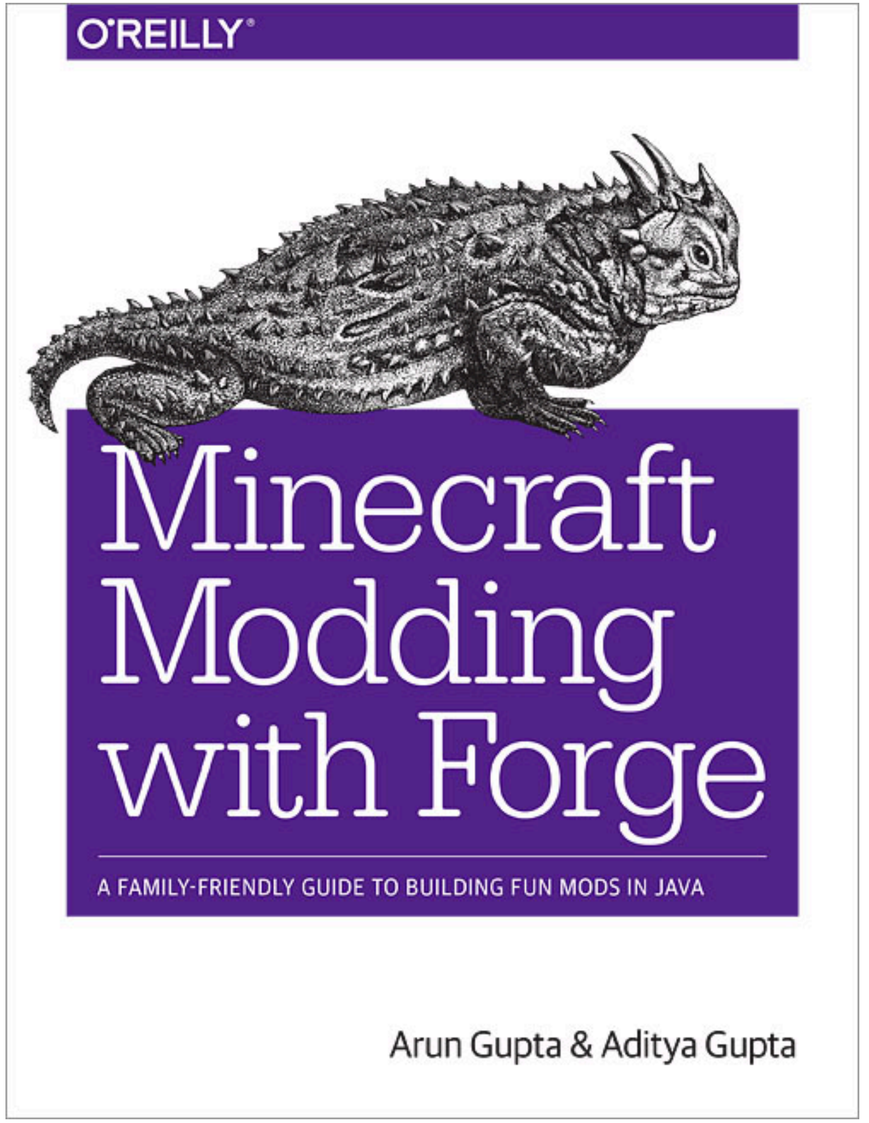
Docker for Java Developers

Arun Gupta, @arungupta
VP Developer Advocacy, Couchbase

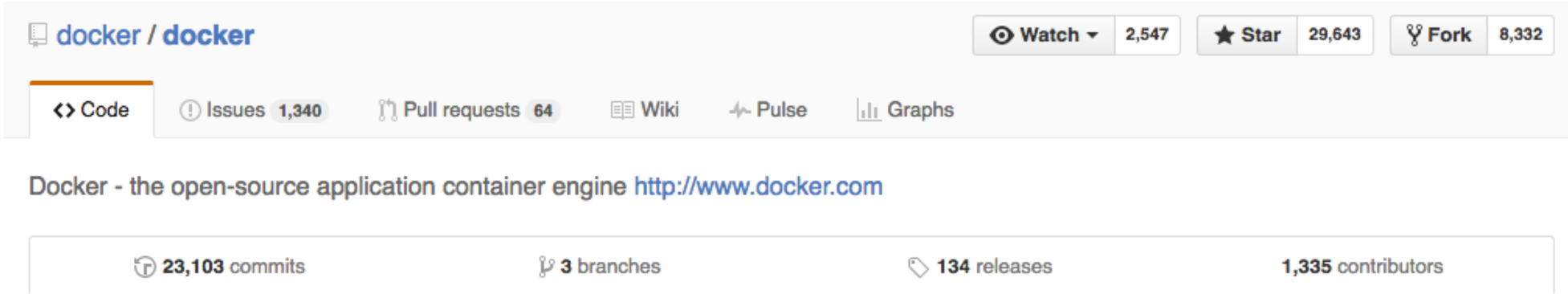
Couchbase

DEVOXX™ 4KIDS USA

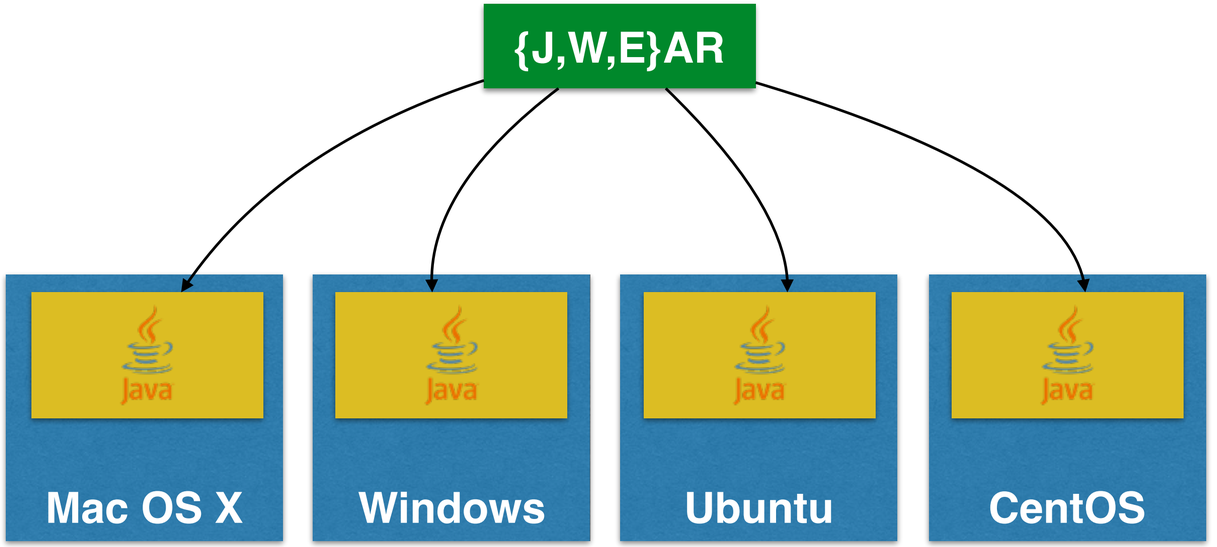


What is Docker?

