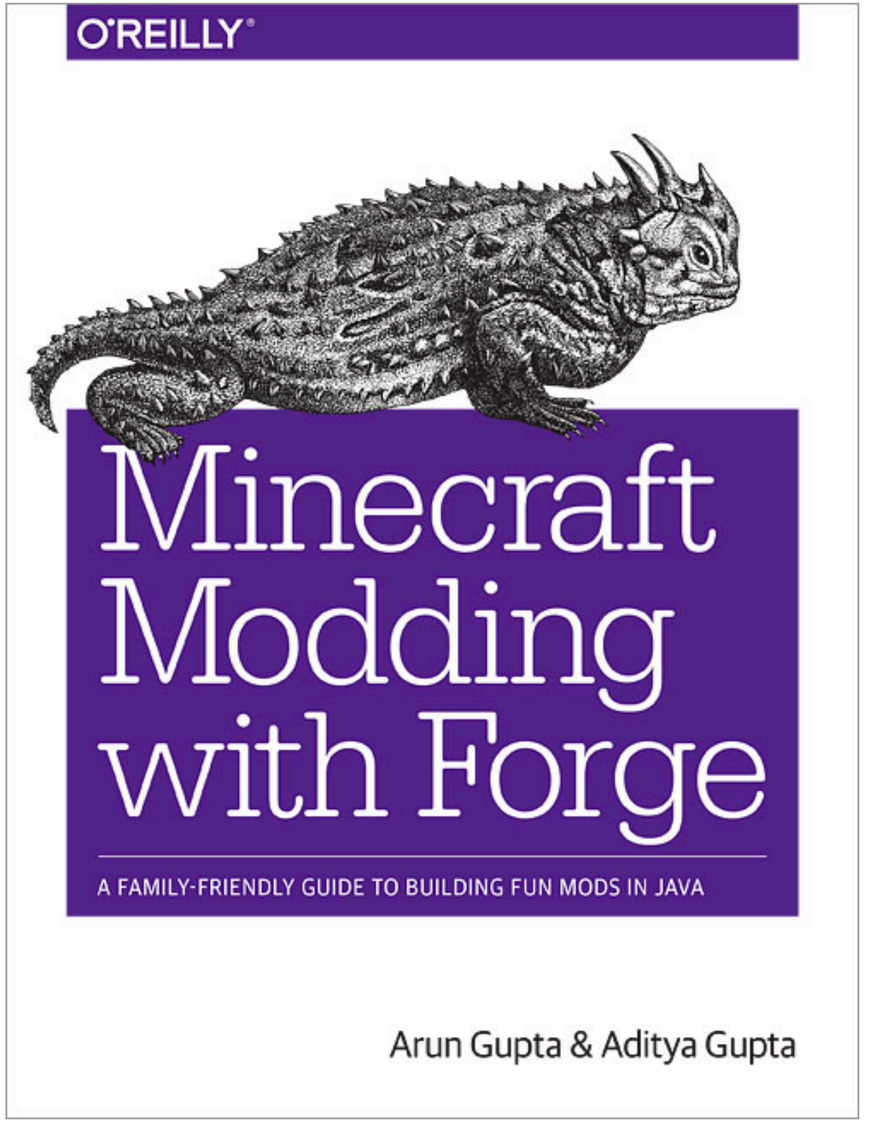
Docker for Java Developers

Arun Gupta, @arungupta
VP Developer Advocacy, Couchbase

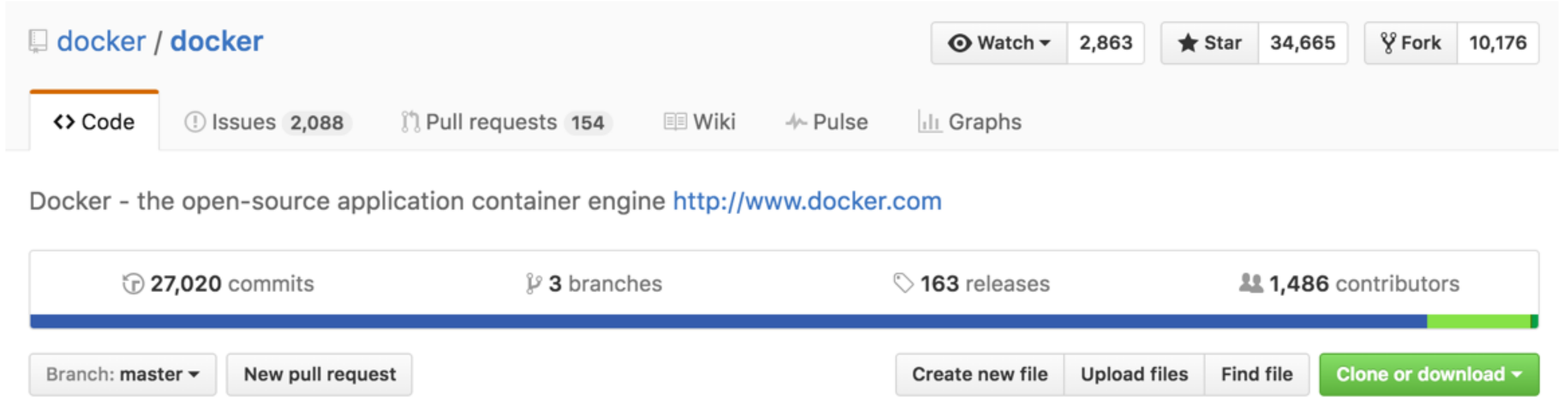
Couchbase

DEVOXX™ 4KIDS USA

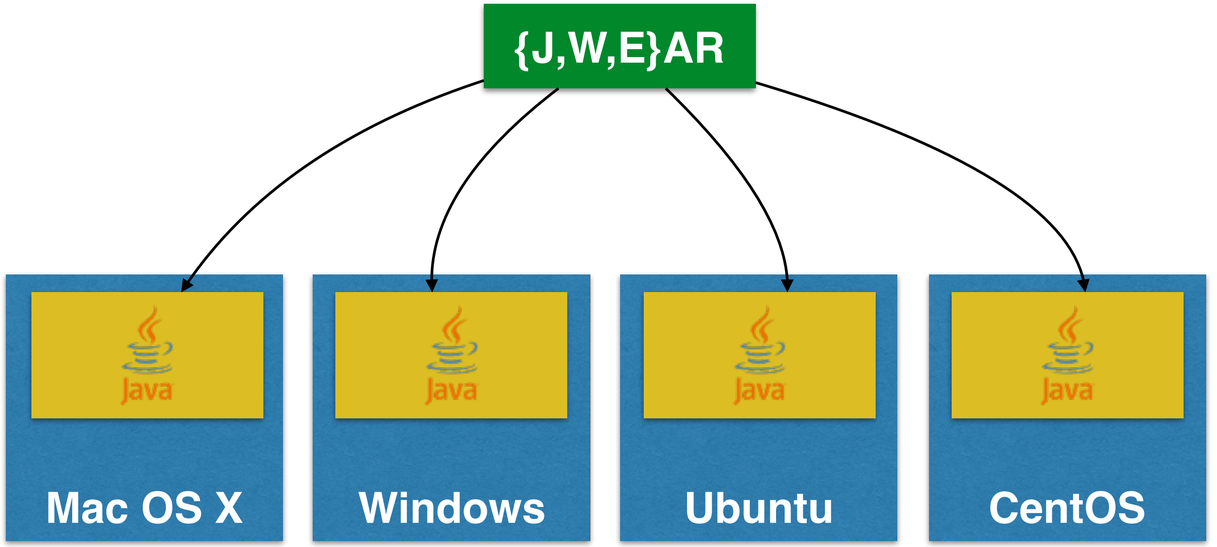


What is Docker?

