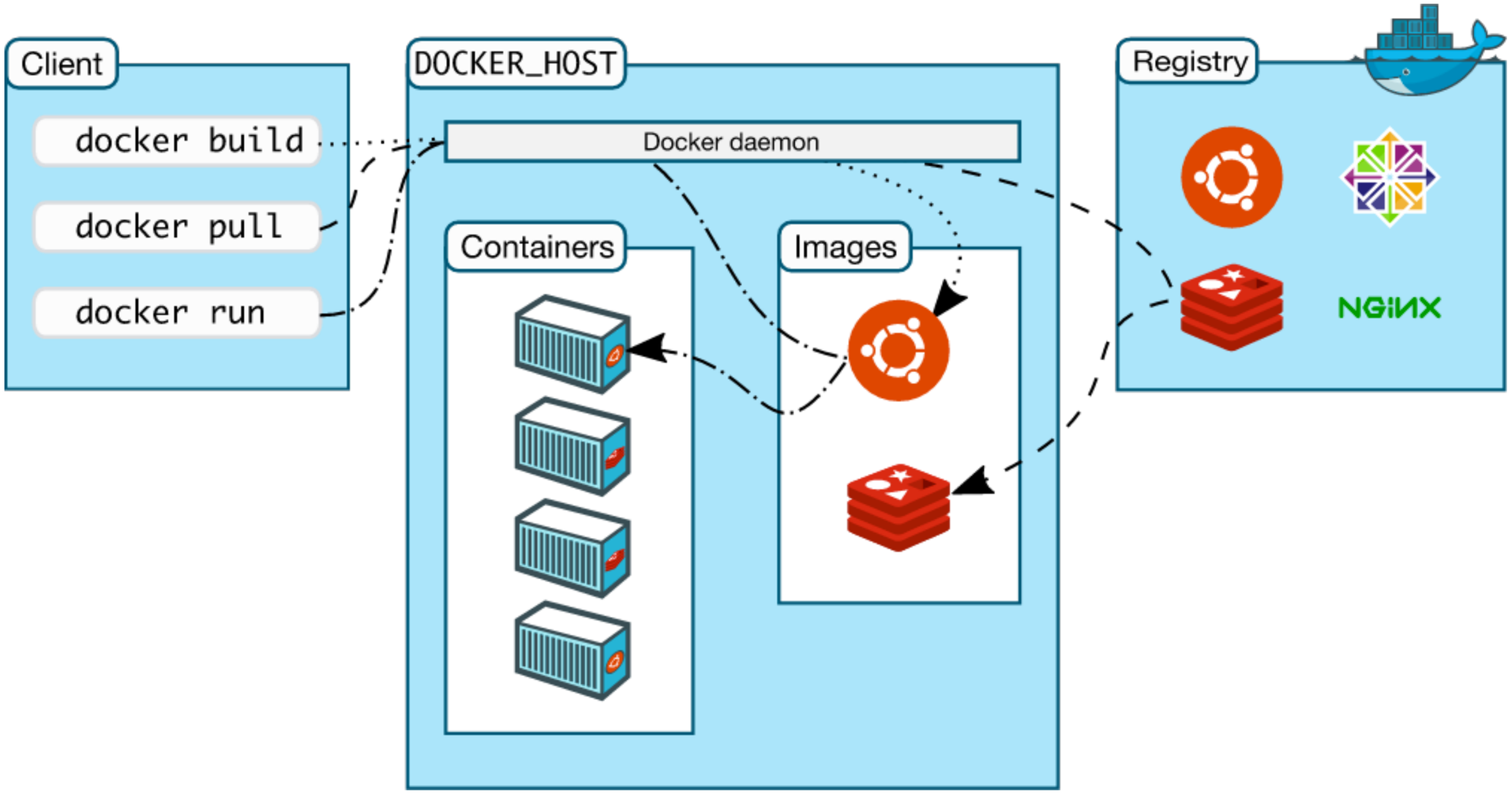
```
FROM jboss/wildfly

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
```

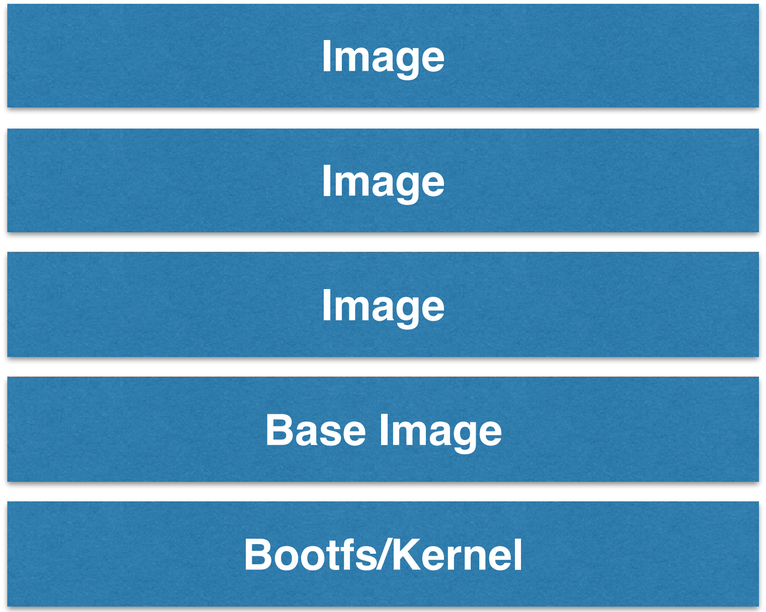
- Dockerfile reference
  - Usage
  - Format
    - Environment replacement
    - .dockerignore file
  - FROM
  - MAINTAINER
  - RUN
    - Known issues (RUN)
  - CMD
  - LABEL
  - EXPOSE
  - ENV
  - ADD
  - COPY
  - ENTRYPOINT
    - Exec form ENTRYPOINT example
    - Shell form ENTRYPOINT example
  - VOLUME
  - USER
  - WORKDIR
  - ARG
  - ONBUILD
  - STOPSIGNAL
  - Dockerfile examples