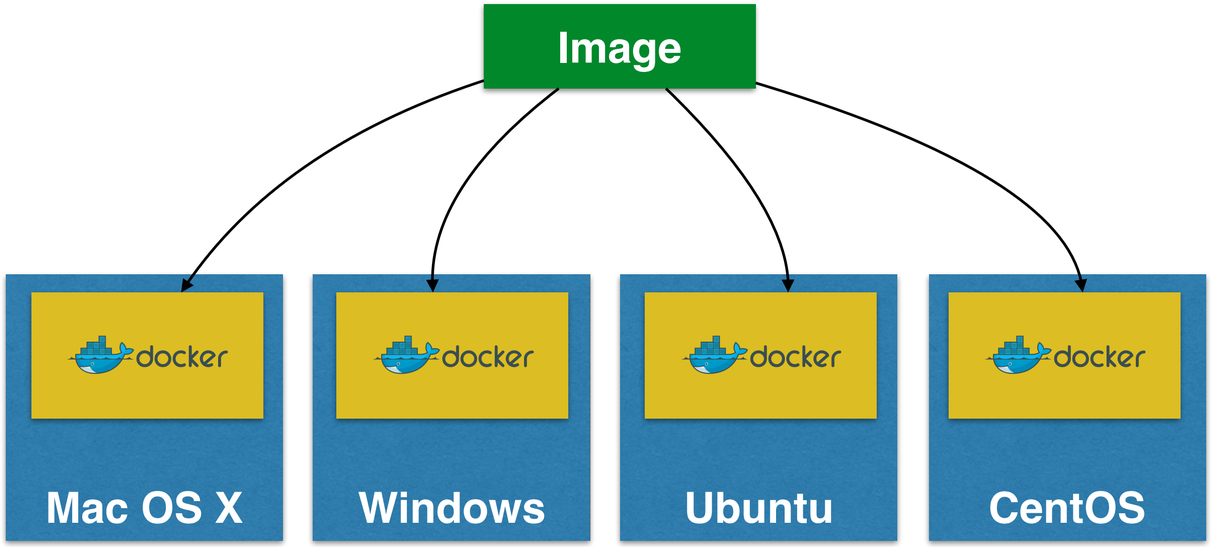
- Open source project and company



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

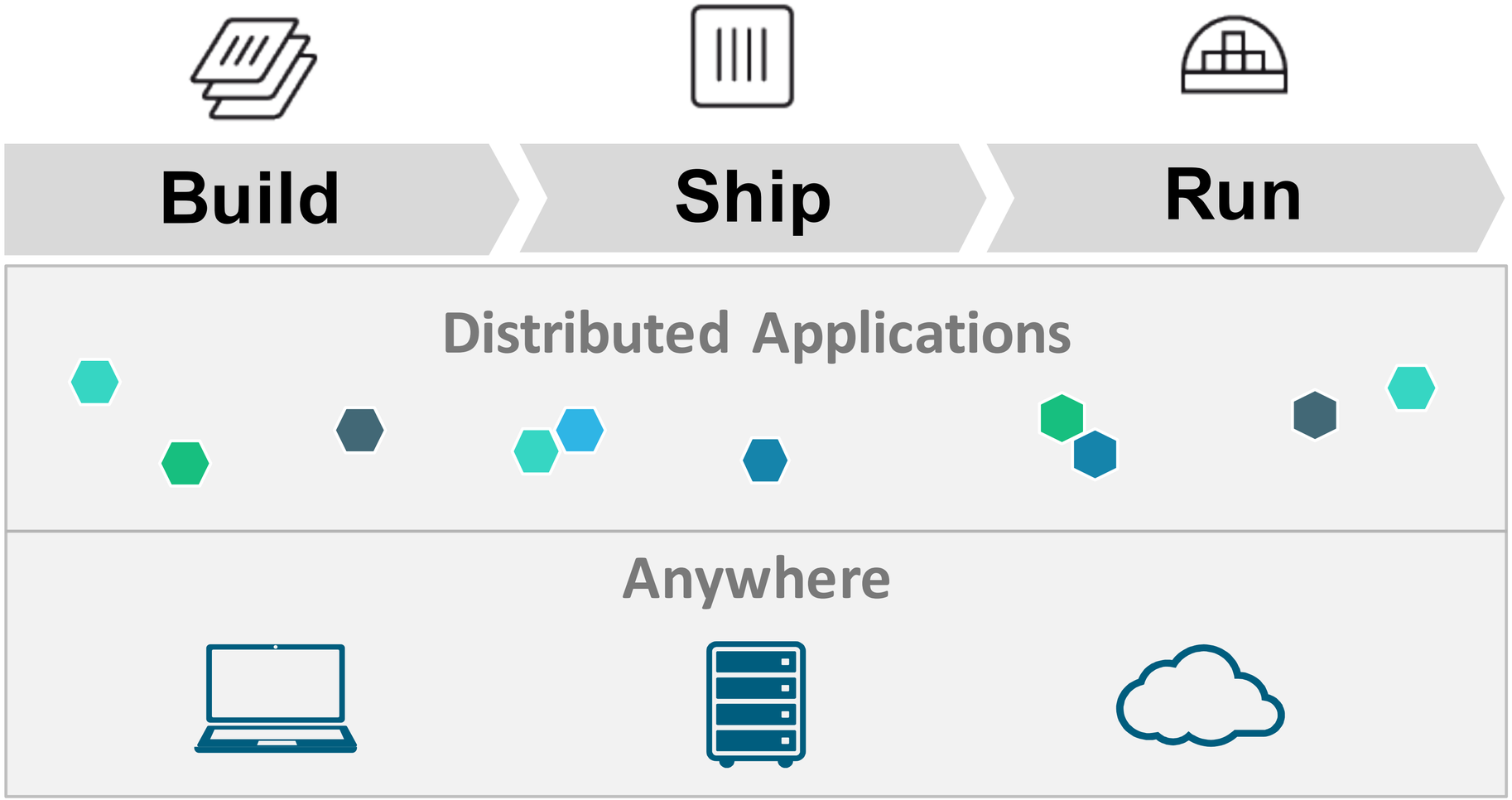


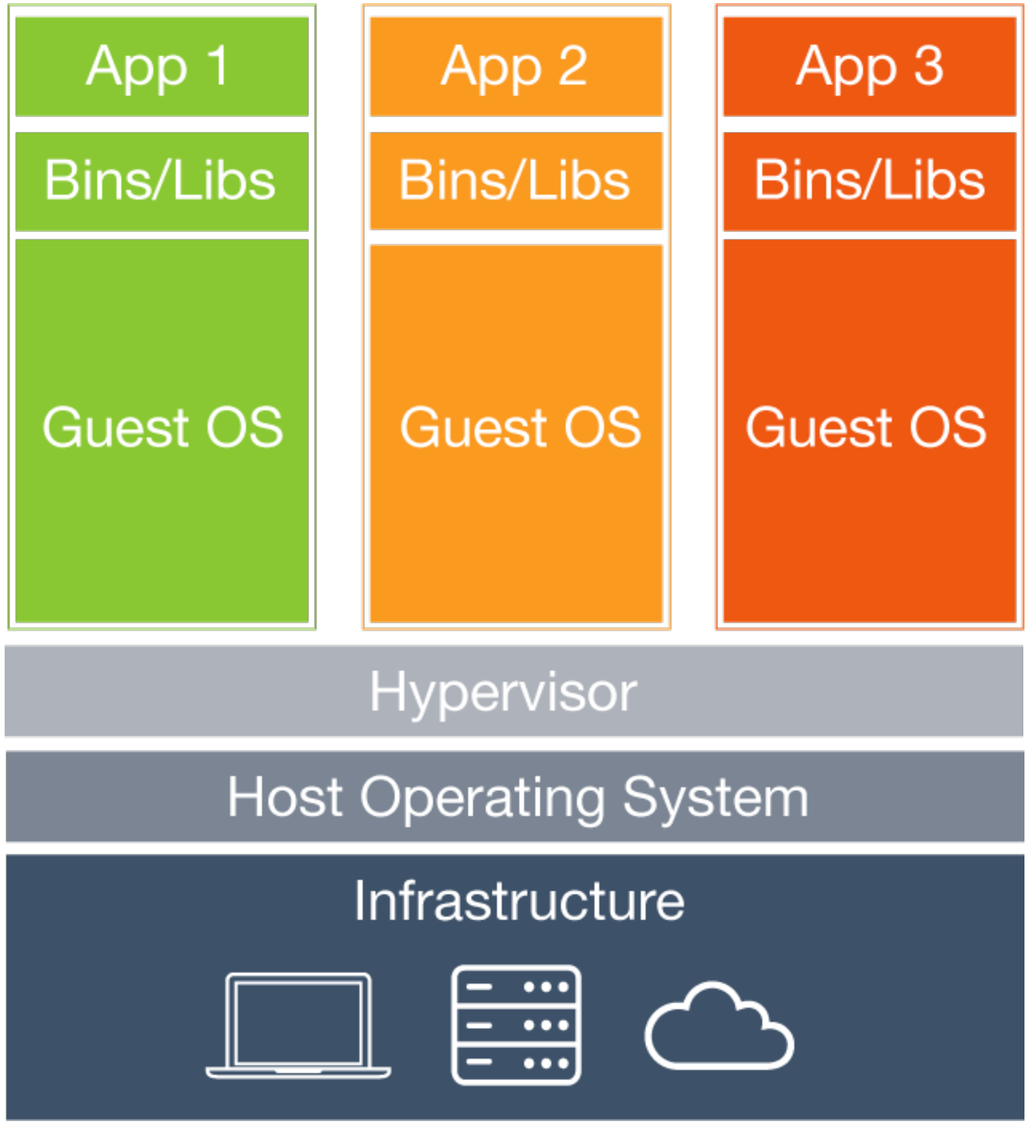
WORA = Write Once Run Anywhere

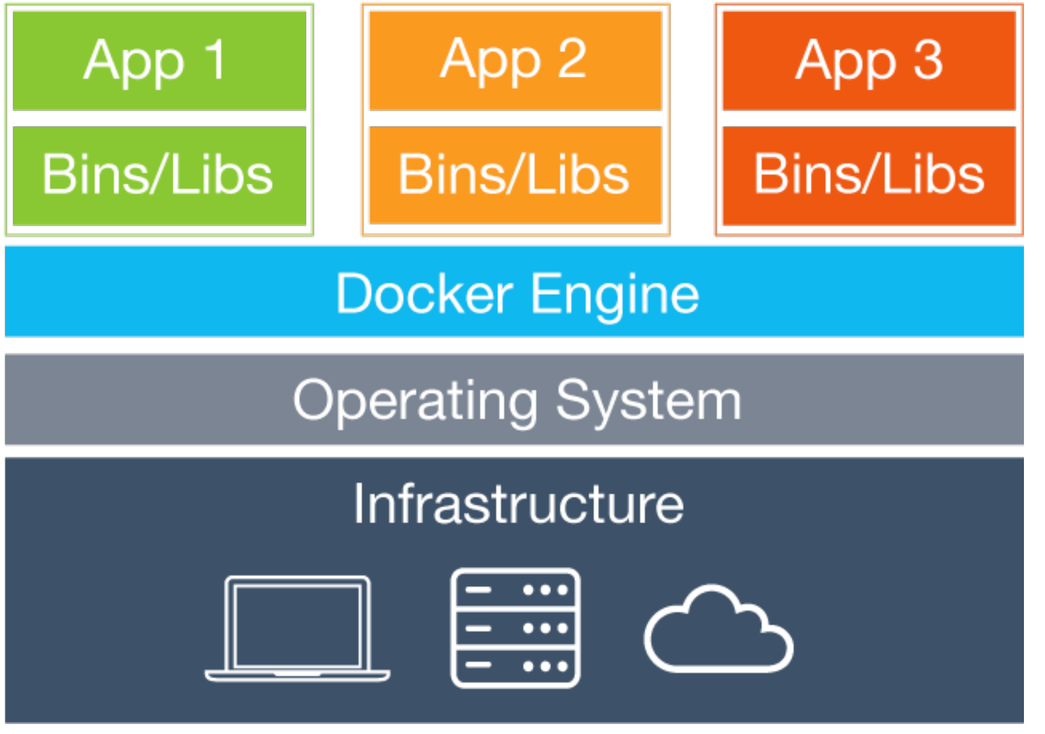


PODA = Package Once Deploy Anywhere

Docker Mission







Build

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

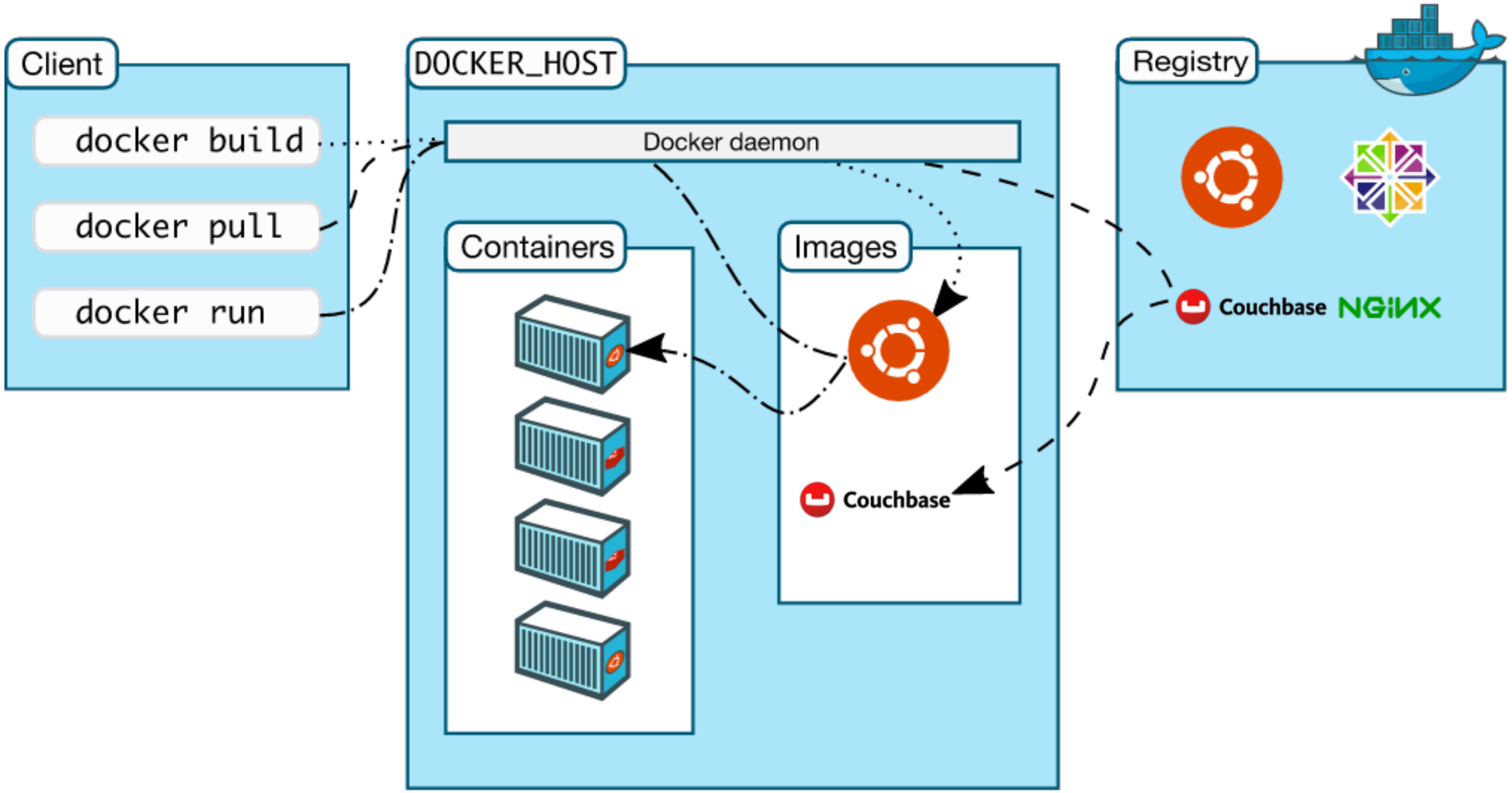
```
FROM ubuntu

CMD echo “Hello world”
```

```
FROM java

COPY target/hello.jar /usr/src/hello.jar

CMD java -cp /usr/src/hello.jar org.example.App
```

- 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

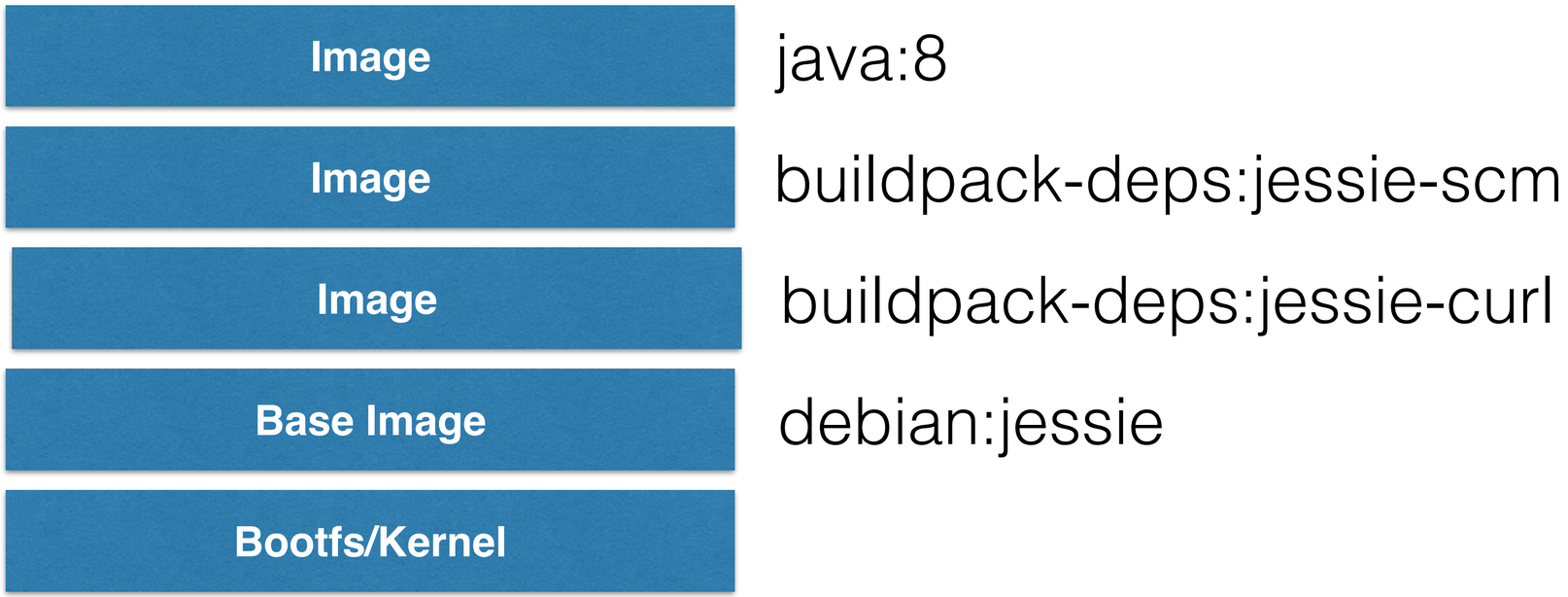


Image Layers - Couchbase

```
~ > 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
COMMENT
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
18165b90fefa        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_VERSIO   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 - Java