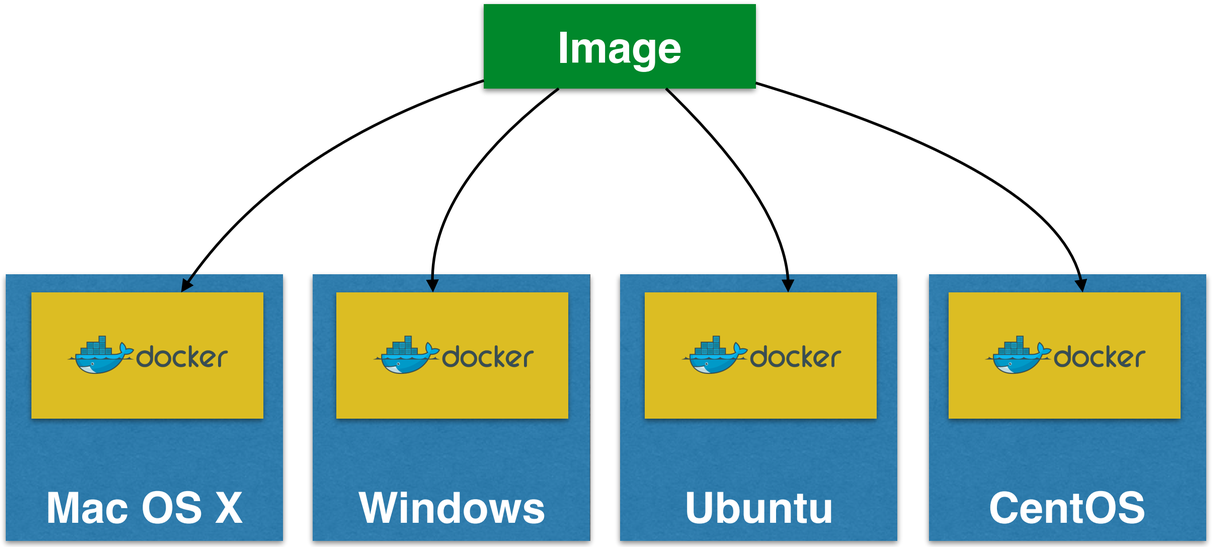
- Open source project and company



- Used to create containers for software applications

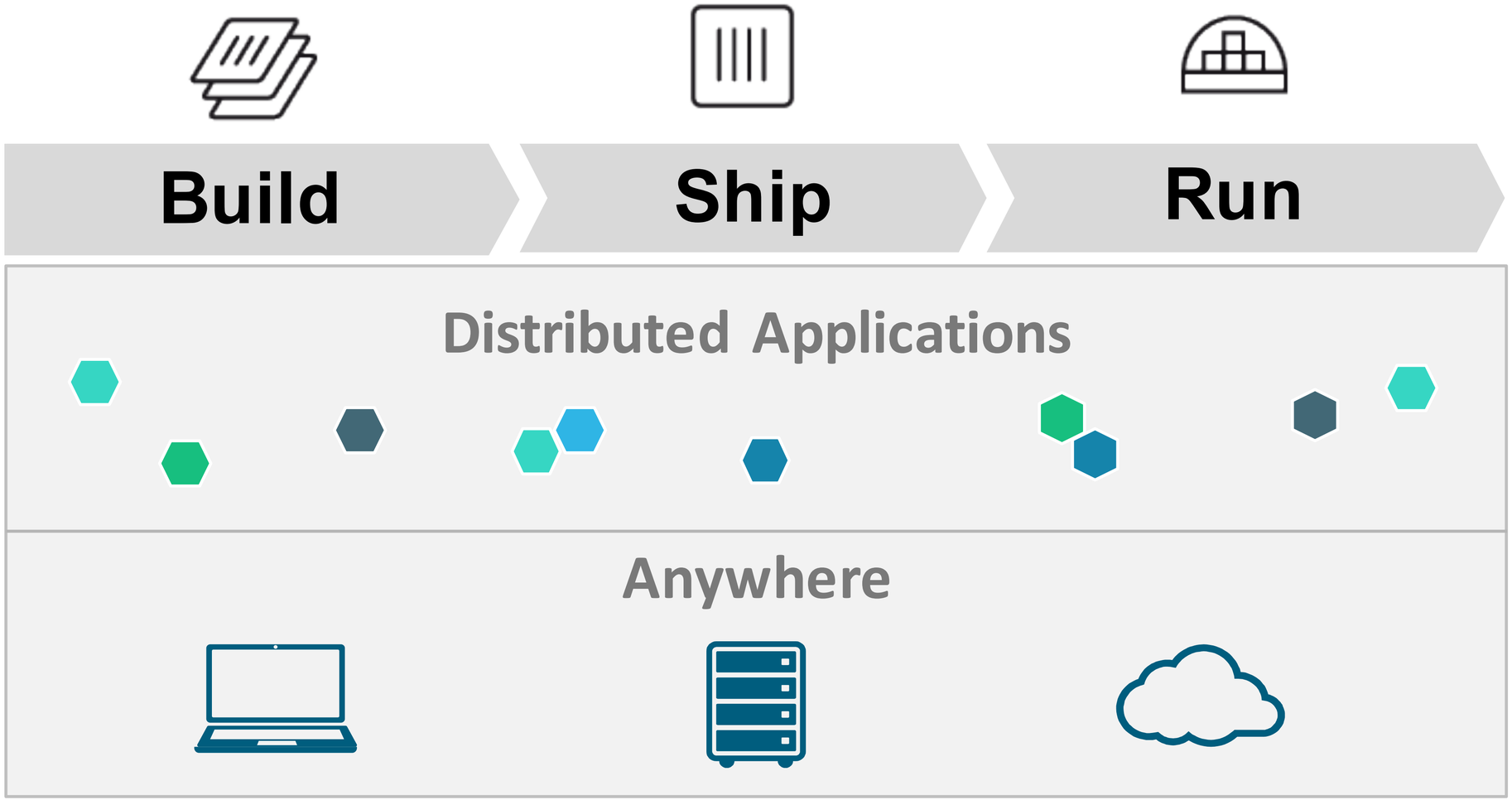


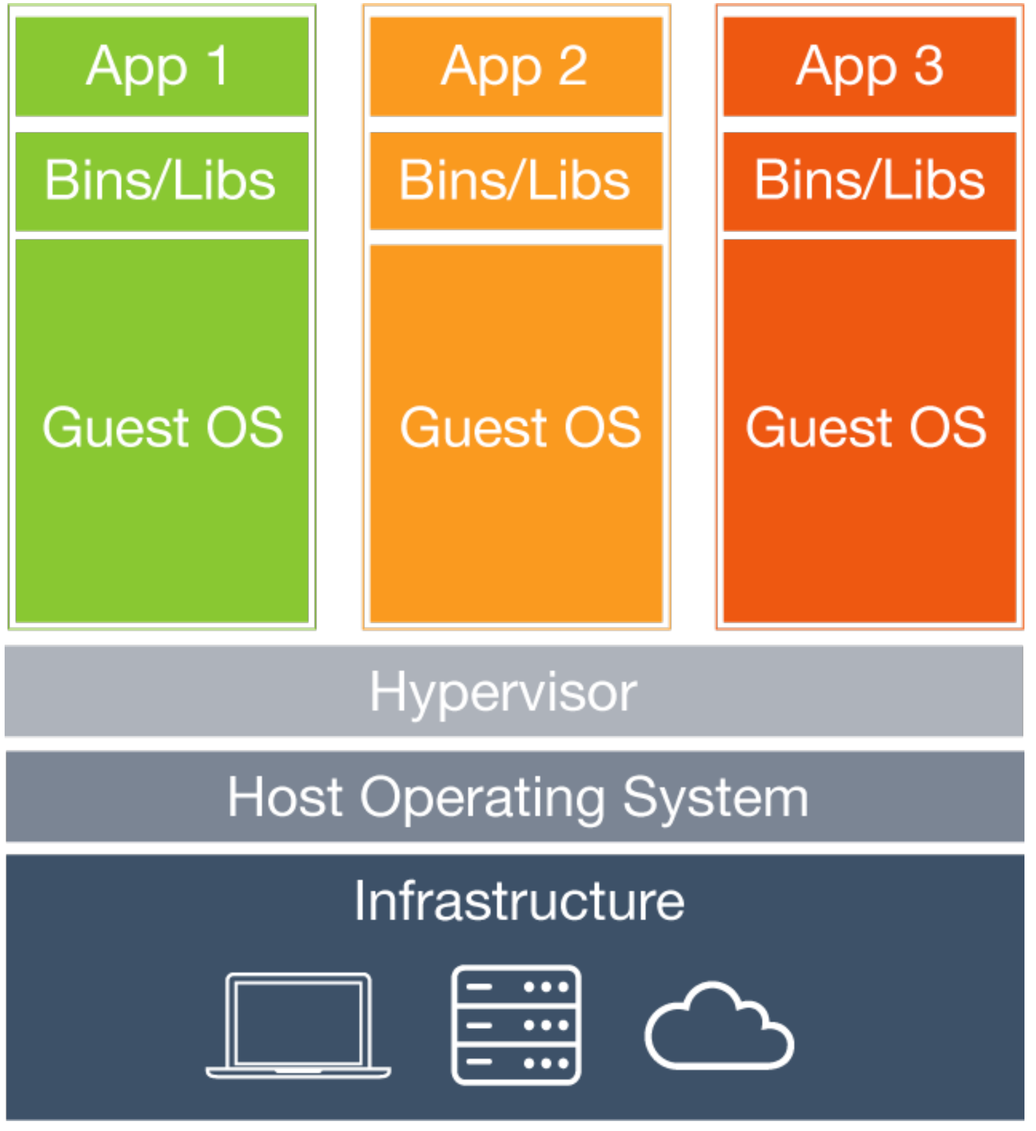
WORA = Write Once Run Anywhere

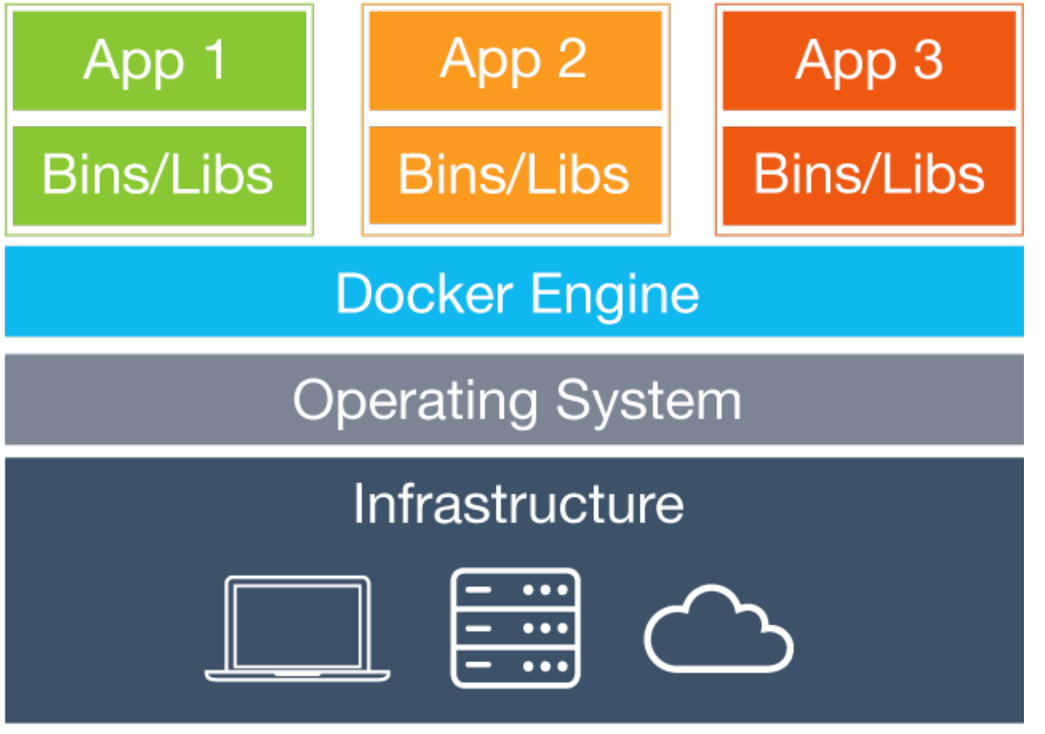


PODA = Package Once Deploy Anywhere

Docker Mission







Build