http://docs.docker.com/engine/reference/builder/

Docker Workflow



Union File System



jboss/wildfly

jboss/base-jdk:8

jboss/base

centos:7

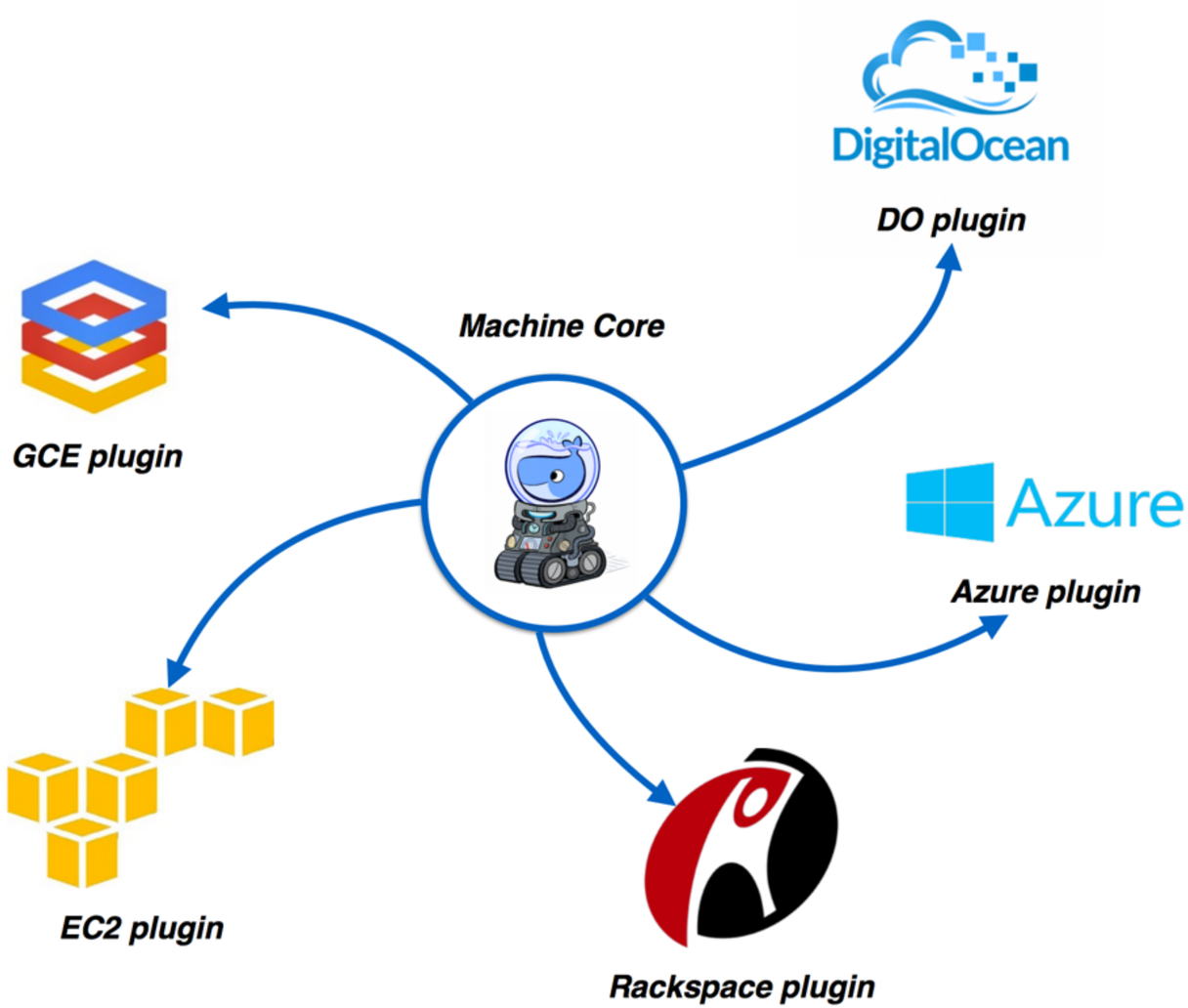
Docker Machine

- 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 1.9.0
- Docker Machine 0.5.0
- Docker Compose 1.5.0 ~~(Mac only)~~
- Docker Kitematic 0.9.3
- Boot2Docker ISO 1.9.0
- Virtualbox 5.0.8

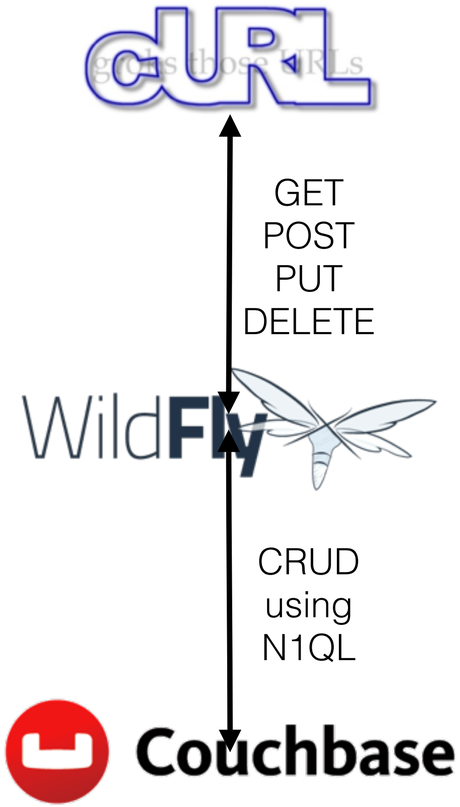

docker

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 Compose - Two Services

```
mycouchbase:
 image: couchbase/server
  ports:
    - 8091:8091
    - 8092:8092
    - 8093:8093
    - 11210:11210
mywildfly:
  image: jboss/wildfly
  environment:
    - COUCHBASE_URI=192.168.99.100
  ports:
    - 8080:8080
```



```
docker run
--name mysqldb
-e MYSQL_USER=mysql
-e MYSQL_PASSWORD=mysql
-e MYSQL_DATABASE=sample
-e MYSQL_ROOT_PASSWORD=supersecret
-d
mysql
```

```
docker run
--name mywildfly
--link mysqldb:db
-p 8080:8080
-d
arungupta/wildfly-mysql-javaee7
```

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.yml

```
mysqldb:
  image: mysql
  environment:
    MYSQL_DATABASE: sample
    MYSQL_USER: mysql
    MYSQL_PASSWORD: mysql
    MYSQL_ROOT_PASSWORD: supersecret
mywildfly:
  image: arungupta/wildfly-mysql-javaee7
  links:
    - mysqldb:db
```