```
~ > docker images java
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
java                openjdk-8-jdk-alpine   78afabc37d4f        6 days ago          145.5 MB
java                8                   f298aed75633        6 days ago          642.9 MB
java                latest              f298aed75633        6 days ago          642.9 MB
~ > docker history java
IMAGE               CREATED             CREATED BY                                      SIZE
COMMENT
f298aed75633        6 days ago          /bin/sh -c /var/lib/dpkg/info/ca-certificates   418.2 kB
<missing>           6 days ago          /bin/sh -c set -x  && apt-get update  && apt-   349.2 MB
<missing>           6 days ago          /bin/sh -c #(nop) ENV CA_CERTIFICATES_JAVA_VE   0 B
<missing>           6 days ago          /bin/sh -c #(nop) ENV JAVA_DEBIAN_VERSION=8u7   0 B
<missing>           6 days ago          /bin/sh -c #(nop) ENV JAVA_VERSION=8u72         0 B
<missing>           6 days ago          /bin/sh -c #(nop) ENV JAVA_HOME=/usr/lib/jvm/   0 B
<missing>           6 days ago          /bin/sh -c {   echo '#!/bin/sh';   echo 'set    87 B
<missing>           6 days ago          /bin/sh -c #(nop) ENV LANG=C.UTF-8              0 B
<missing>           6 days ago          /bin/sh -c echo 'deb http://httpredir.debian.   61 B
<missing>           6 days ago          /bin/sh -c apt-get update && apt-get install    1.289 MB
<missing>           2 weeks ago         /bin/sh -c apt-get update && apt-get install    122.6 MB
<missing>           2 weeks ago         /bin/sh -c apt-get update && apt-get install    44.32 MB
<missing>           2 weeks ago         /bin/sh -c #(nop) CMD ["/bin/bash"]             0 B
<missing>           2 weeks ago         /bin/sh -c #(nop) ADD file:b5391cb13172fb513d   125.1 MB
```

Docker for Mac/Window

- Native application and UI
- Auto update capability
- No additional software required, e.g. VirtualBox
  - OSX: xhyve VM using Hypervisor.framework
  - Windows: Hyper-V VM
- Better networking and filesystem mounting/notification
- Public Beta: docker.com/getdocker
- Requires Yosemite 10.10+ or Windows 10 64-bit

Docker for AWS

- Amazon CloudFormation template to start a Swarm of Docker Engines
- Integrated with Autoscaling, ELB, and EBS
- Only a browser and AWS account
- Sign up at beta.docker.com (restricted availability)
- Show demo?

Create stack

Select Template
Specify Details
Options
Review

Review

Template

Template URL	[redacted]
Description	Docker for AWS Beta 1 (docker 1.12.0-rc3)
Estimate cost	Cost

Details

Stack name	Docker4AWS

Swarm Size

ManagerSize	1
ClusterSize	3

Swarm Properties

ManagerInstanceType	t2.micro
InstanceType	m3.medium
KeyName	arun@couchbase
Create IAM resources	Yes

Docker for Azure

- Same as Docker for AWS, but for Azure
- Integrated with VM Scale Sets for autoscaling, Azure Load Balancer, Azure Storage

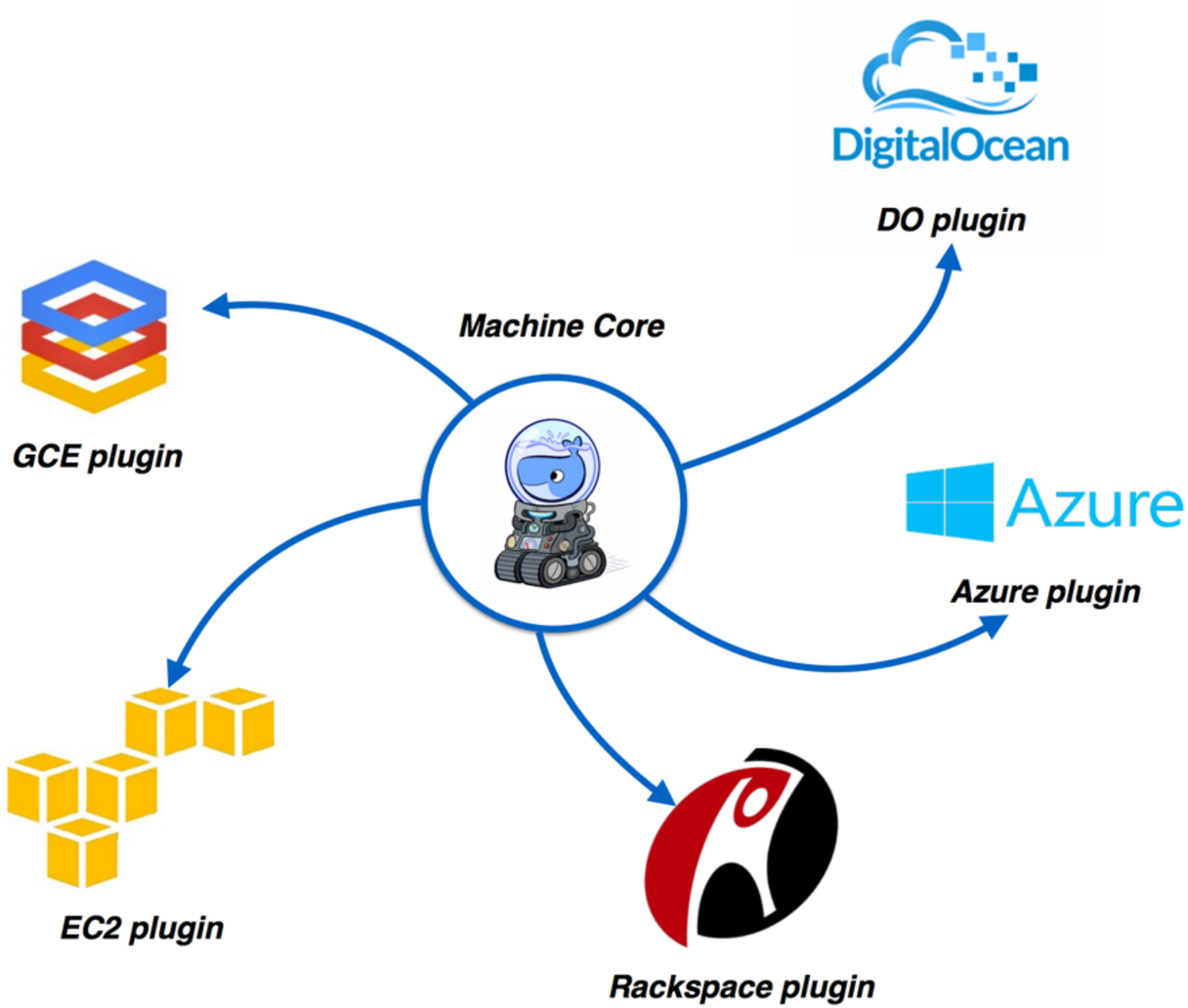
Docker Machine

- Pre-Docker 1.12
- 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

Docker Machine Providers



Docker Toolbox

- Docker Client
- Docker Machine
- Docker Compose
- Docker Kitematic
- Boot2Docker ISO
- Virtualbox

https://www.docker.com/toolbox

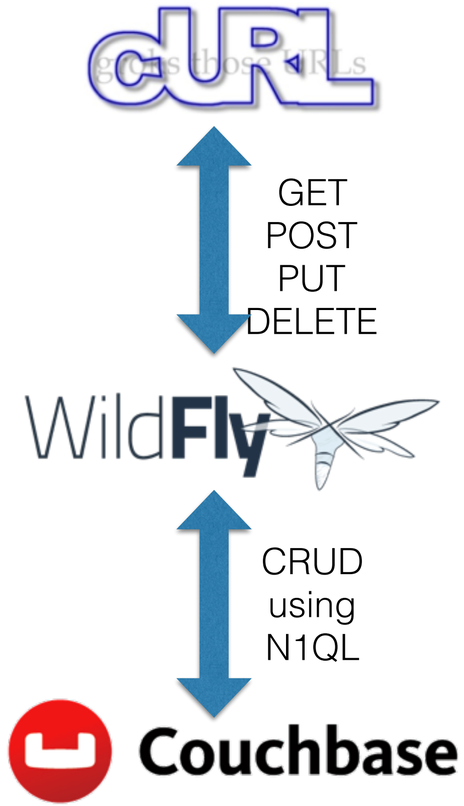
Docker Compose

- 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 - One Service


```
version: “2”
services:
  db:
   image: couchbase
   volumes:
     - ~/couchbase:/opt/couchbase/var
   ports:
     - 8091:8091
     - 8092:8092
     - 8093:8093
     - 11210:11210
```

Docker Compose - Two Services



Docker Compose - Two Services

```
version: “2”
services:
  db:
    image: couchbase
    ports:
      - 8091:8091
      - 8092:8092
      - 8093:8093
      - 11210:11210
  web:
    image: arungupta/wildfly
    environment:
      - COUCHBASE_URI=db
    ports:
      - 8080:8080
      - 9990:9990
```

Overriding Services in Docker Compose

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

docker-compose.yml

```
web:
  ports:
    - 9080:8080
```

docker-compose.override.yml

```
docker-compose up -d
```

Dev/Prod with Compose

```
db-dev:
  image: arungupta/couchbase
  ports:
    - . . .
web:
  image: arungupta/wildfly
  environment:
    - COUCHBASE_URI=db-dev:8093
  ports:
    - 8080:8080
```

docker-compose.yml