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

```
FROM ubuntu

CMD echo “Hello world”
```

```
FROM openjdk

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

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

- Usage
- Format
- Parser directives
- escape
- 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
  - Understand how CMD and ENTRYPOINT interact
- VOLUME
- USER
- WORKDIR
- ARG
  - Impact on build caching
- ONBUILD
- STOPSIGNAL
- HEALTHCHECK
- SHELL
- Dockerfile examples

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

Docker Workflow

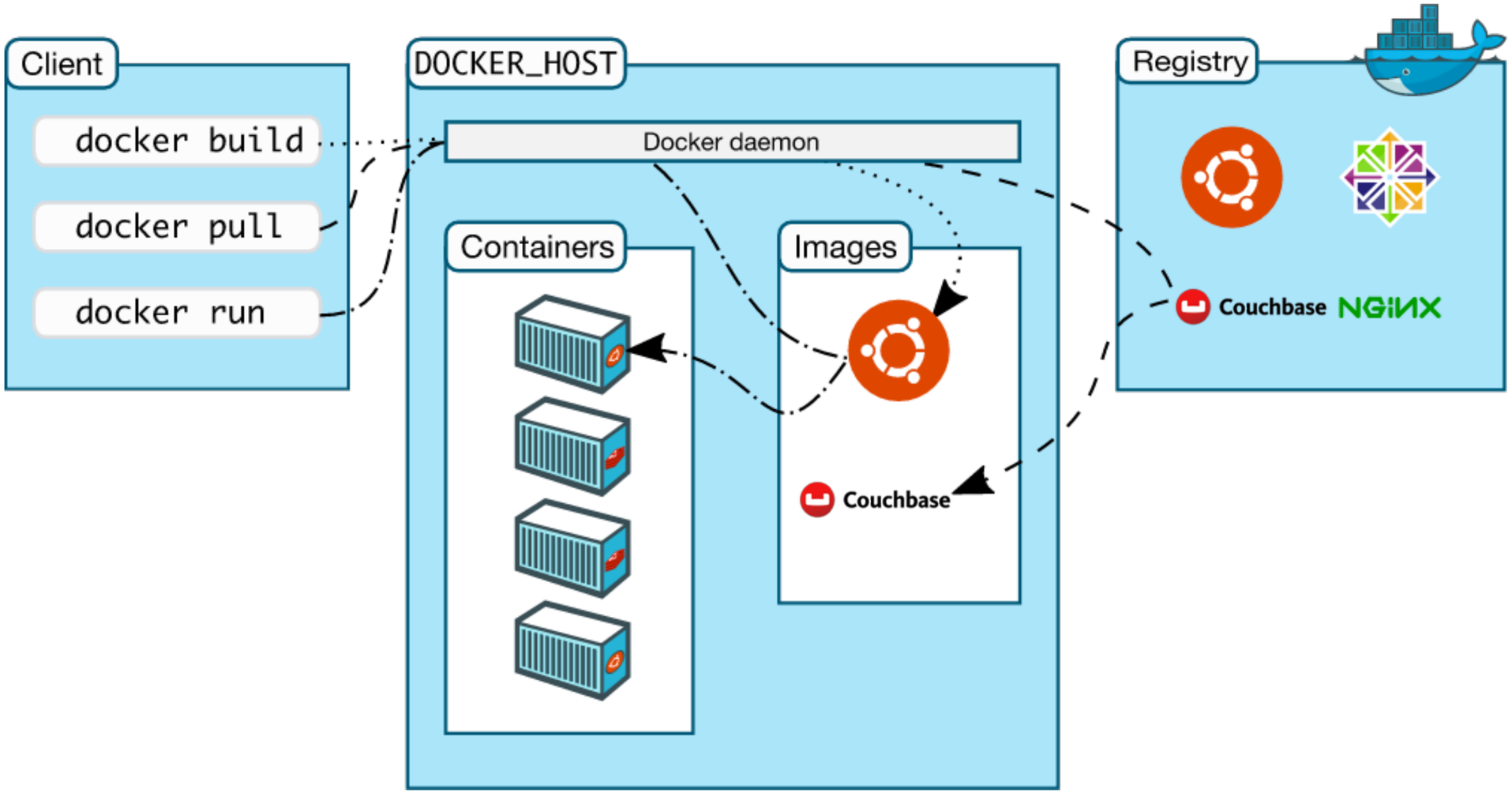
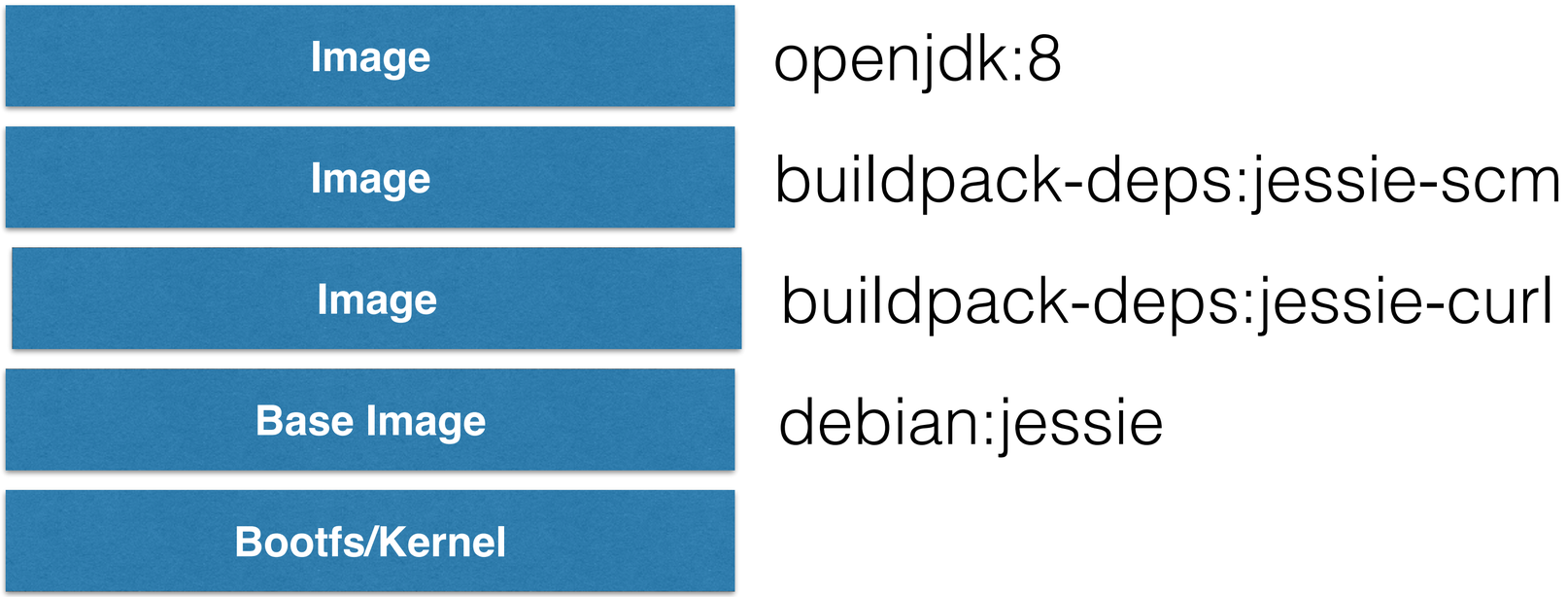


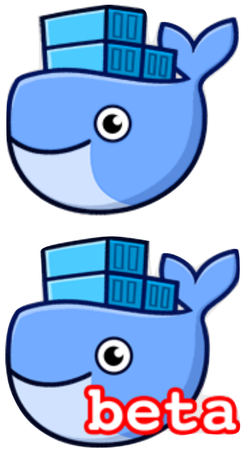
Image Layers - OpenJDK