Links - JDBC Connection

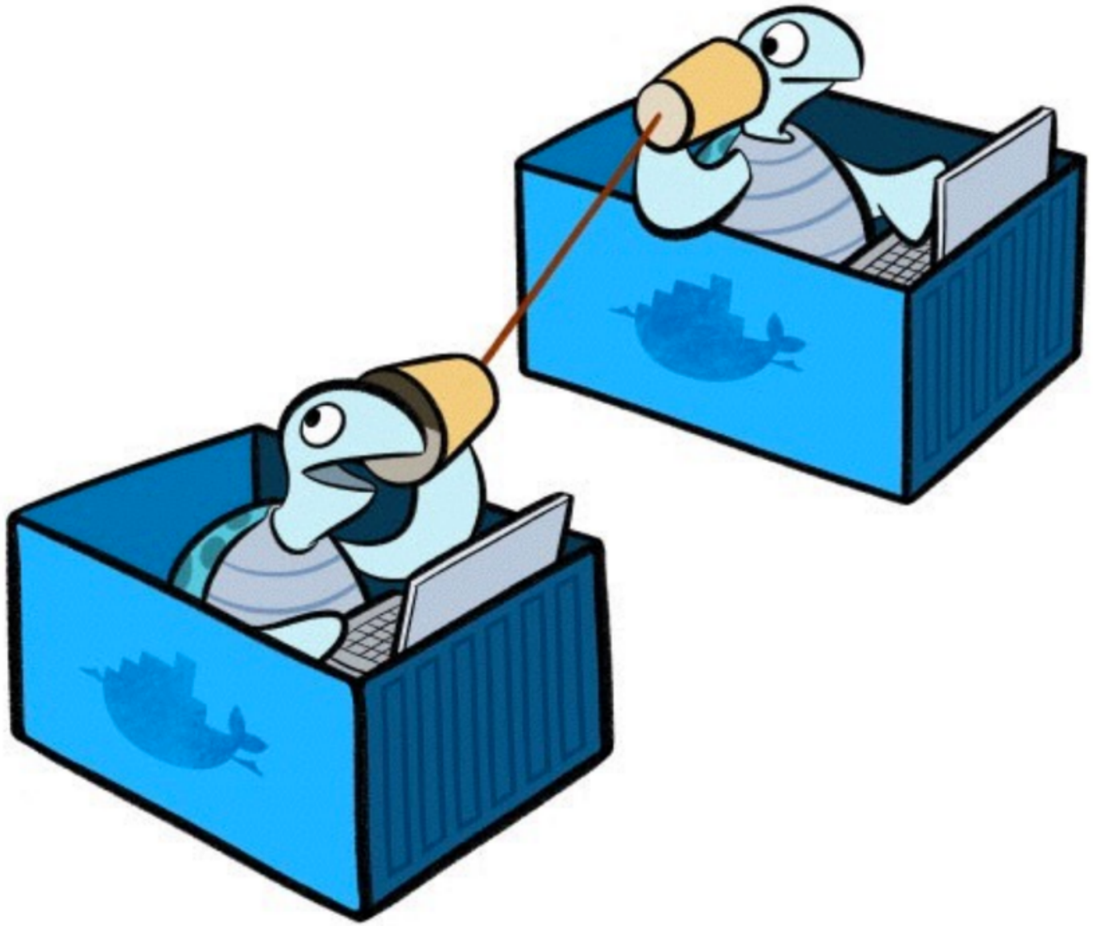
```
data-source add
--name=mysqlDS
--driver-name=mysql
--jndi-name=java:jboss/datasources/
ExampleMySQLDS
--connection-url=jdbc:mysql://
$DB_PORT_3306_TCP_ADDR:
$DB_PORT_3306_TCP_PORT/sample?
useUnicode=true&amp;characterEncoding=UT
F-8
--user-name=mysql
--password=mysql
--use-ccm=false
--max-pool-size=25
--blocking-timeout-wait-millis=5000
--enabled=true
```

```
<connection-url>jdbc:mysql://
172.17.0.2:3306/sample?
useUnicode=true&amp;amp;characterEncodin
g=UTF-8</connection-url>
```

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
```

Multi Host Networking



Multi-host Networking

- Create virtual networks and attach containers
  - Software defined networking
- **`Bridge`** network span single host
- **`Overlay`** network spans multiple hosts
- 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 w/o disrupting connection to other containers
- Can be created in any order

Links

```
mysqldb:
  image: mysql
  environment:
    MYSQL_DATABASE: sample
    MYSQL_USER: mysql
    MYSQL_PASSWORD: mysql
    MYSQL_ROOT_PASSWORD: supersecret
mywildfly:
  image: arungupta/wildfly-mysql-javaee7
  links:
    - mysqldb:db
```

Networking

```
mysqldb:
  container_name: "db"
  image: mysql:latest
  environment:
    MYSQL_DATABASE: sample
    MYSQL_USER: mysql
    MYSQL_PASSWORD: mysql
    MYSQL_ROOT_PASSWORD: supersecret
mywildfly:
  image: arungupta/wildfly-mysql-javaee7
  environment:
    - MYSQL_URI=db:3306
  ports:
    - 8080:8080
```

Networking - JDBC Connection

```
data-source add
--name=mysqlDS
--driver-name=mysql
--jndi-name=java:jboss/datasources/
ExampleMySQLDS
--connection-url=jdbc:mysql://$MYSQL_URI/
sample?
useUnicode=true&amp;ch
F-8
--user-name=mysql
--password=mysql
--use-ccm=false
--max-pool-size=25
--blocking-timeout-wait-millis=5000
--enabled=true
```