```
docker-compose up -d
```

```
web:
 environment:
    - COUCHBASE_URI=db-prod:8093
  ports:
    - 8080:80
db-prod:
  image: . . .
```

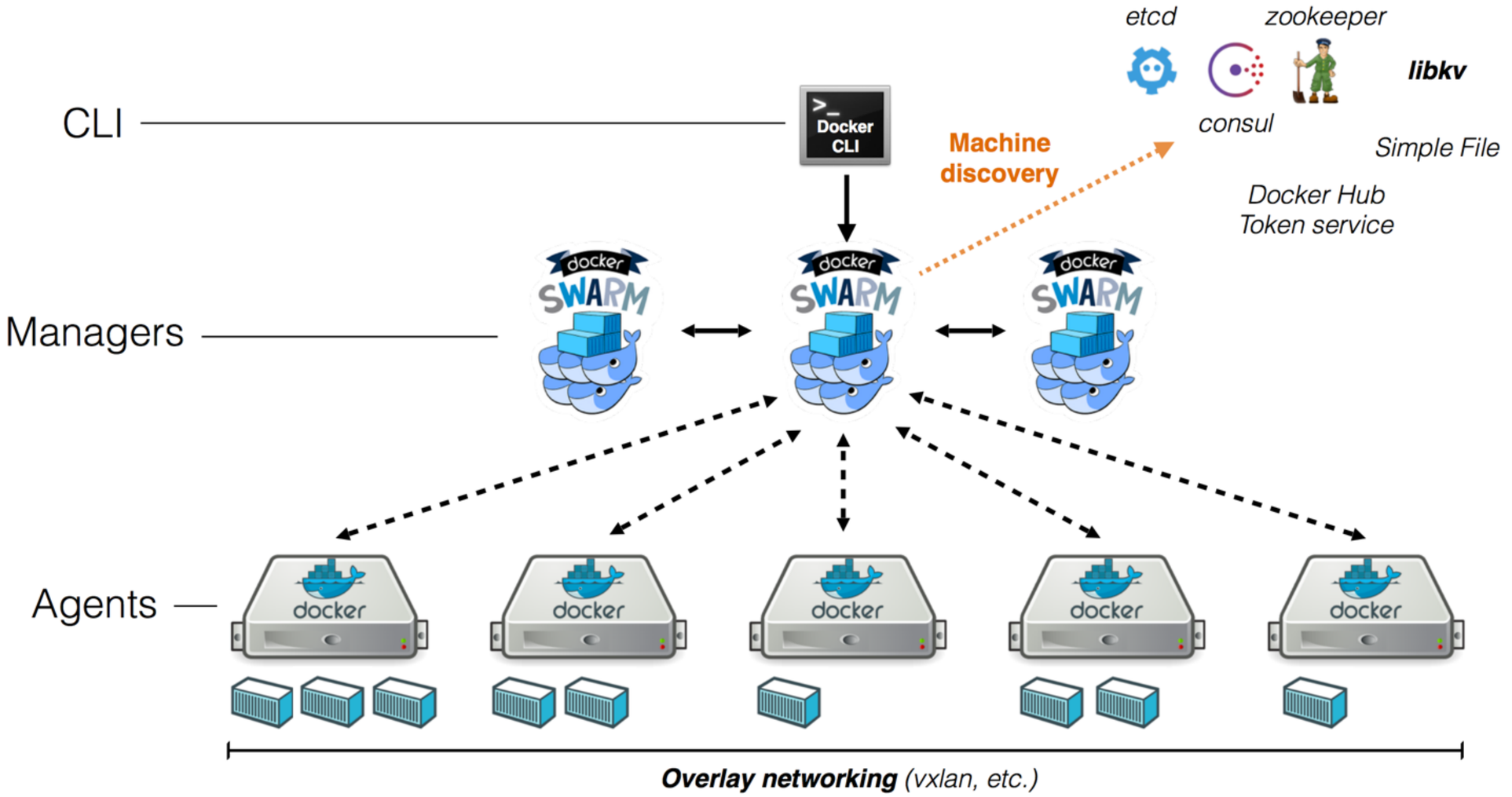
production.yml

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

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
- 1.2.4 - Ready for production
  - Reschedule containers when a node fails
  - 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

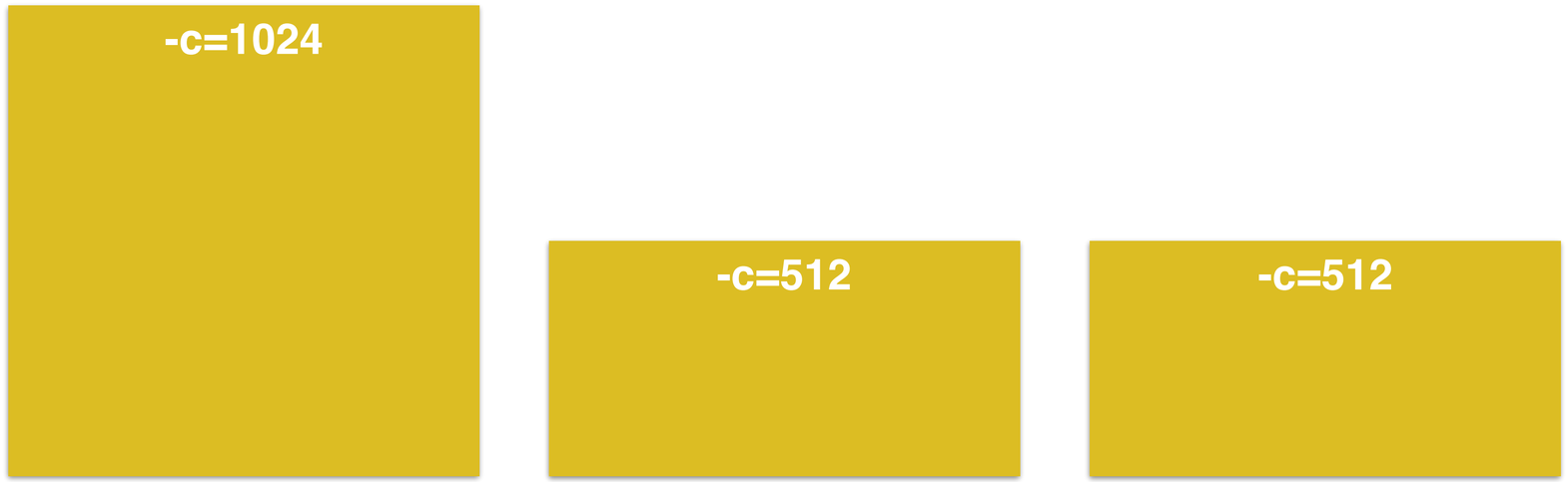
https://docs.docker.com/swarm/multi-manager-setup/

Scheduling Backends

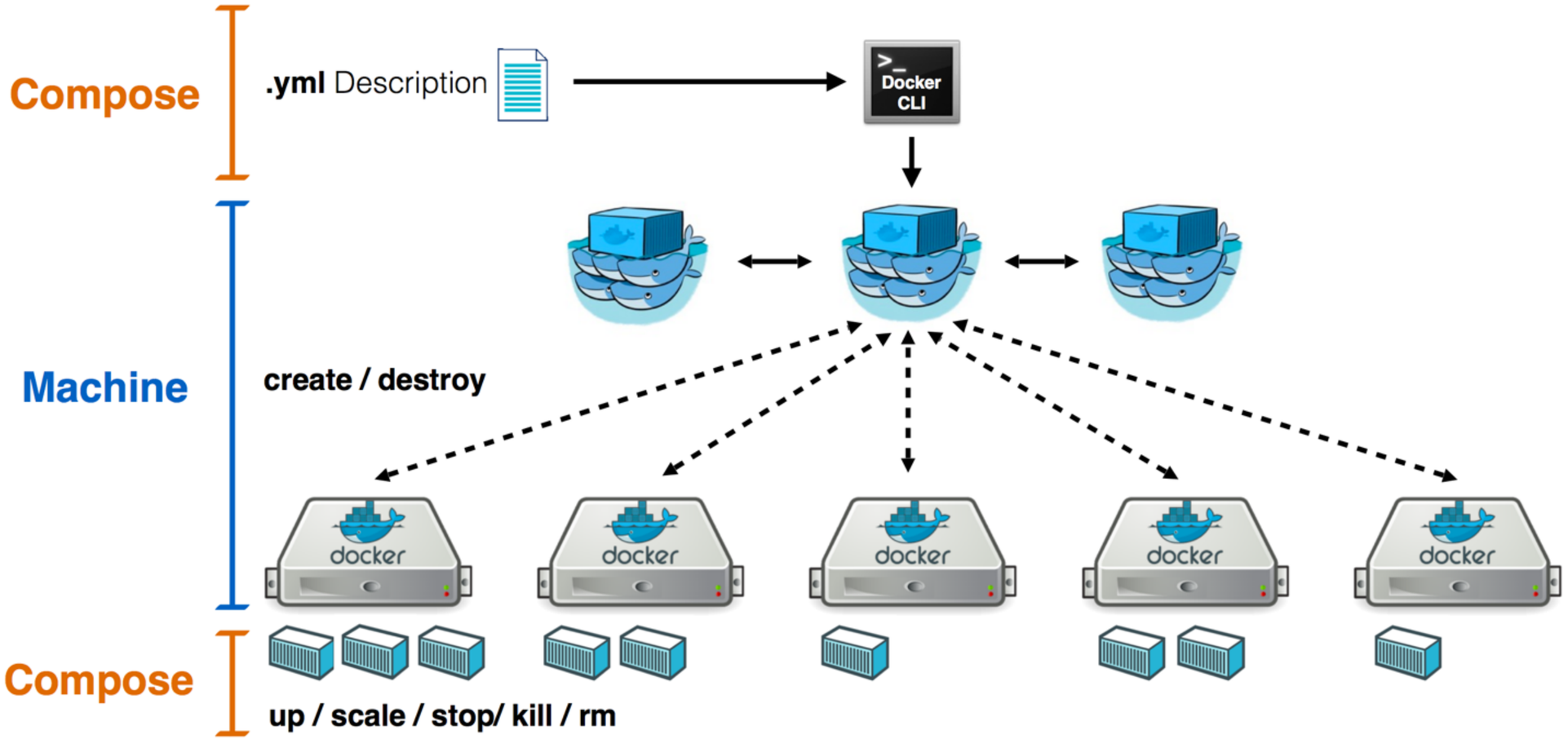
- Based on CPU (`-c`), RAM (`-m`), number of containers
- `docker machine create –strategy <value>`
  - `spread` (default): node with least number of running containers
  - `binpack`: node with most number of running containers
  - `random`: mostly for debugging
- API for pluggable backends (e.g. Mesos)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

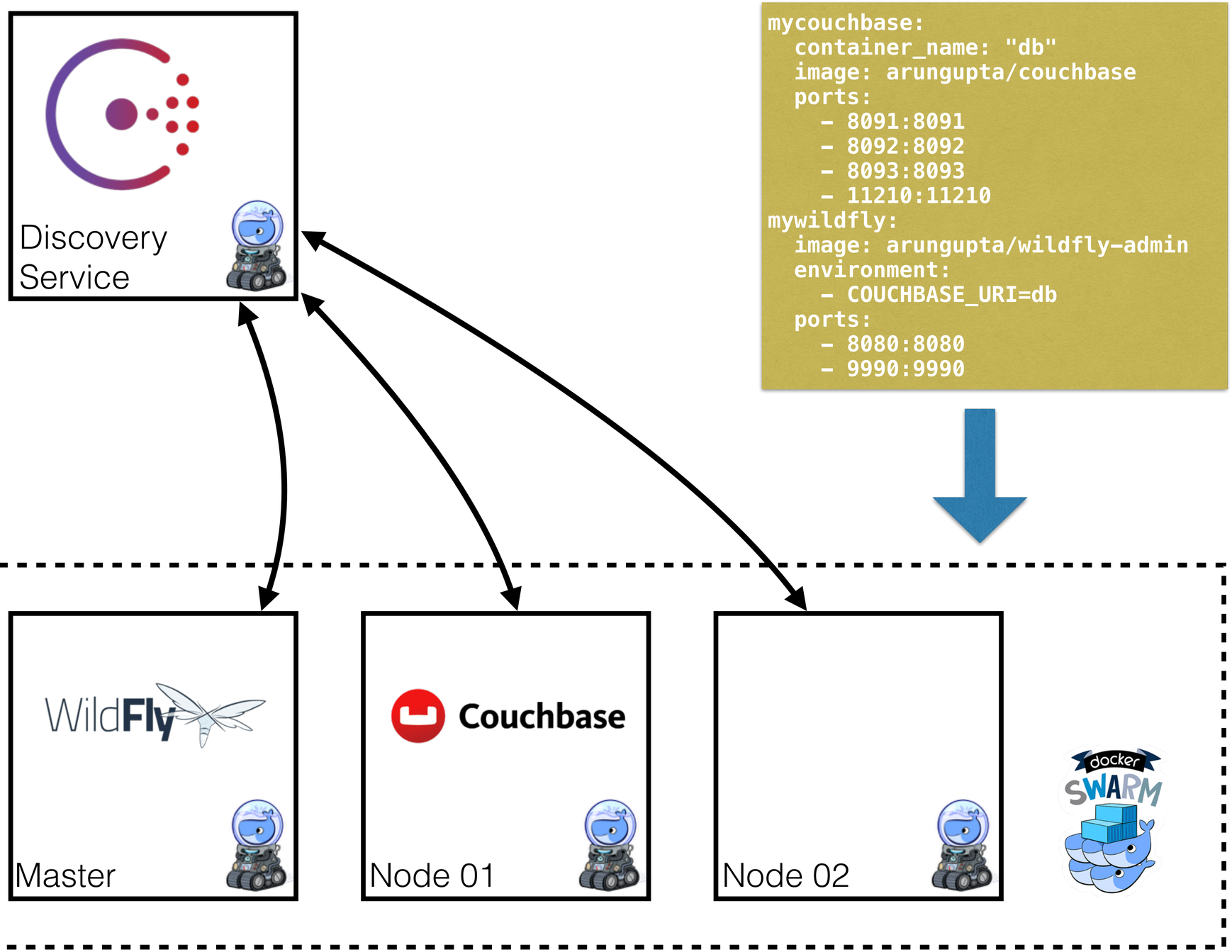
Limiting CPU resources

```
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





http://blog.arungupta.me/docker-machine-swarm-compose-couchbase-wildfly/

Swarm Mode Overview

- New in 1.12, may be backwards incompatible
- Natively managing a cluster of Docker Engines called a Swarm
- Use Docker CLI to create a swarm, deploy apps, and manage swarm
- Uses SwarmKit

Swarm Kit

- Toolkit for orchestrating distributed systems
- Raft-based consensus
- Swarm node - Worker and/or Manager
  - Worker comes with Docker Container Executor
  - Tasks run on Worker
  - Tasks are organized in Services
  - Service has desired state (e.g. how many replicas)
- Resource/strategy aware - *spread* is default
- Reconcile the desired and actual state
- Secure OOTB using mutual TLS

Swarm Mode Features

- Cluster management integrated with Docker Engine
- Decentralized design
- Declarative service model
- Scaling
- Desired state reconciliation
- Multi-host networking
- Service discovery
- Load balancing
- Secure by default
- Rolling updates

Swarm Mode cluster on Amazon

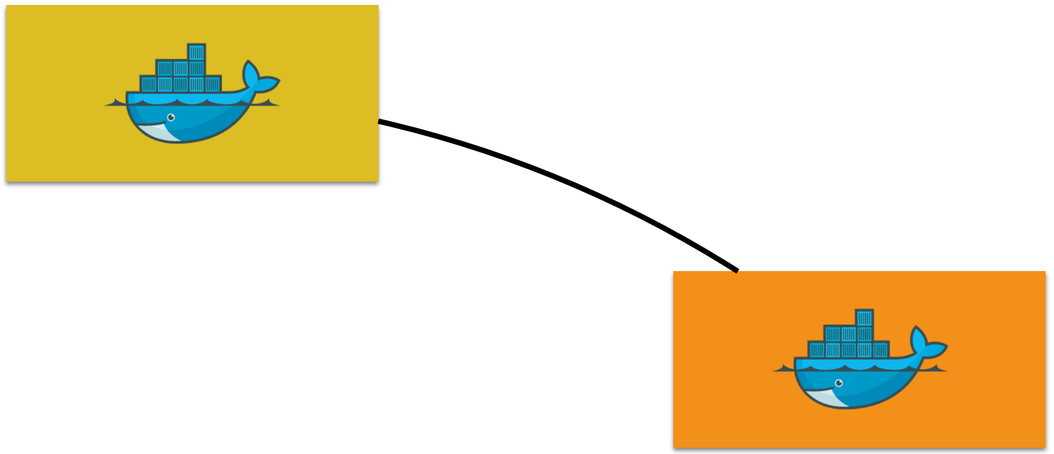
Type	Protocol	Port Range	Source
SSH	TCP	22	Anywhere 0.0.0.0/0
Custom TCP Rule	TCP	2377	Anywhere 0.0.0.0/0
Custom TCP Rule	TCP	7946	Anywhere 0.0.0.0/0
Custom UDP Rule	UDP	7946	Anywhere 0.0.0.0/0
Custom TCP Rule	TCP	4789	Anywhere 0.0.0.0/0
Custom UDP Rule	UDP	4789	Anywhere 0.0.0.0/0

- 2377: Cluster management communications
- 7946: Communication among nodes
- 4789: Overlay network traffic

Swarm Mode: Initialize

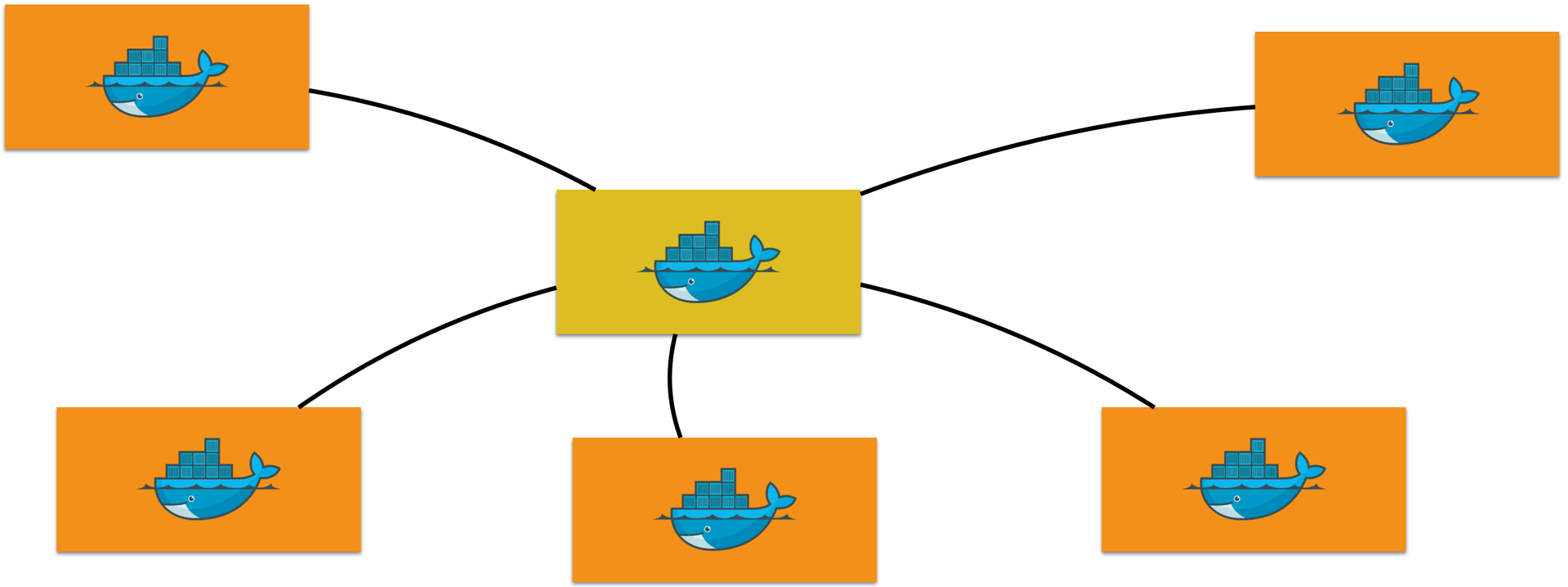
```
docker swarm init --listen-addr <ip>:2377 --secret <SECRET>
```