```
~ > docker images openjdk
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
openjdk             latest              ea40c858f006        13 days ago         643.1 MB
~ > docker history openjdk
IMAGE               CREATED             CREATED BY                                      SIZE
COMMENT
ea40c858f006        13 days ago         /bin/sh -c /var/lib/dpkg/info/ca-certificates   418.2 kB
<missing>           13 days ago         /bin/sh -c set -x  && apt-get update  && apt-   349.3 MB
<missing>           13 days ago         /bin/sh -c #(nop)  ENV CA_CERTIFICATES_JAVA_V   0 B
<missing>           13 days ago         /bin/sh -c #(nop)  ENV JAVA_DEBIAN_VERSION=8u   0 B
<missing>           13 days ago         /bin/sh -c #(nop)  ENV JAVA_VERSION=8u102       0 B
<missing>           13 days ago         /bin/sh -c #(nop)  ENV JAVA_HOME=/usr/lib/jvm   0 B
<missing>           13 days ago         /bin/sh -c {   echo '#!/bin/sh';   echo 'set    87 B
<missing>           13 days ago         /bin/sh -c #(nop)  ENV LANG=C.UTF-8             0 B
<missing>           13 days ago         /bin/sh -c echo 'deb http://httpredir.debian.   61 B
<missing>           13 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    44.31 MB
<missing>           2 weeks ago         /bin/sh -c #(nop)  CMD ["/bin/bash"]            0 B
<missing>           2 weeks ago         /bin/sh -c #(nop) ADD file:f2453b914e7e026efd   125.1 MB
```

Union File System



Docker for Mac/Windows



- Native application and UI
- Auto update capability
- No additional software required, e.g. VirtualBox
  - OSX: xhyve VM using **`Hypervisor.framework`**
  - Windows: Hyper-V VM
- Download: docker.com/getdocker
- Requires Yosemite 10.10+ or Windows 10 64-bit

Docker for AWS/Azure

- Amazon Web Services
  - Amazon CloudFormation templates
  - Integrated with Autoscaling, ELB, and EBS
- Azure
  - Integrated with VM Scale Sets for autoscaling, Azure Load Balancer, Azure Storage
- beta.docker.com (restricted availability)

Create stack

- Select Template
- Specify Details
- Options
- **Review**

Review

Template

Template URL	https://docker-for-aws.s3.amazonaws.com/aws/beta/aws-v1.12.1-beta5.json
Description	Docker for AWS 1.12.1 (beta5)
Estimate cost	Cost

Details

Stack name	Docker

Swarm Size

ManagerSize	1
ClusterSize	3

Swarm Properties

ManagerInstanceType	t2.micro
InstanceType	m3.medium
KeyName	arun@couchbase

Options

Tags

No tags provided

Advanced

Notification	
Timeout	none
Rollback on failure	Yes

Docker Toolbox

- Docker Engine
- Docker Machine
- Docker Compose
- Docker Kitematic
- Virtual box
- Quickstart Terminal

https://www.docker.com/toolbox

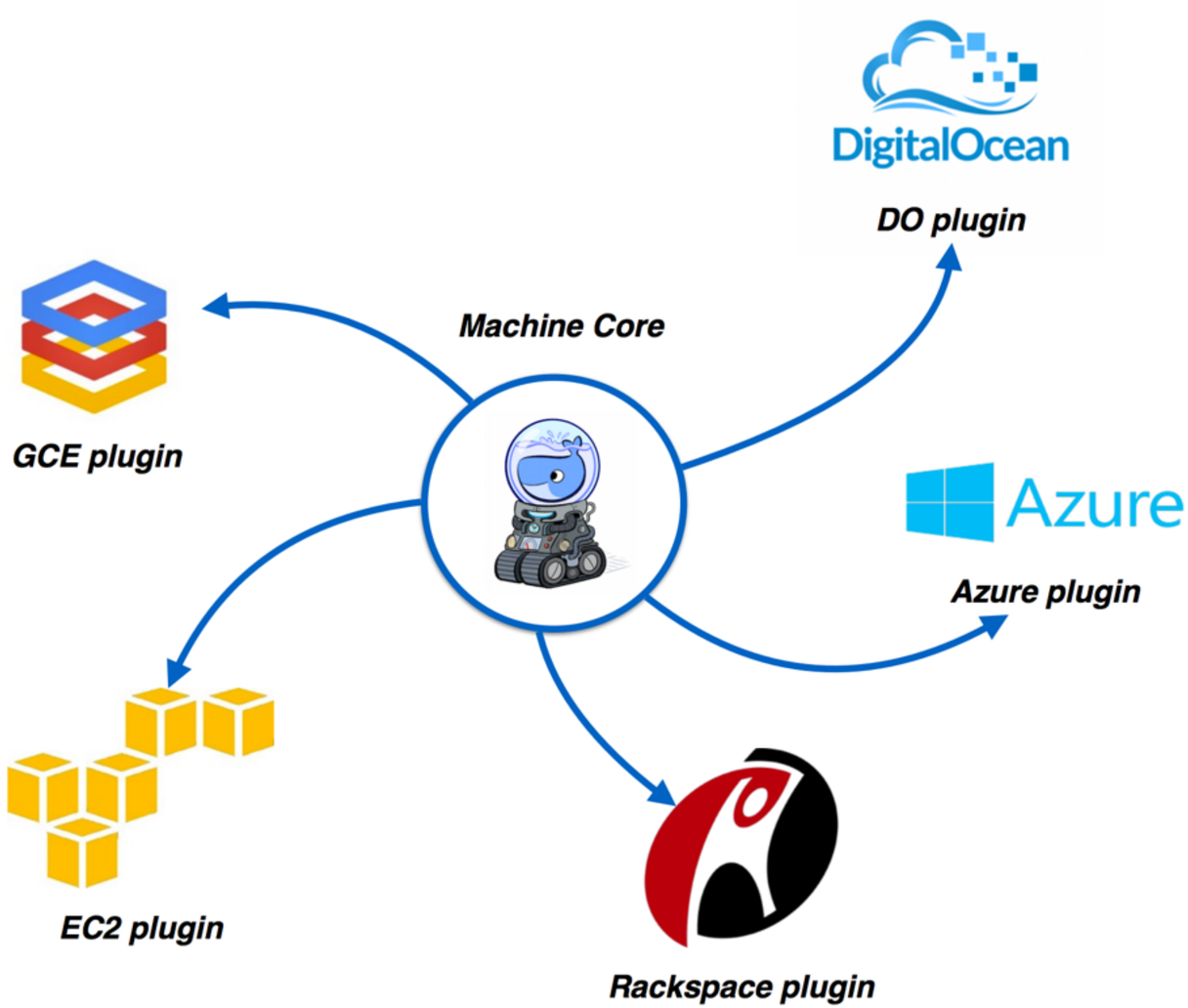
Docker Machine

- 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

Docker Machine Providers



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 up -d
```