```
<connection-url>jdbc:mysql://db:3306/
sample?
useUnicode=true&amp;amp;characterEncodin
g=UTF-8</connection-url>
```

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-javaee7
  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

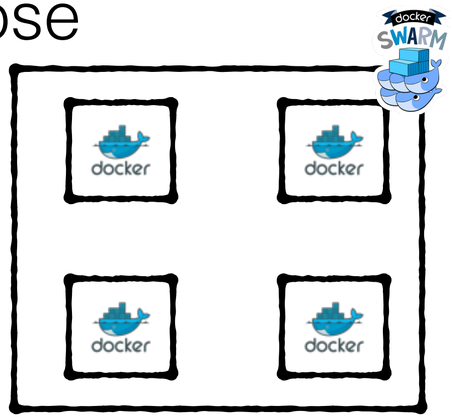
```
docker-compose up
-f docker-compose.yml
-f production.yml
-d
```

Compose in Production

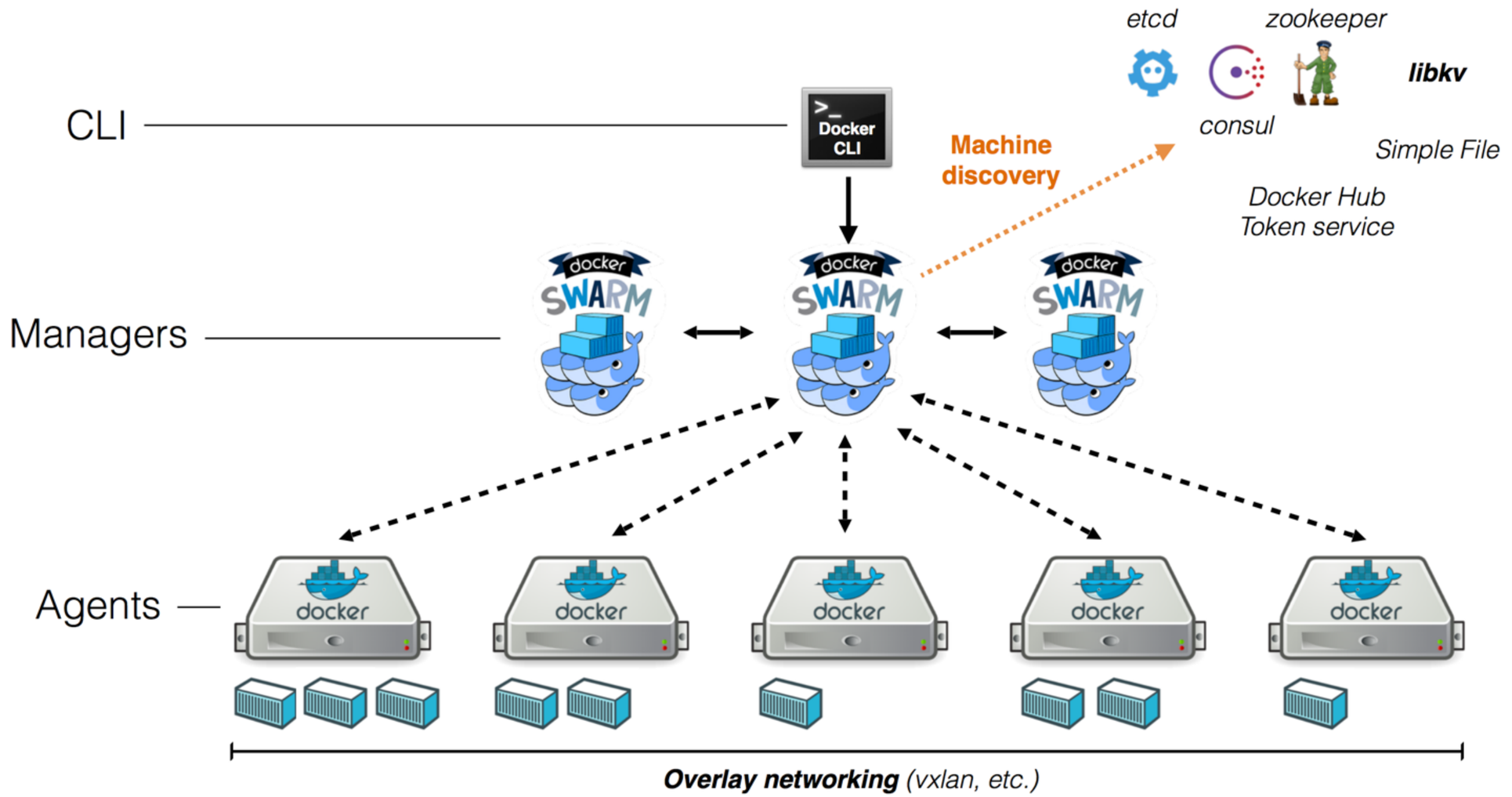
- Primarily aimed at development and testing environments
- May be for smaller deployments
- Not suitable for larger deployments

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
- 1.0.0 - Ready for production?



Stress tested on 1000 EC2 nodes, ~30k containers


etcd
zookeeper
libkv
consul
Simple File
Docker Hub
Token service
CLI
Docker CLI
Machine discovery
Managers
Agents
docker
SWARM
Overlay networking (vxlan, etc.)

Node Discovery