Swarm Mode: Add Worker



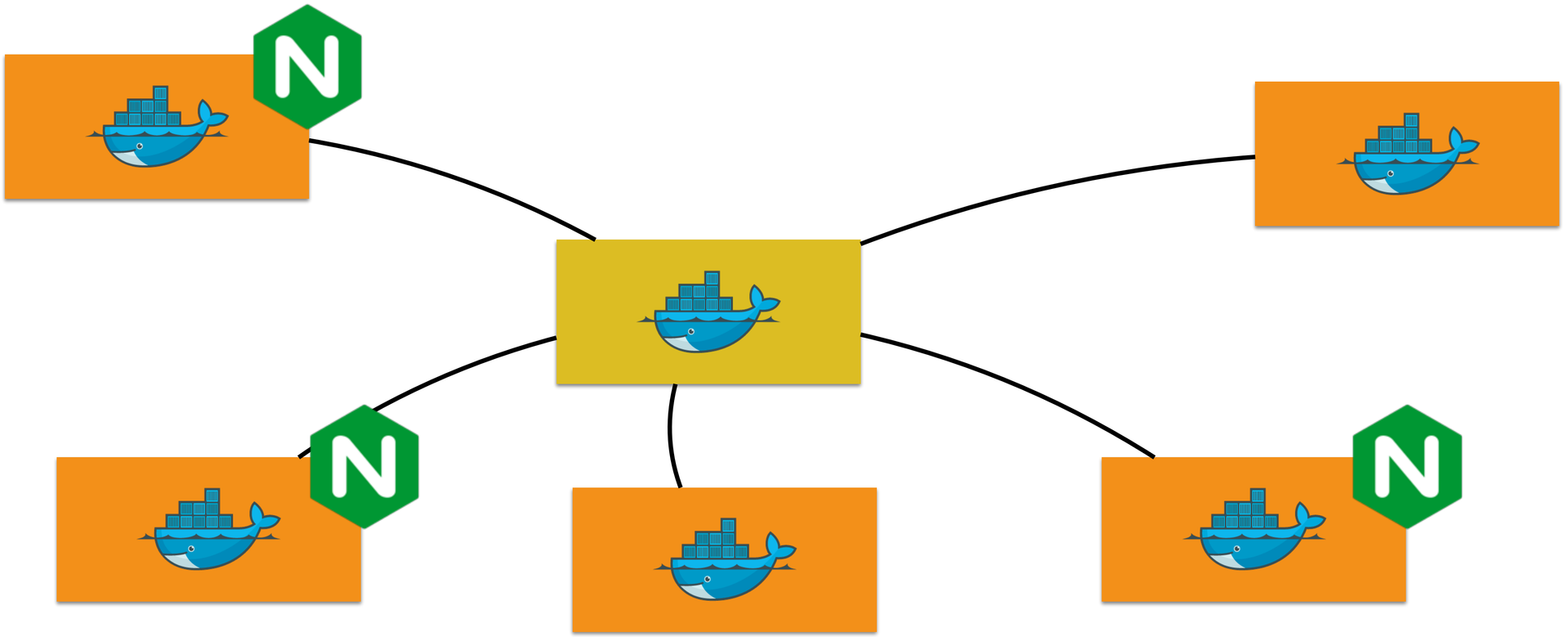
```
docker swarm join --secret <SECRET> <IP of manager>:2377
```

Swarm Mode: Add More Workers



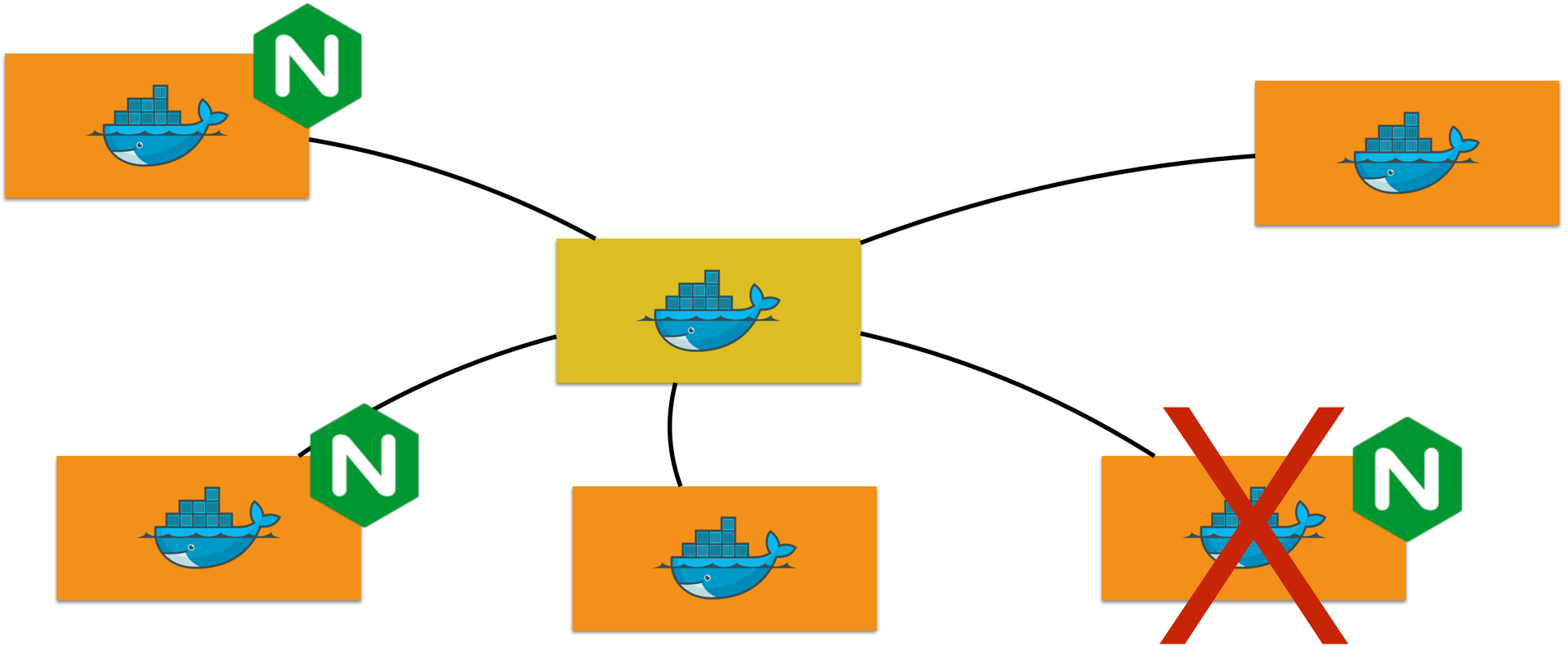
```
docker swarm join --secret <SECRET> <IP of manager>:2377
```

Swarm Mode: Replicated Service

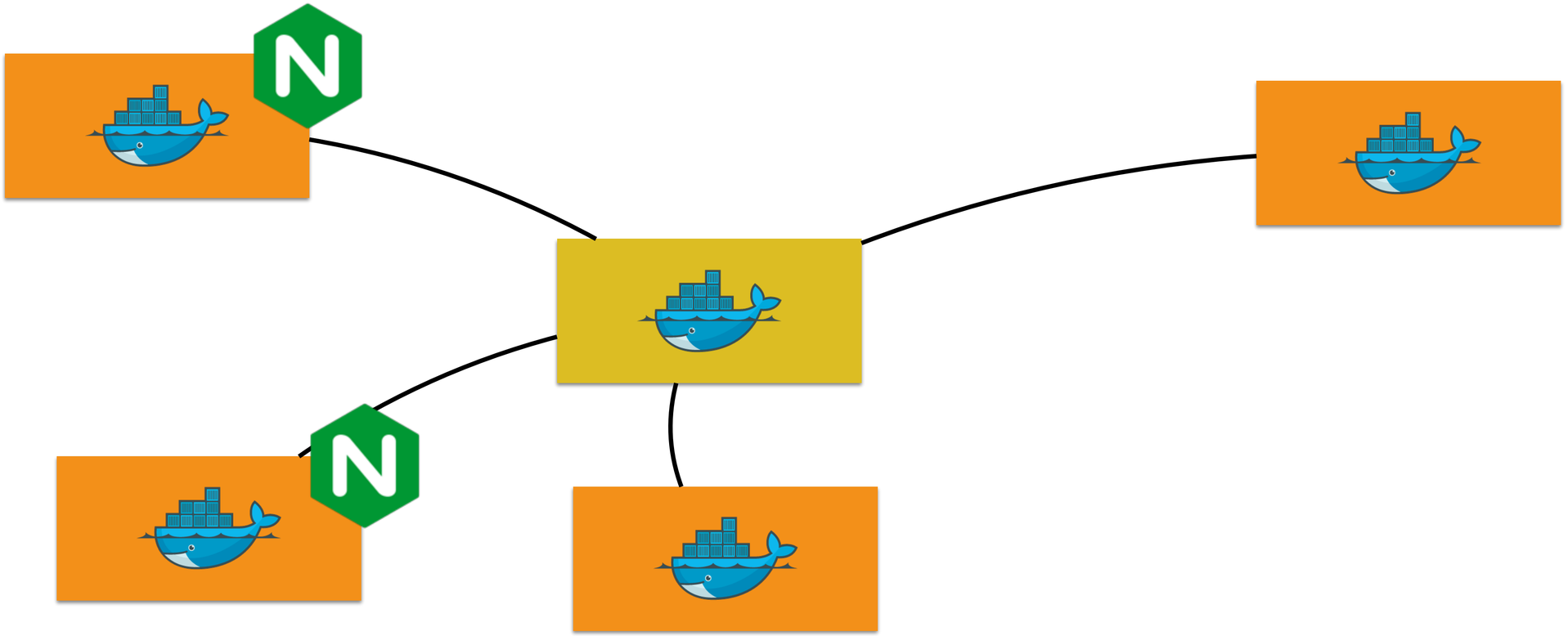


```
docker service create --replicas 3 --name frontend nginx:latest
```