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:
    - 80:8080
db-prod:
  image: . . .
```

production.yml

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

Docker Compose Common Use Cases

Use Case	Command
Dev Setup	`docker-compose up`
Local/remote host	`DOCKER_HOST, DOCKER_TLS_VERIFY, DOCKER_CERT_PATH`
Single/multiple hosts	Integrated with Swarm
Multiple isolated environments	`docker-compose up -p <project>`

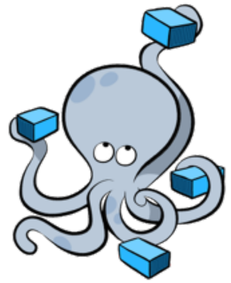
Automated test setup	`docker-compose up`
	`mvn test`
	`docker-compose down`

Dev/Prod Impedance mismatch	`docker-compose up -f docker-compose.yml -f production.yml`

Java Opinionated Container Stack (JOCS)

MySQL

Minikube

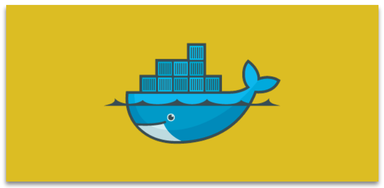


Gradle

Swarm Mode

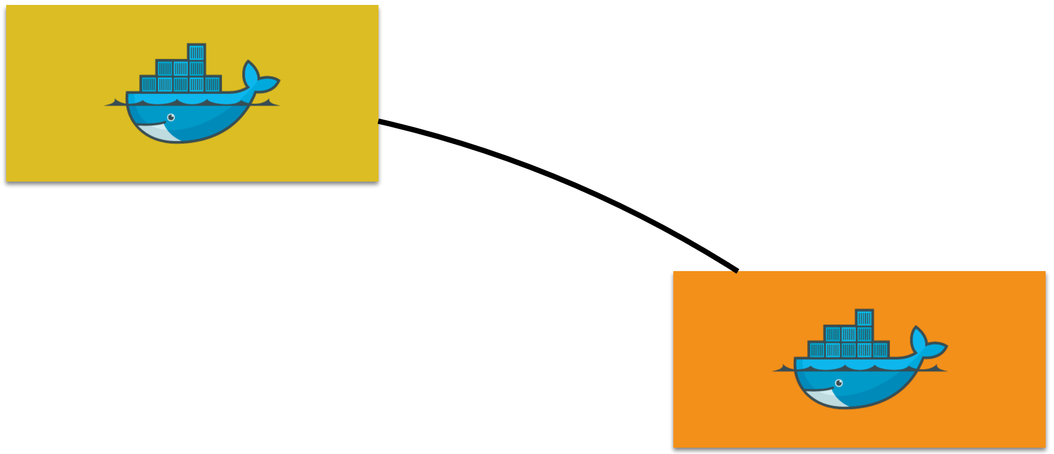
- New in 1.12
- Natively managing a cluster of Docker Engines called a Swarm
- Docker CLI to create a swarm, deploy apps, and manage swarm
- No Single Point of Failure (SPOF)
- Declarative state model
- Self-organizing, self-healing
- Service discovery, load balancing and scaling
- Rolling updates
- Optional feature, need to be explicitly enabled

Swarm Mode: Initialize



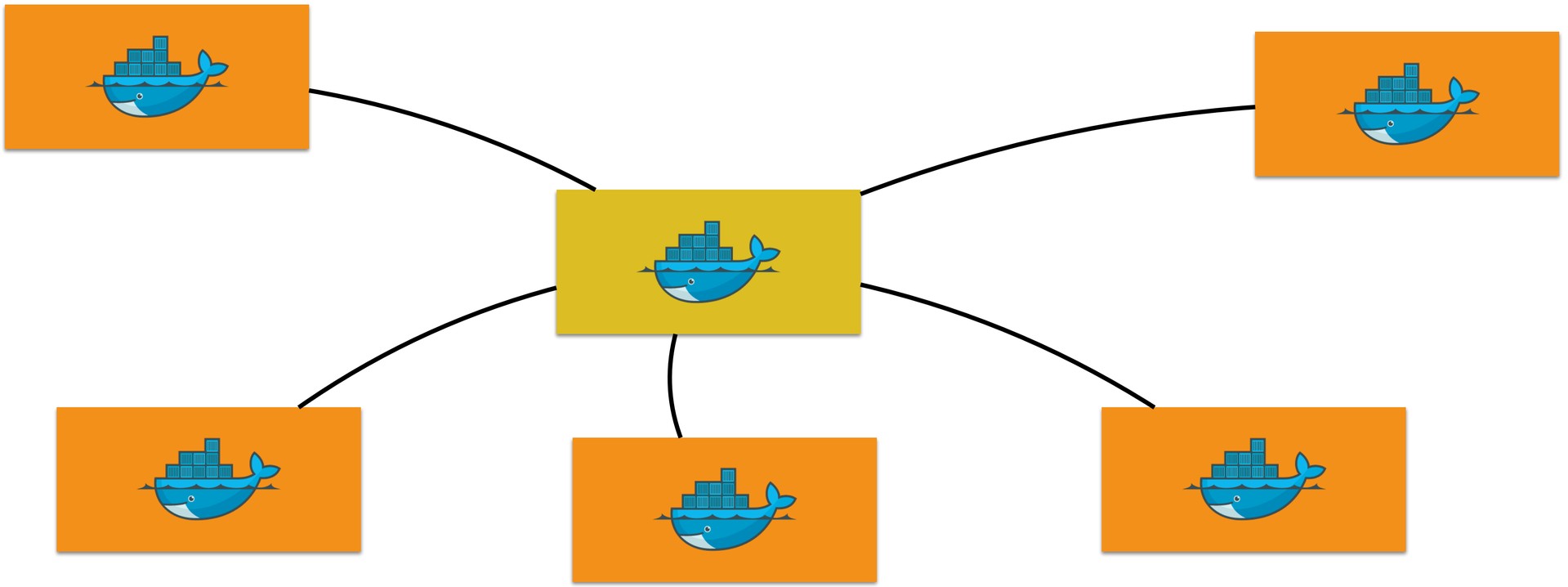
```
docker swarm init --listen-addr <ip>:2377
```

Swarm Mode: Add Worker



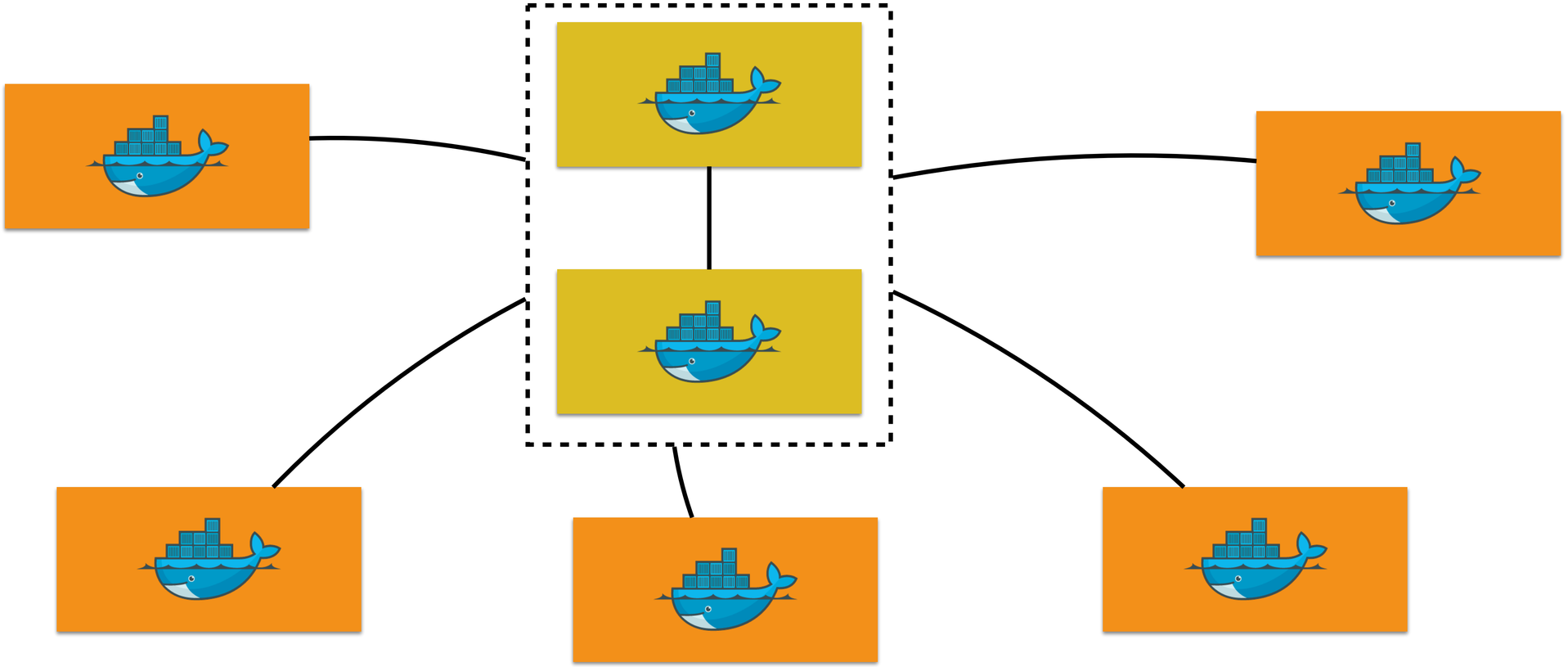
```
docker swarm join --token <worker_token> <manager>:2377
```

Swarm Mode: Add More Workers