```
docker-machine create
--engine-opt --cluster-advertise=<value>
--engine-opt --cluster-store=<value>
--engine-opt -cluster-store-opt=<value>
-d=virtualbox
myMachine
```

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

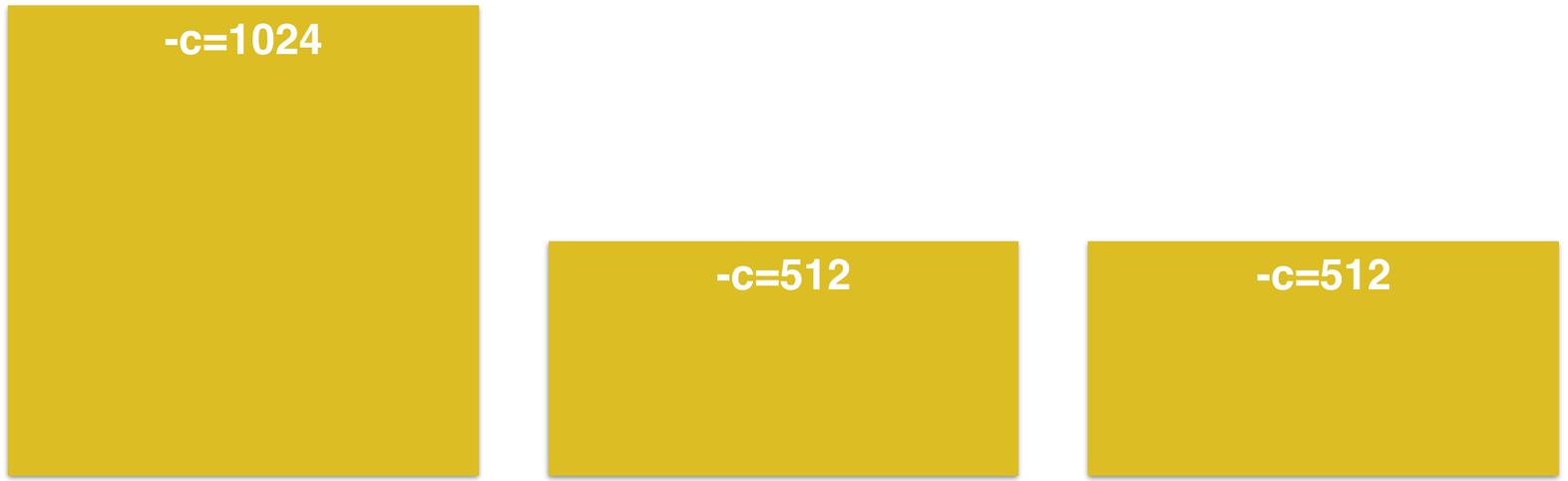
https://docs.docker.com/swarm/multi-manager-setup/

Scheduling Backends

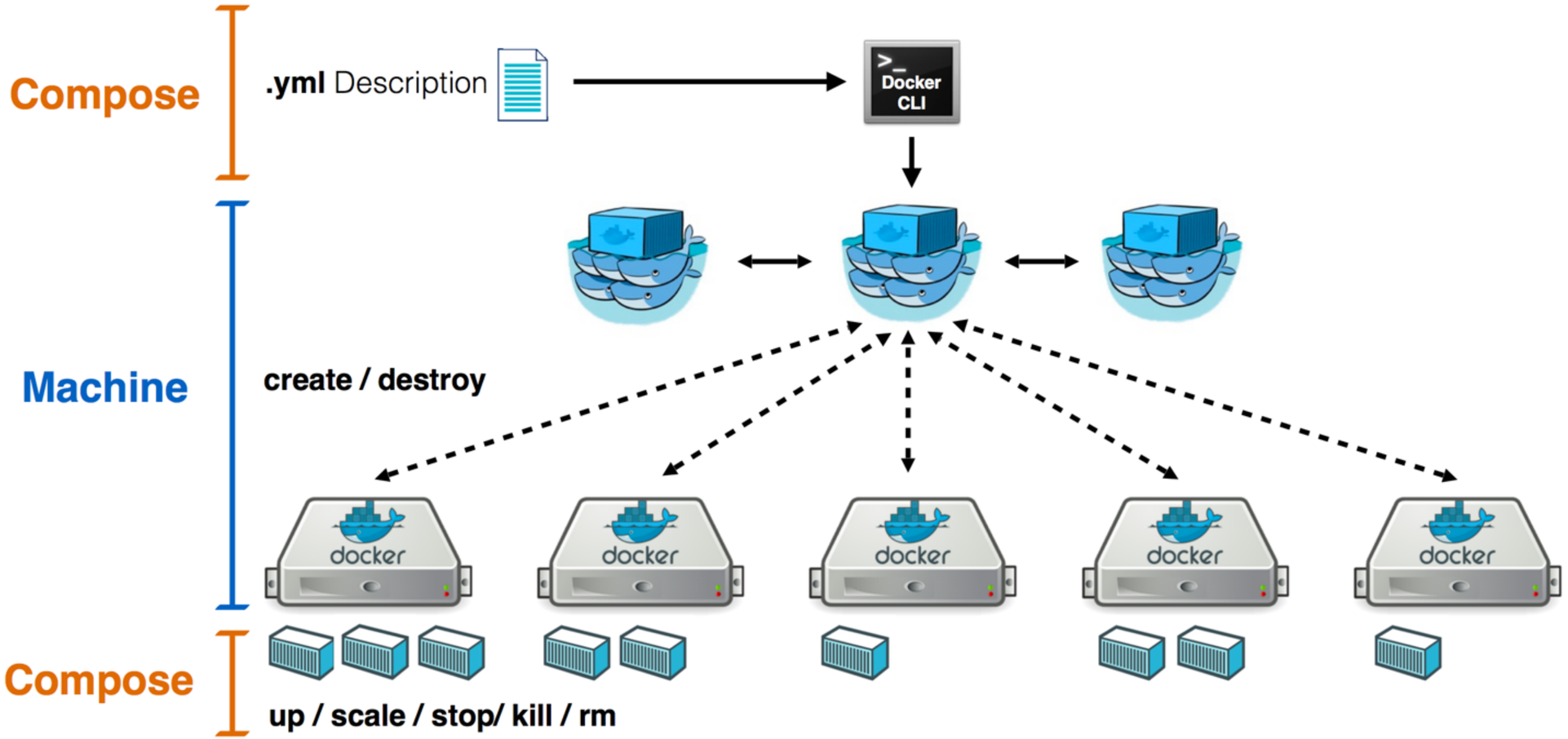
- Based on CPU (`-c`), RAM (`-m`), number of containers
- `docker machine create –strategy <value>`
  - `spread` (default): node with least number of running containers
  - `binpack`: node with most number of running containers
  - `random`: mostly for debugging
- API for pluggable backends (e.g. Mesos)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

Limiting CPU resources

```
docker run --help | grep cpu
  --cpu-shares=0                 CPU shares (relative weight)
  --cpu-period=0                 Limit CPU CFS (Completely Fair Scheduler) period
  --cpu-quota=0                  Limit CPU CFS (Completely Fair Scheduler) quota
  --cpuset-cpus=                 CPUs in which to allow execution (0-3, 0,1)
  --cpuset-mems=                 MEMs in which to allow execution (0-3, 0,1)
```



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/server
```