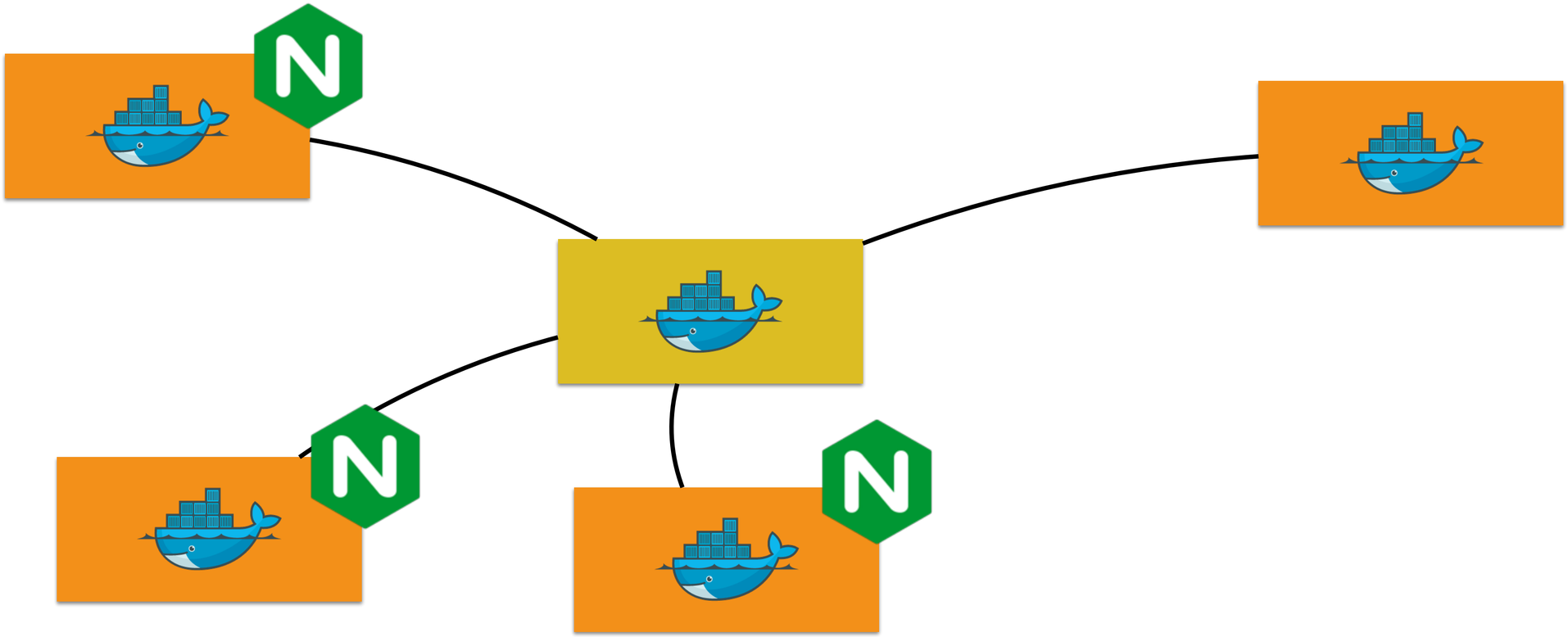
Swarm Mode: Node Failure



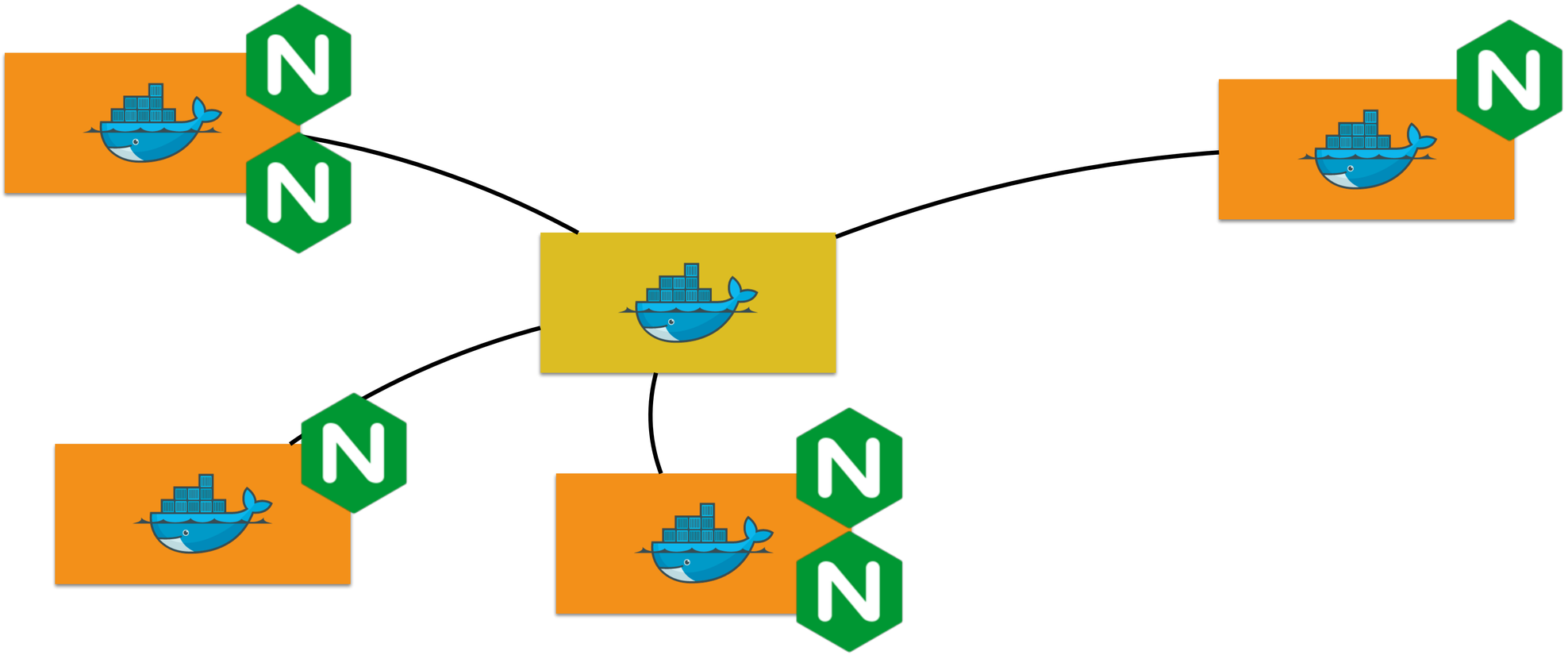
Swarm Mode: Desired != Actual



Swarm Mode: Reconcile

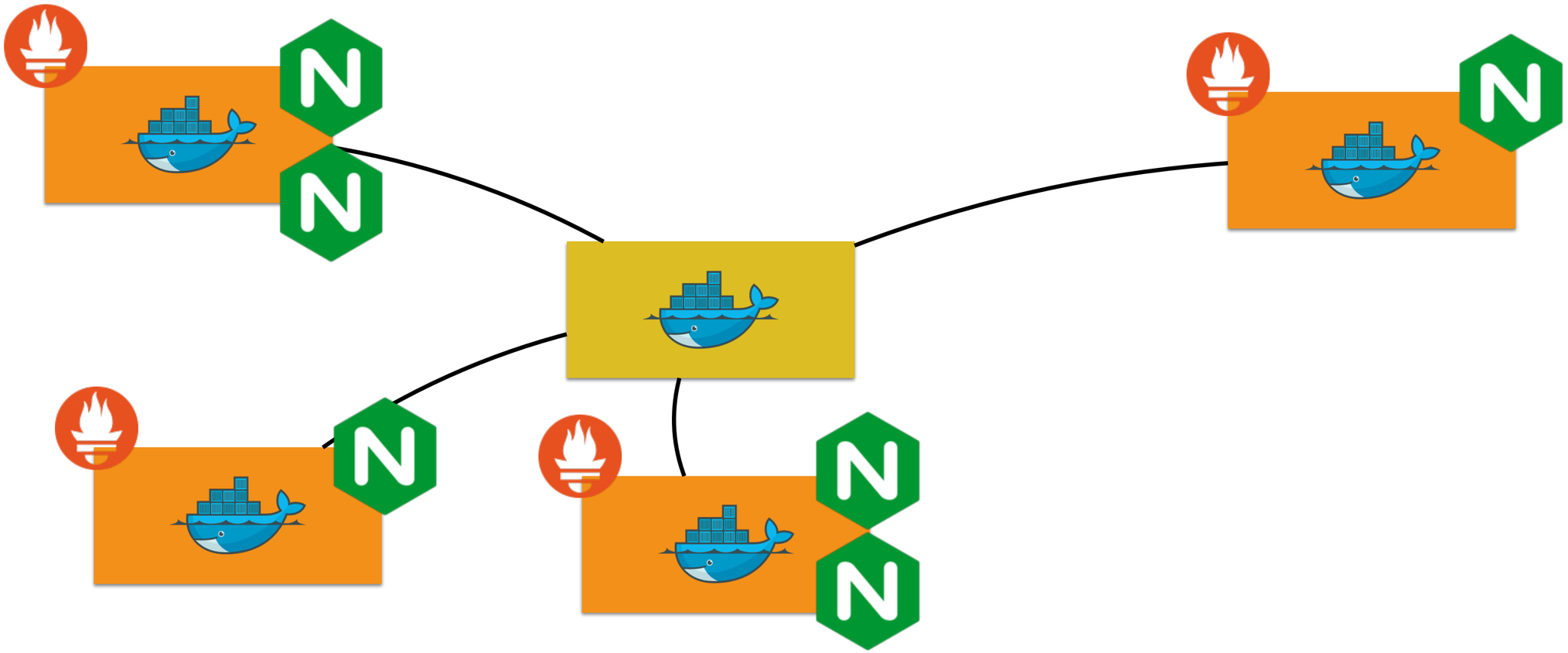


Swarm Mode: Scale



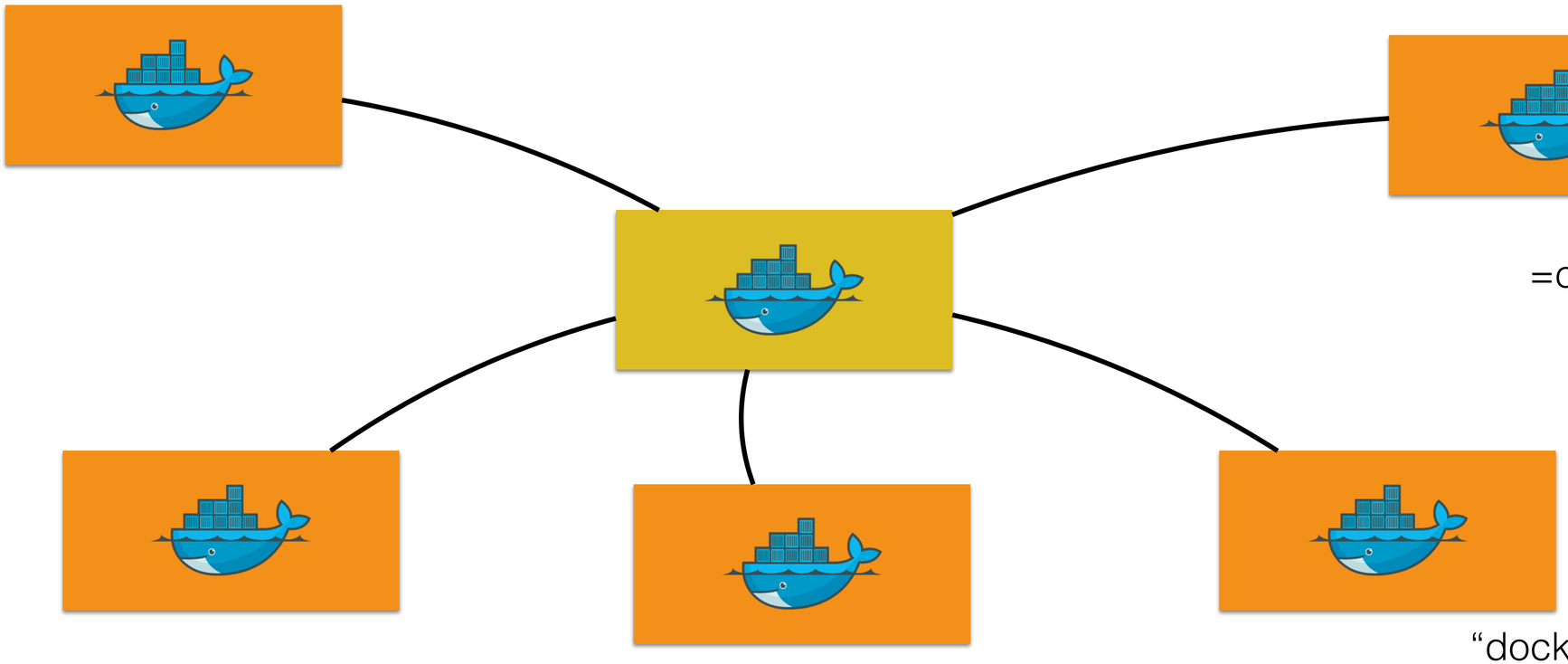
```
docker service scale frontend=6
```

Swarm Mode: Global Service



```
docker service create --mode=global --name=prom prom/prometheus
```

Swarm Mode: Label

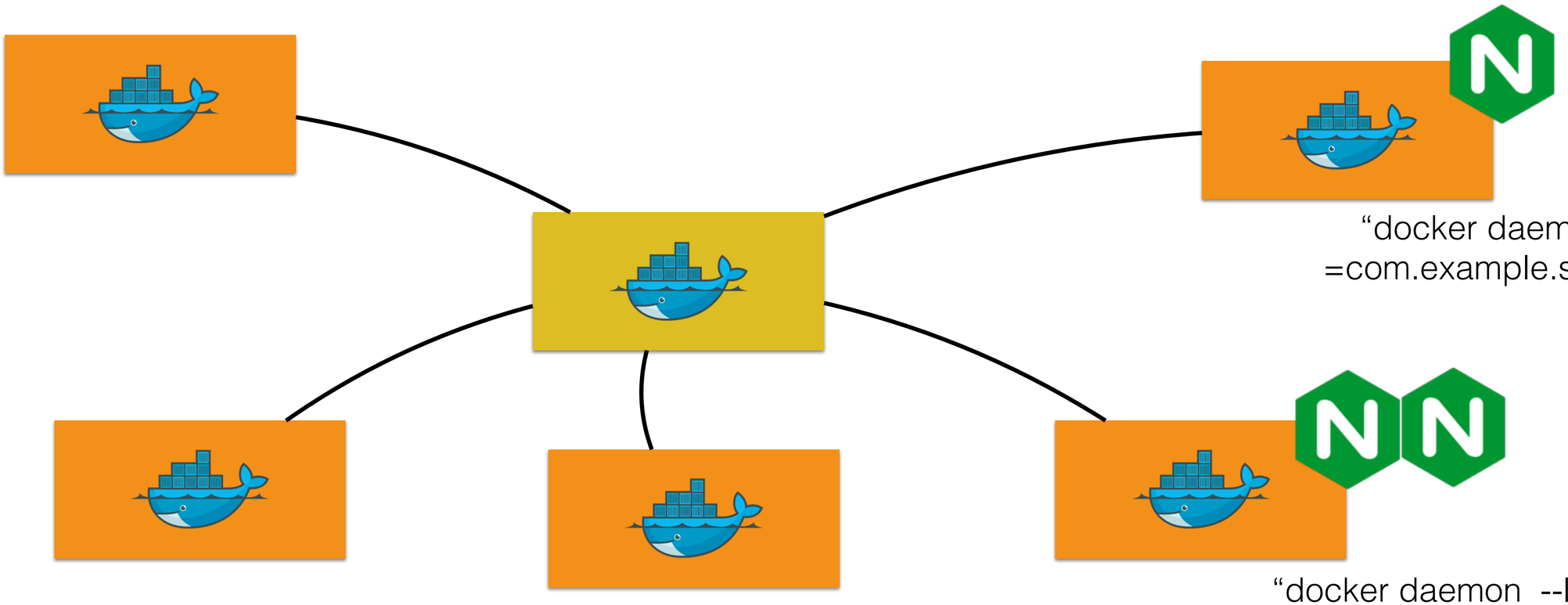


“docker daemon --label =com.example.storage=ssd”

“docker daemon --label =com.example.storage=ssd”

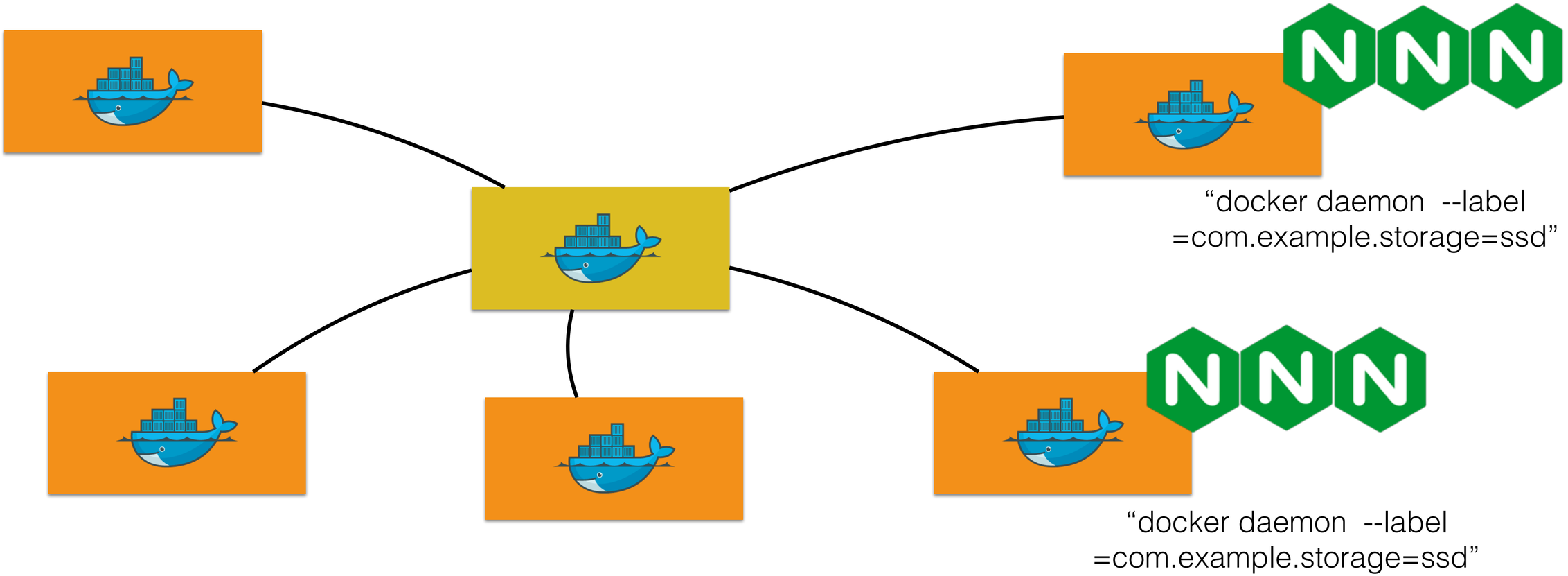
```
DOCKER_OPTS="--label=com.example.storage=ssd"
```