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

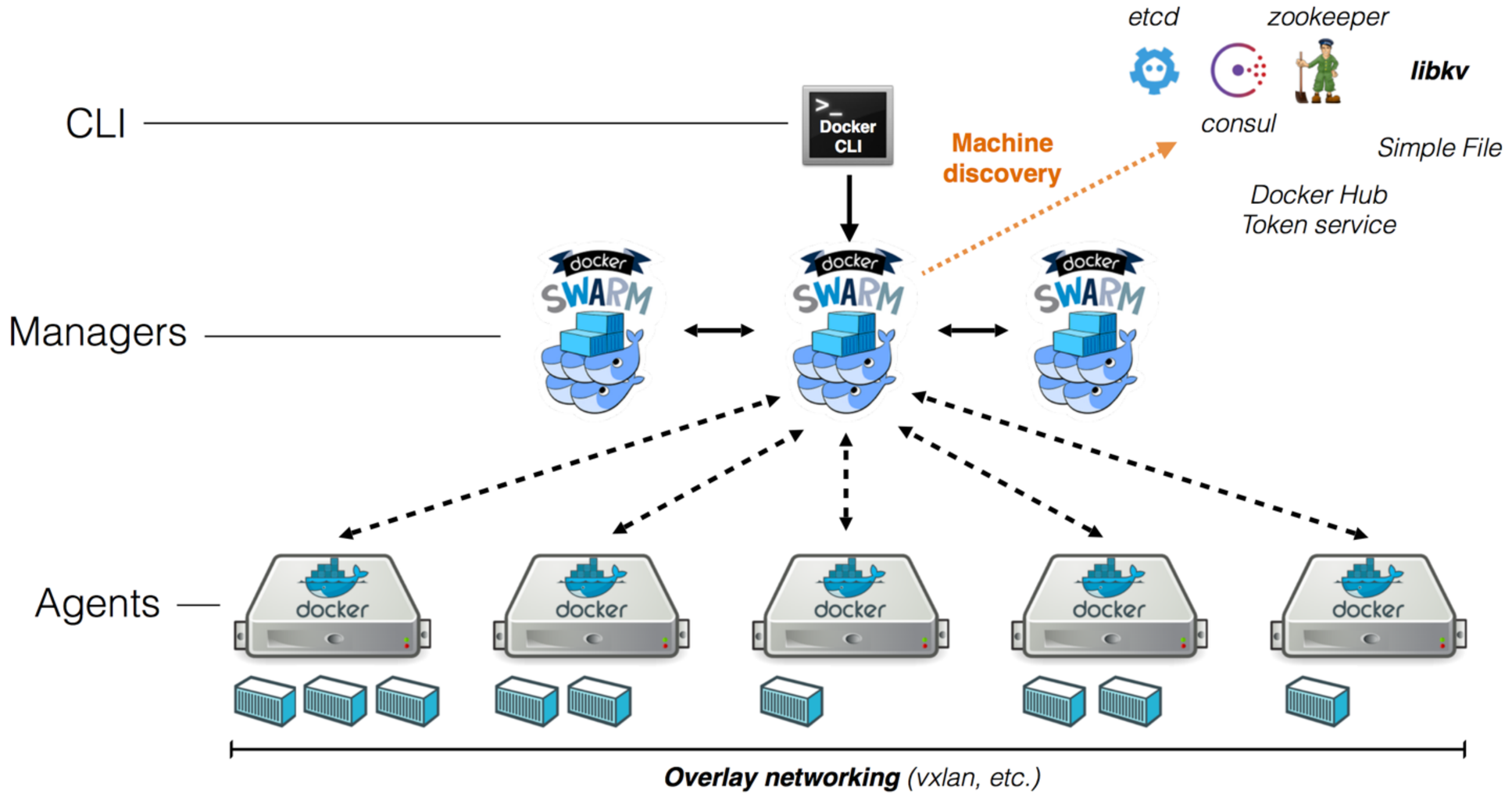
- `docker network ls`

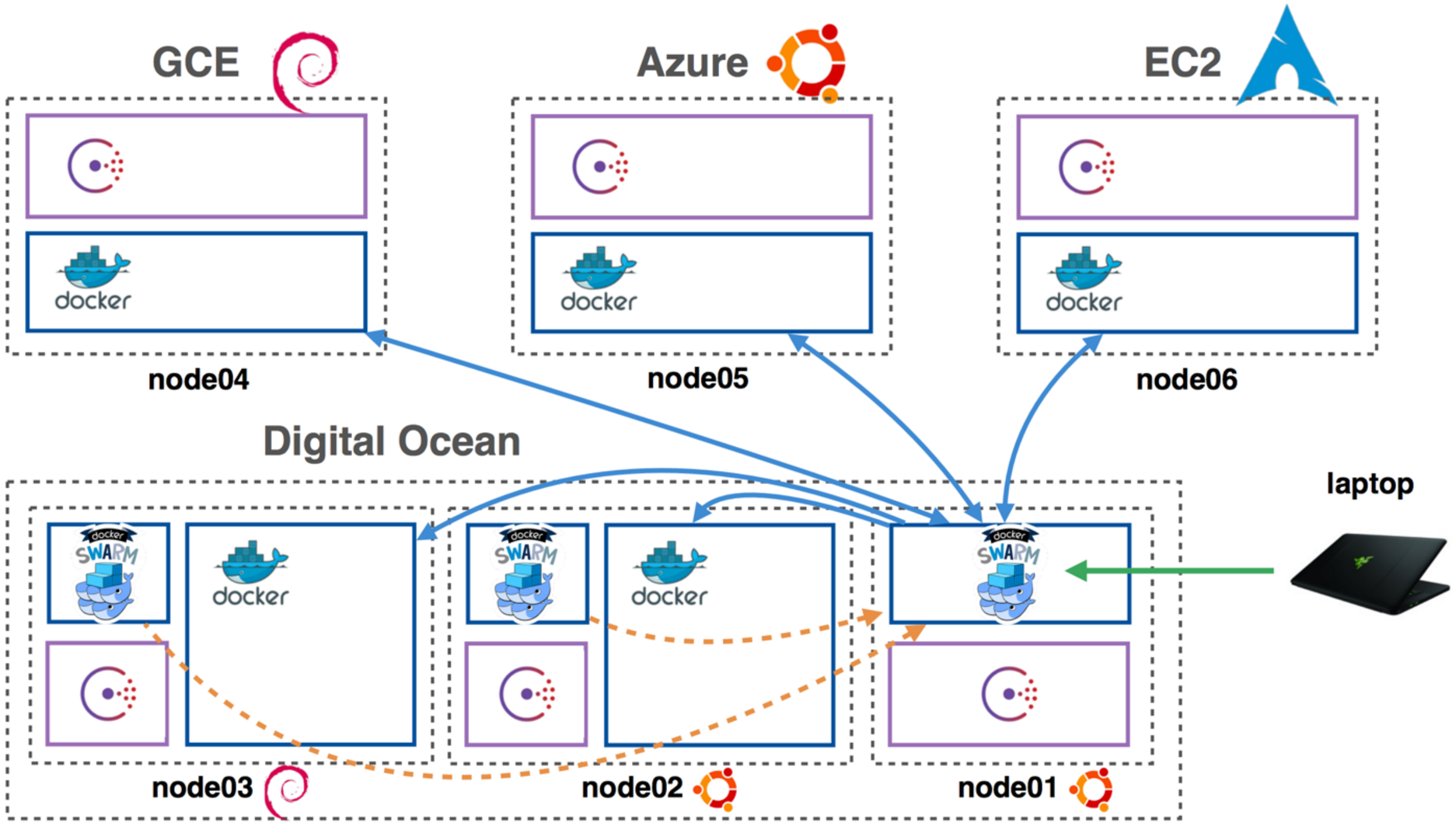
```
NETWORK ID          NAME                DRIVER
1d4d4152b99a        bridge              bridge
8396171e6bbc        none                null
8b50dda5fe61        host                host
```

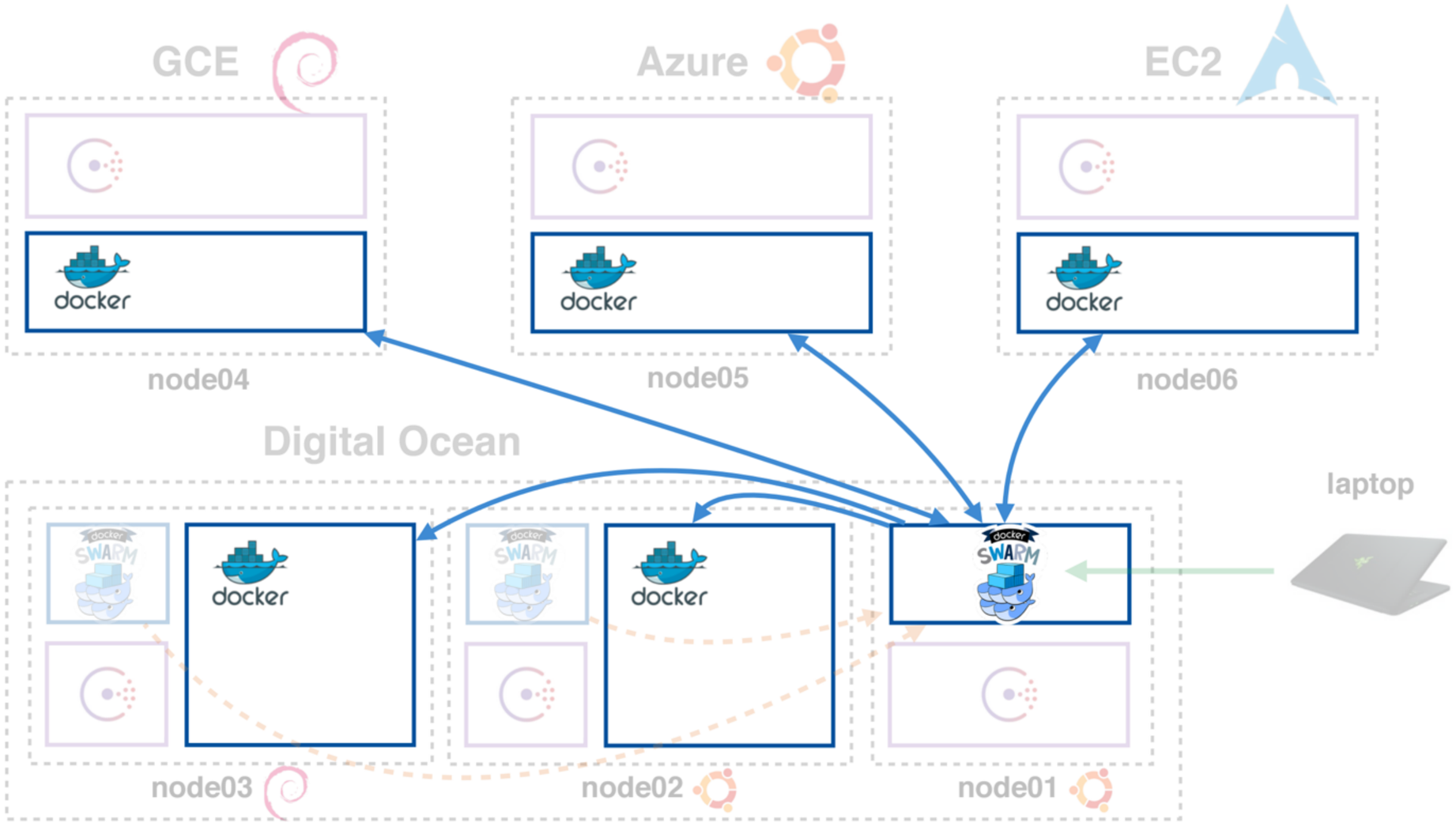
- `docker-compose --x-networking up -d`

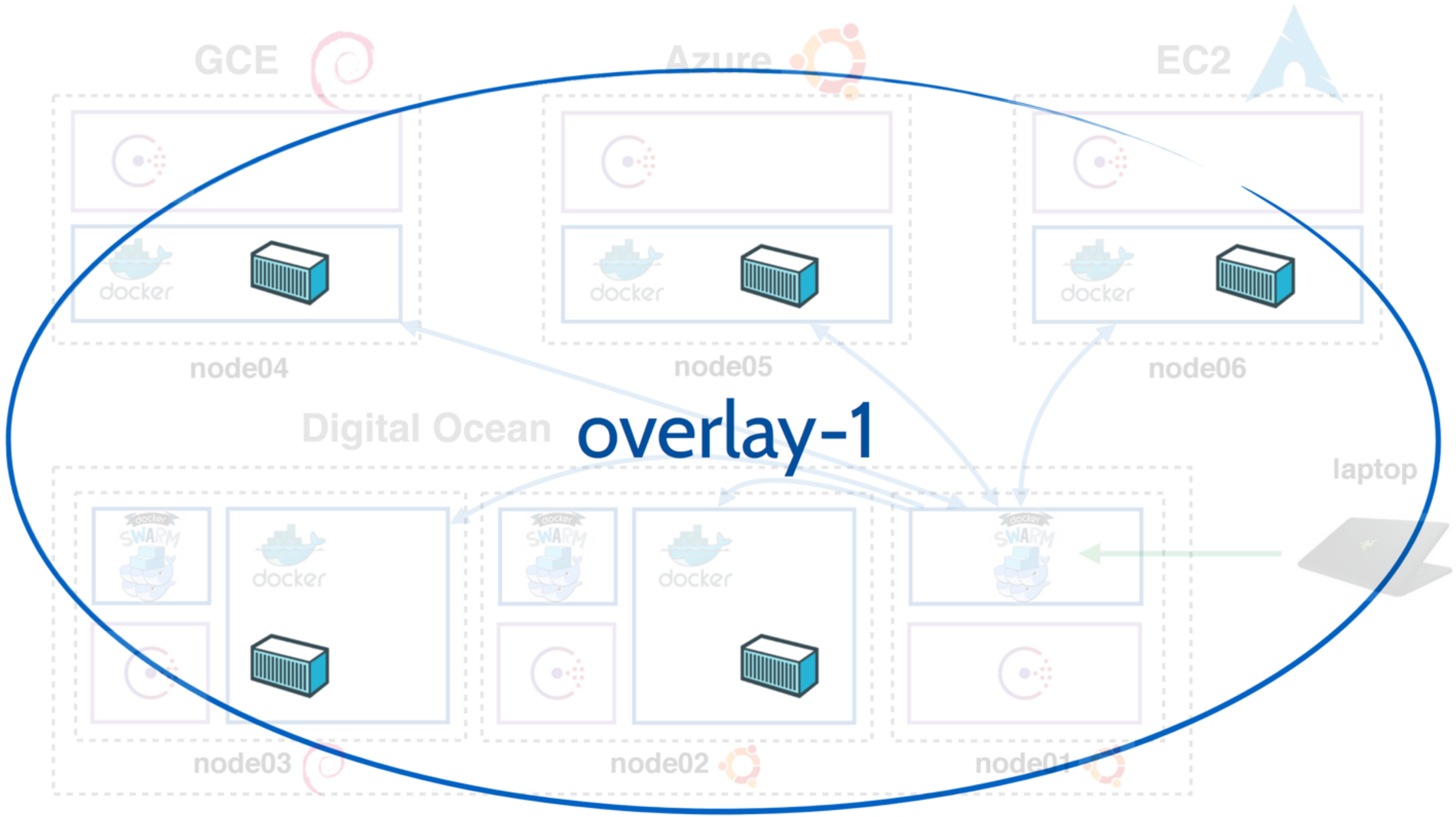
- `docker network ls`

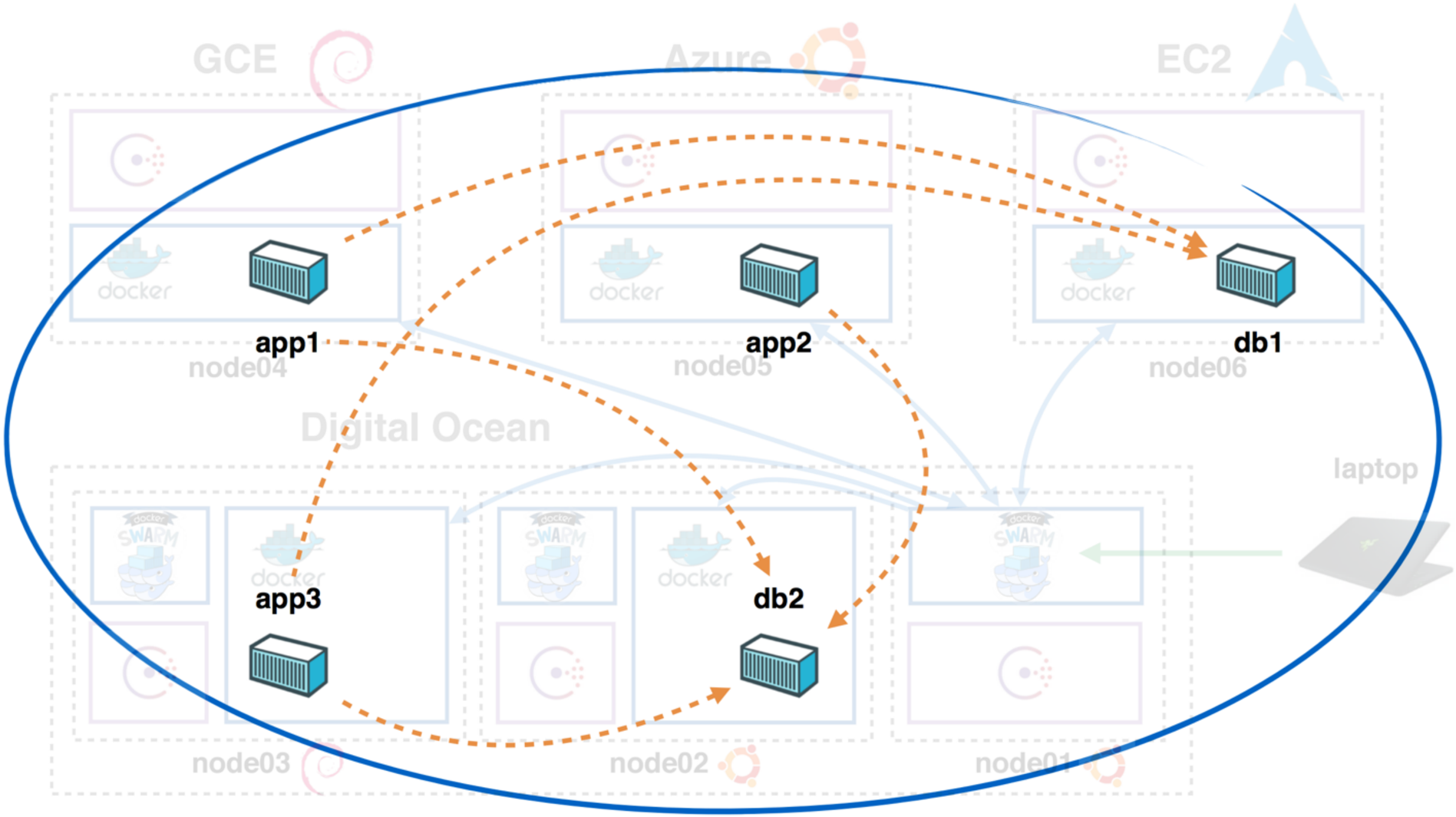
```
NETWORK ID          NAME                DRIVER
1d4d4152b99a        bridge              bridge
8396171e6bbc        none                null
8b50dda5fe61        host                host
b1770955e5aa        wildflymysqljavaee7  bridge
```


etcd
zookeeper
libkv
consul
Simple File
Docker Hub
Token service
CLI
Docker CLI
Machine discovery
Managers
Agents
docker
SWARM
Overlay networking (vxlan, etc.)


GCE
Azure
EC2
docker
node04
node05
node06
Digital Ocean
laptop
SWARM
node03
node02
node01


GCE
Azure
EC2
docker
node04
node05
node06
Digital Ocean
laptop
SWARM
node03
node02
node01


GCE
Azure
EC2
node04
node05
node06
Digital Ocean
overlay-1
laptop
node03
node02
node01


GCE
Azure
EC2
docker
app1
app2
db1
node04
node05
node06
Digital Ocean
laptop
SWARM
app3
db2
node03
node02
node01

References

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

Continuous Delivery using Docker and Jenkins Workflow