Swarm Mode: Constraints



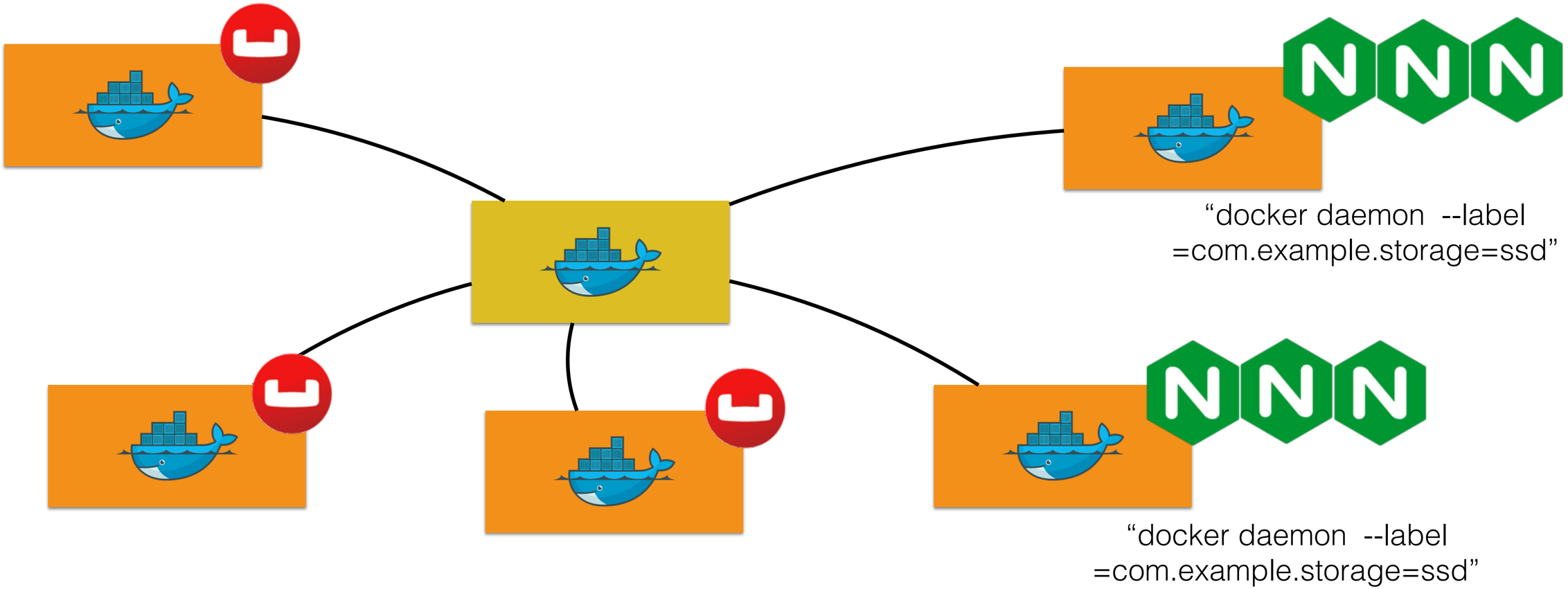

```
docker service create --replicas=3 --name=frontend --
constraint engine.labels.com.example.storage==ssd nginx:latest
```

Swarm Mode: Constraints




```
docker service scale frontend=6
```

Swarm Mode: Constraints




```
docker service create --replicas=3 --name=db couchbase
```

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

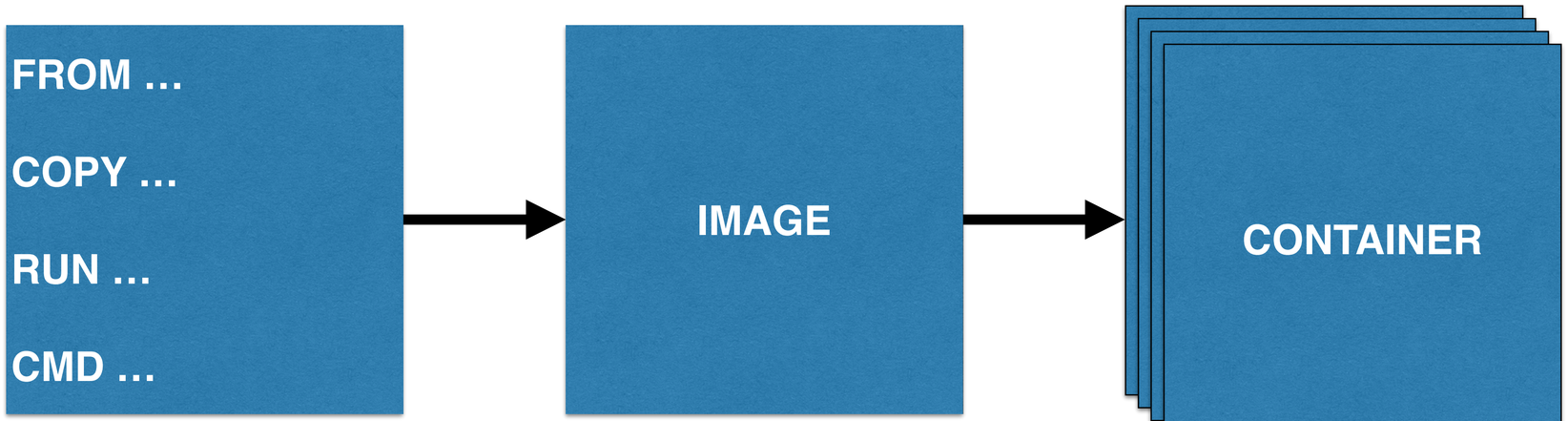
Create a volume

```
docker volume create –name=data data
```

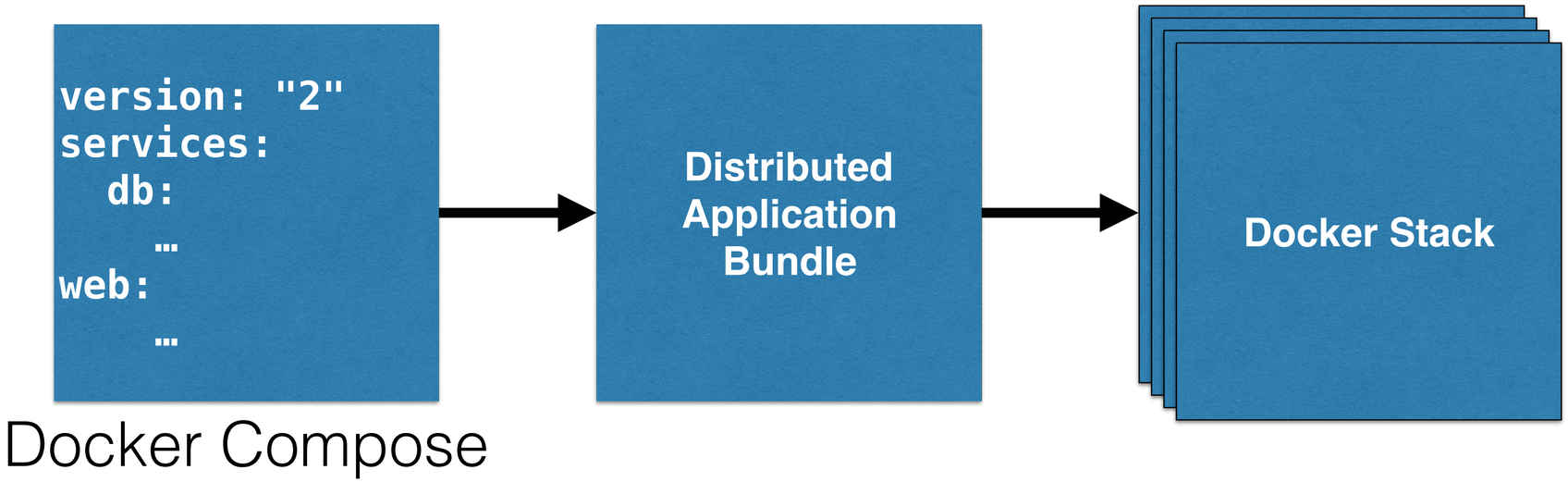
Run a container with the volume

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

Docker Lifecycle



Distributed Application Bundle



Docker Compose

Docker Cloud

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

Docker Cloud

Build

Containerize your applications and accelerate development.

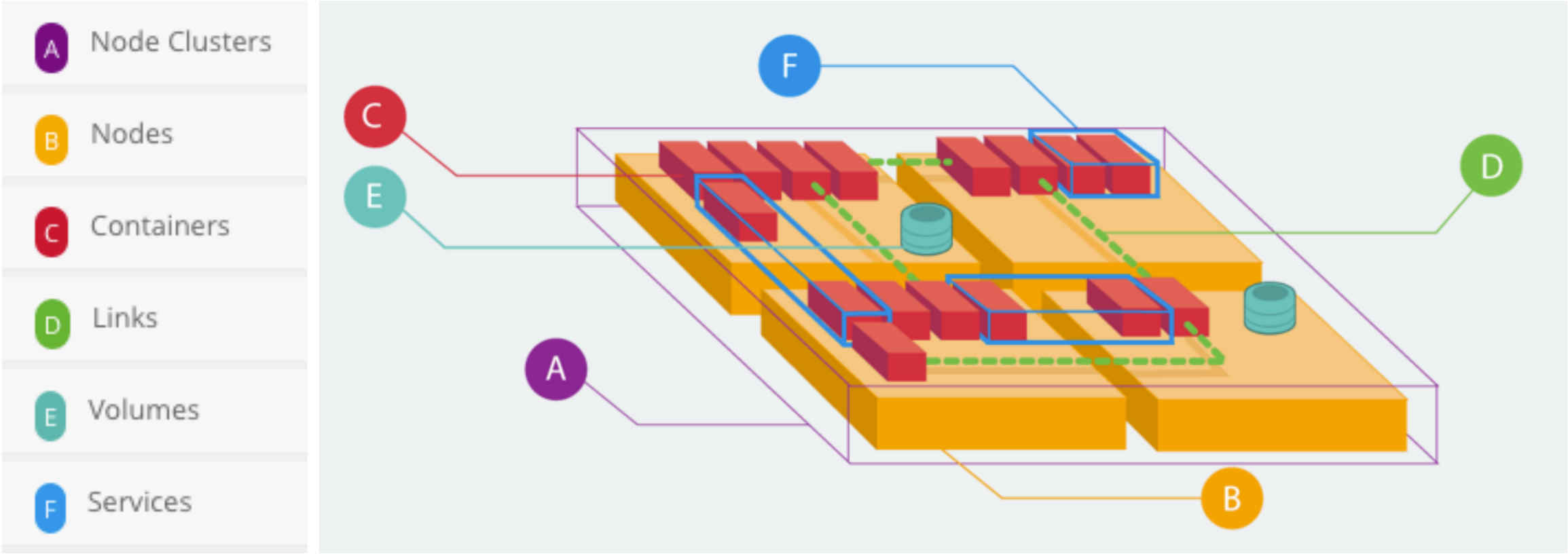
Deploy

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

Manage

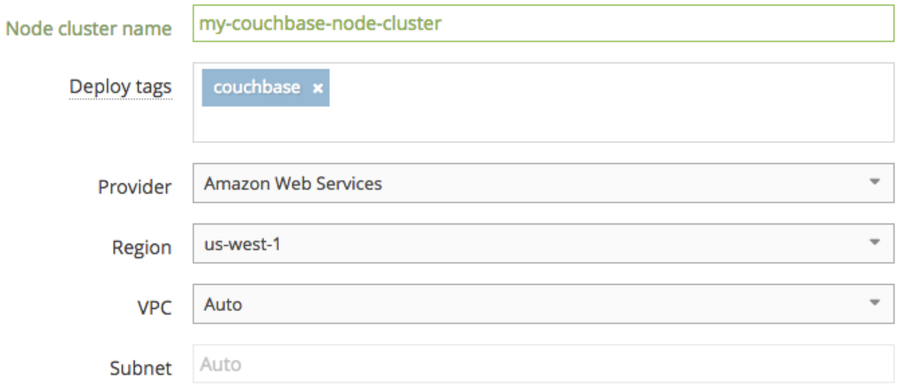
Simplify operations, focus on your code, and forget about managing servers.

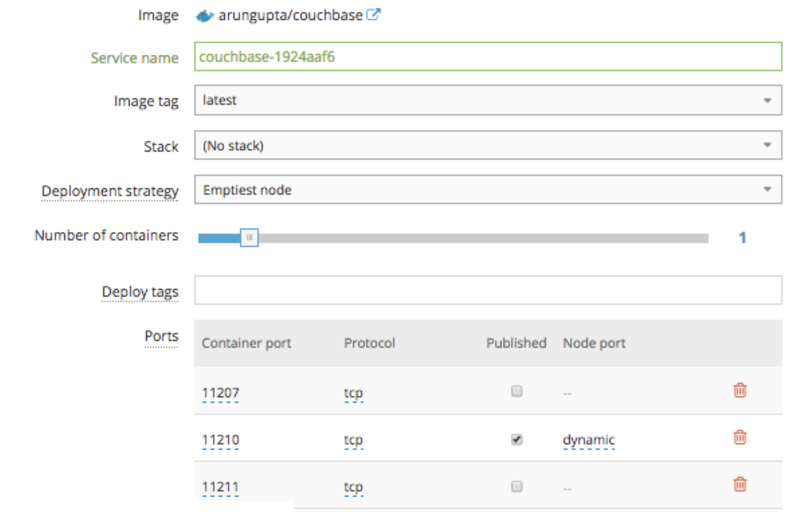
Docker Cloud Architecture

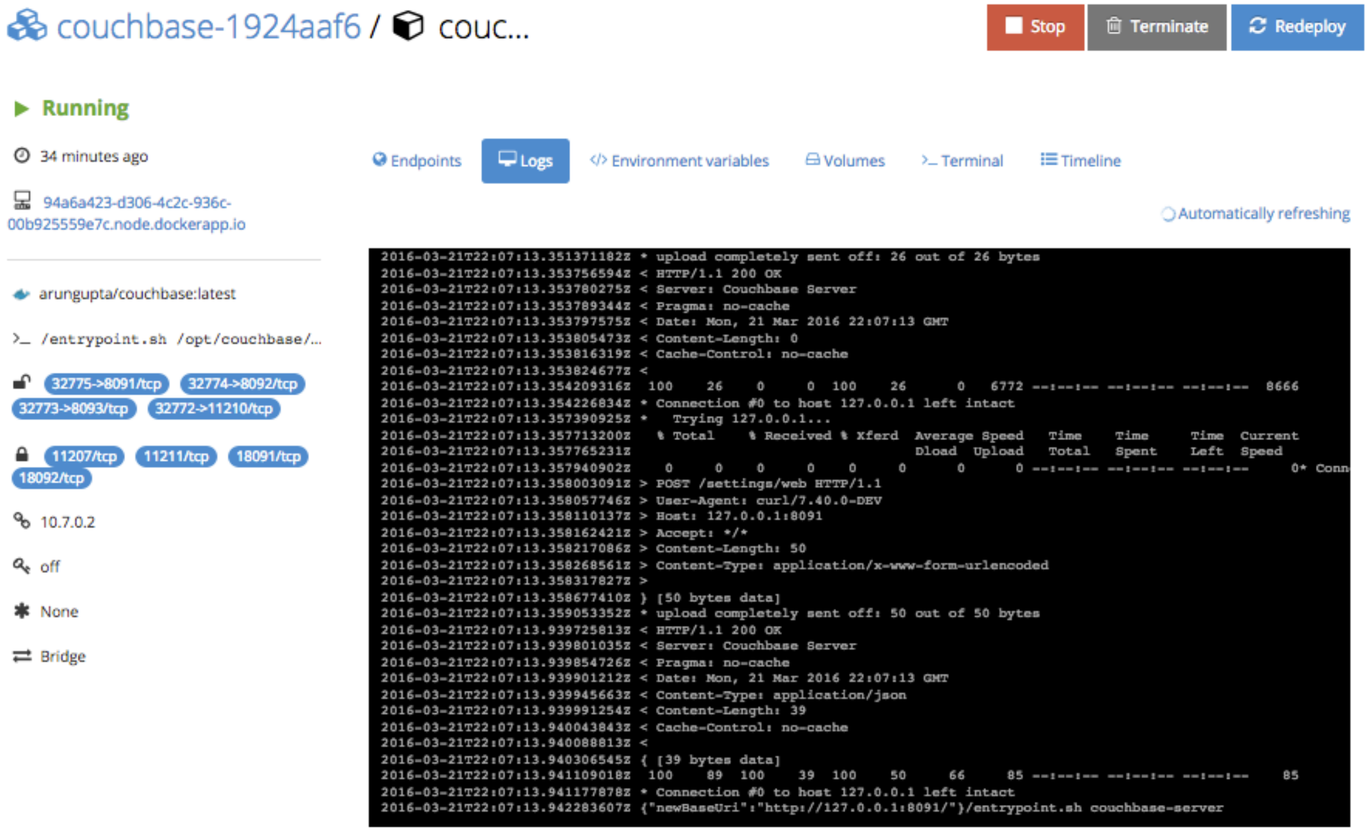


https://www.tutum.co/how/

Create a node cluster







http://blog.couchbase.com/2016/march/getting-started-docker-cloud

Docker Cloud CLI

```
• brew install docker-cloud
• docker-cloud nodecluster create -t 1 --tag couchbase
  couchbase-node aws us-west-1 m3.large
• docker-cloud service create --tag couchbase -p 8091:8091 -p
  8092:8092 -p 8093:8093 -p 11210:11210 arungupta/couchbase
• docker-cloud service start {SERVICE_ID}
• docker-cloud service inspect {SERVICE_ID} | jq
  ".container_ports[0].endpoint_uri" | sed 's/tcp/http/g'
```


http://blog.couchbase.com/2016/march/getting-started-docker-cloud

Docker Registry

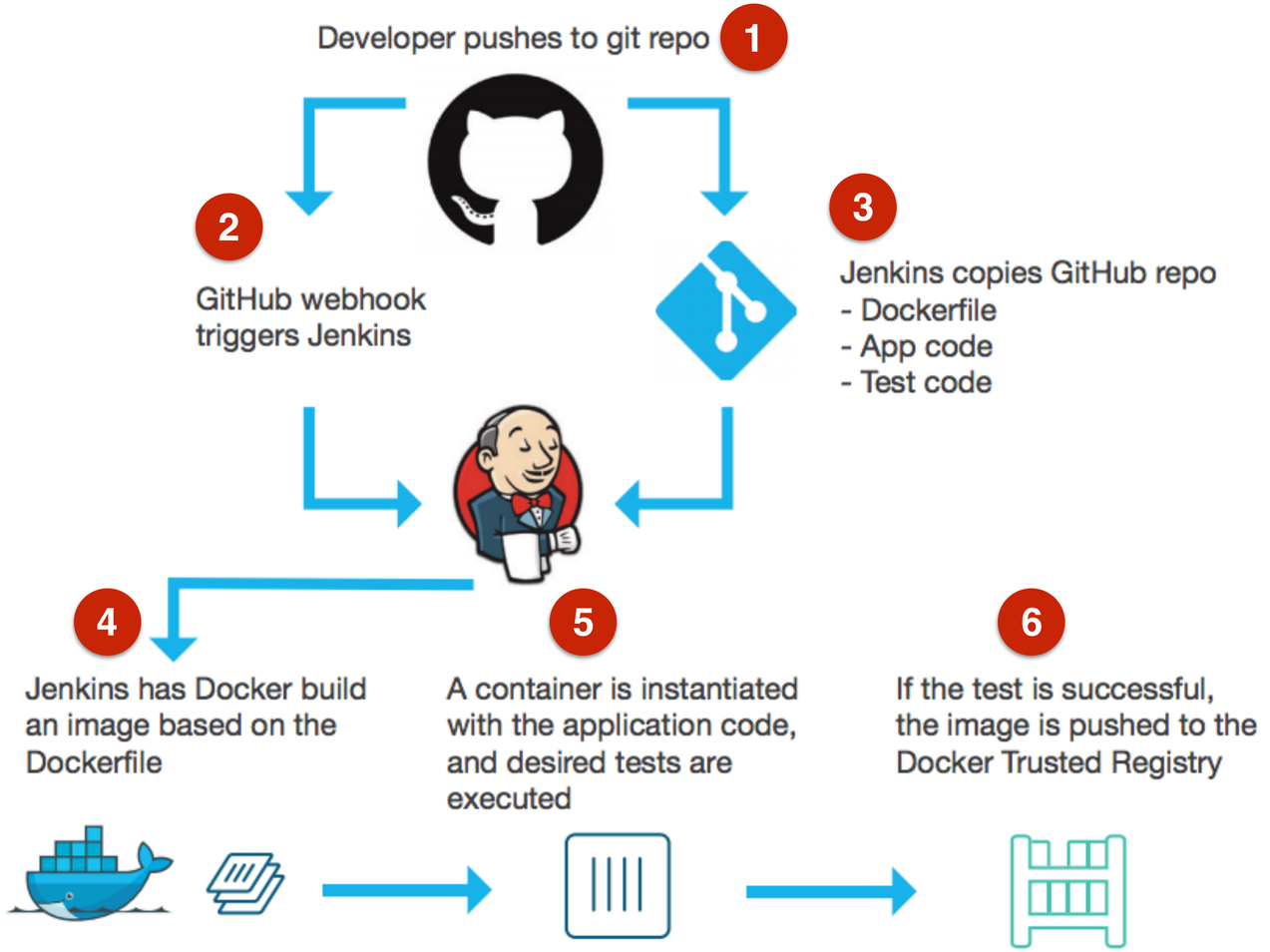
- Store and distribute Docker images
  - Control where images are stored
  - Own image distribution pipeline
  - Integrate image storage/distribution in dev workflow
- Docker Hub
  - Free-to-use and hosted
- Docker Trusted Registry
  - Commercially supported
  - RBAC, LDAP/AD integration, updates, etc

Registry Usage

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

CI/CD with Docker + Jenkins

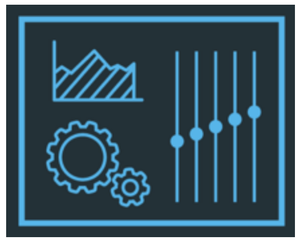
1. Push a commit to GitHub
2. GitHub webhook
3. Jenkins copies artifacts
4. Jenkins builds Docker image
5. Runs test on Docker container
6. Pushes to DTR



https://www.docker.com/sites/default/files/UseCase/RA_CI%20with%20Docker_08.25.2015.pdf

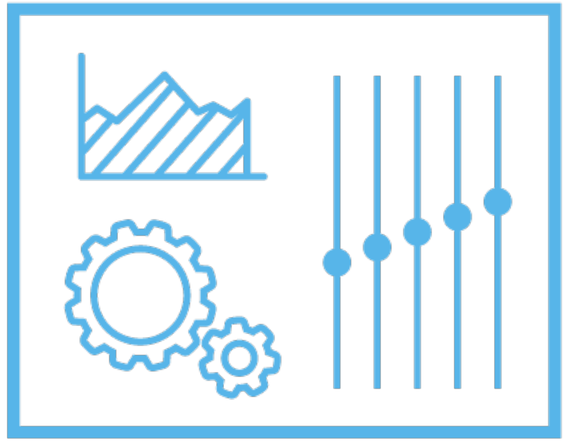
Monitoring Docker Containers

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

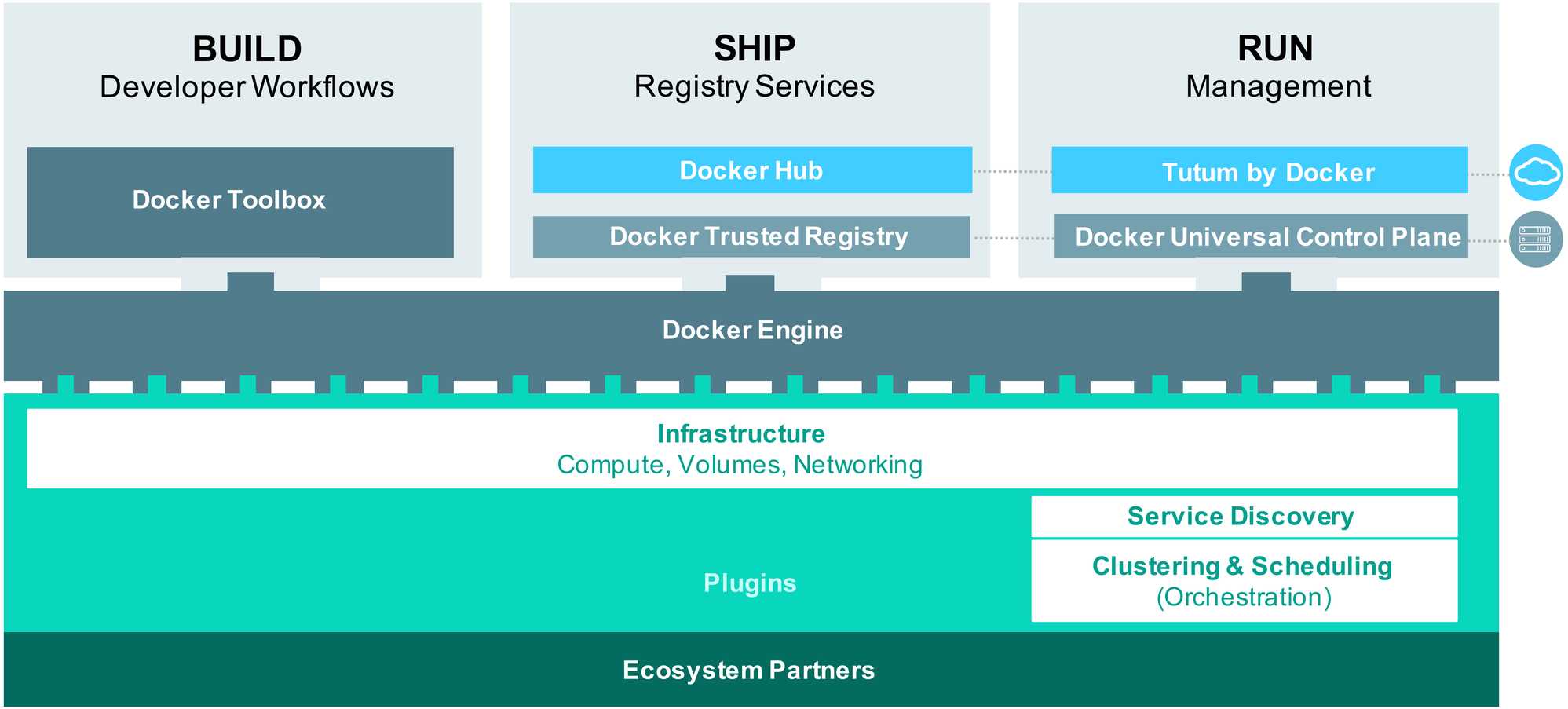


Docker Universal Control Plane

- On-premise management for Docker containers
- Build on Docker Remote API
  - Integrated support for CLI, Swarm, Compose and Trusted Registry
- Integration with LDAP/AD
- Audit Logs



Docker Mission



References

- Docker for Java Tutorial: github.com/docker/labs/tree/master/java
- Docker Documentation: docs.docker.com