```
docker swarm join --token <worker_token> <manager>:2377
```

Swarm Mode: Primary/Secondary Master



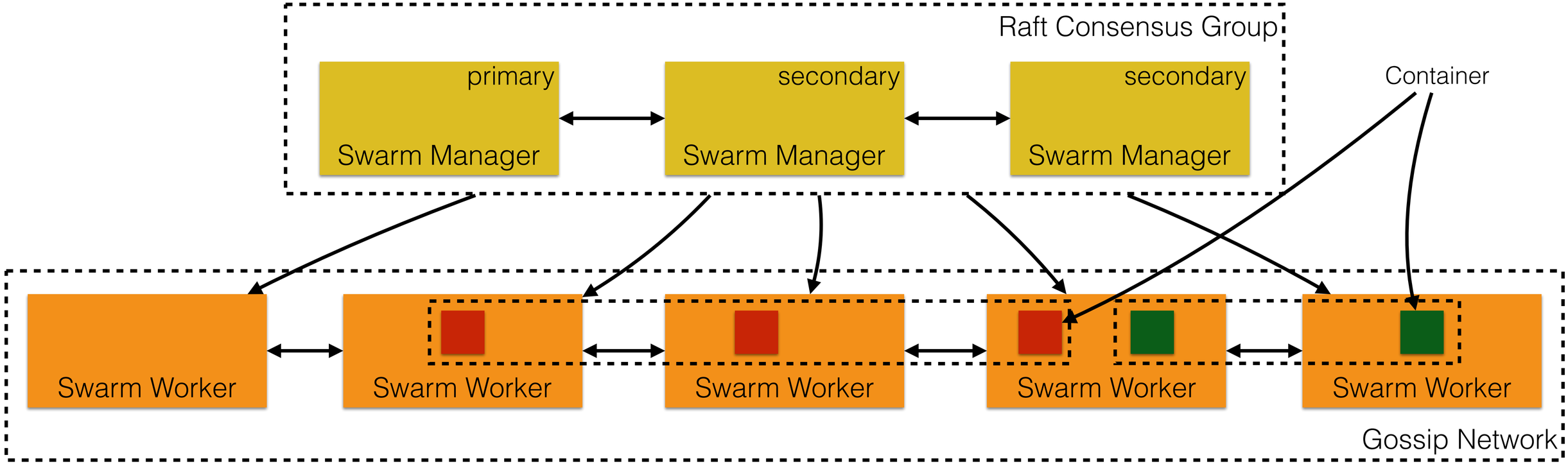
```
docker swarm join --manager --token <manager_token> --listen-
addr <master2>:2377 <master1>:2377
```

Swarm Mode using Docker Machine

Task	Command
Create manger	`docker-machine create -d virtualbox managerX`
Create worker	`docker-machine create -d virtualbox workerX`
Initialize Swarm mode	`docker swarm init --listen-addr <ip1> --advertise-addr <ip1>`
Manager token	`docker swarm join-token manager -q`
Worker token	`docker swarm join-token worker -q`
Manager X join	`docker swarm join --token manager_token --listen-addr <ipX> --advertise-addr <ipX> <ip1>`
Worker X join	`docker swarm join --token worker_token --listen-addr <ipX> --advertise-add <ipX> <ip1>`

https://github.com/docker/labs/blob/master/swarm-mode/quickstart/buildswarm-node-vbox-setup.sh

Swarm Mode: Protocols

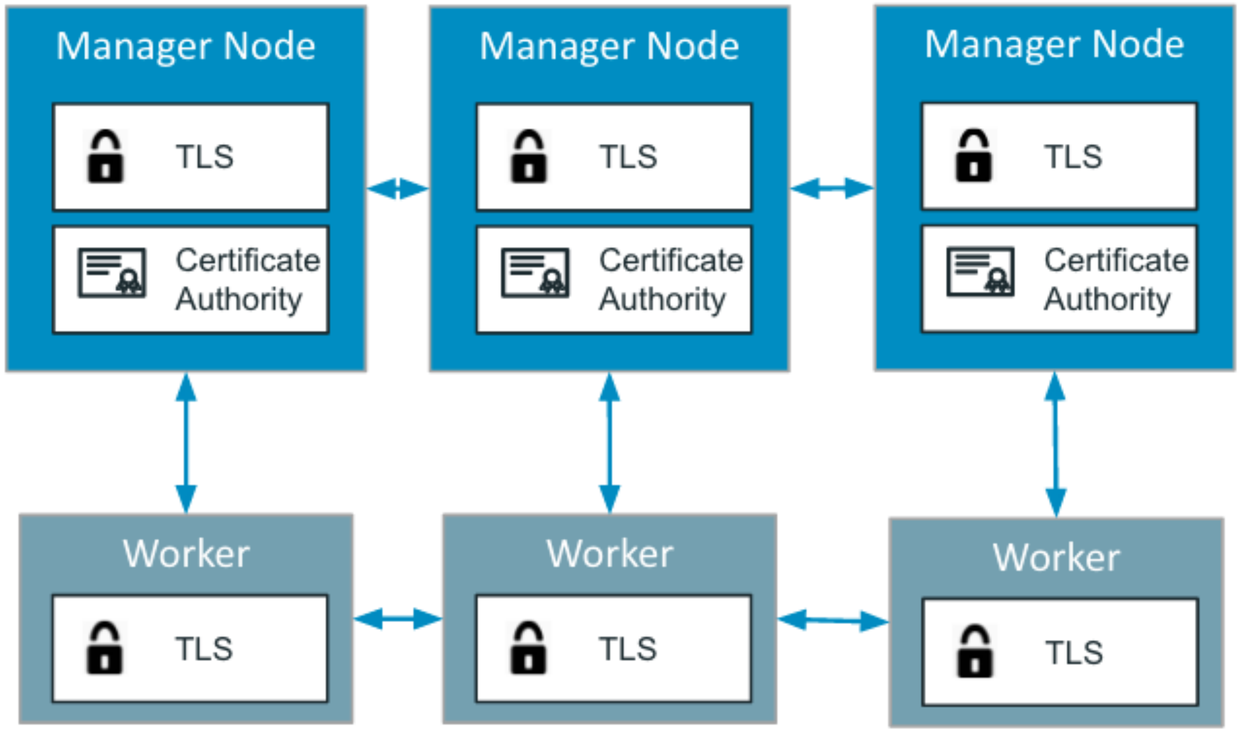


Strongly consistent
Replicated (Raft based)
Extremely fast (in-memory reads)

Swarm Mode in Production

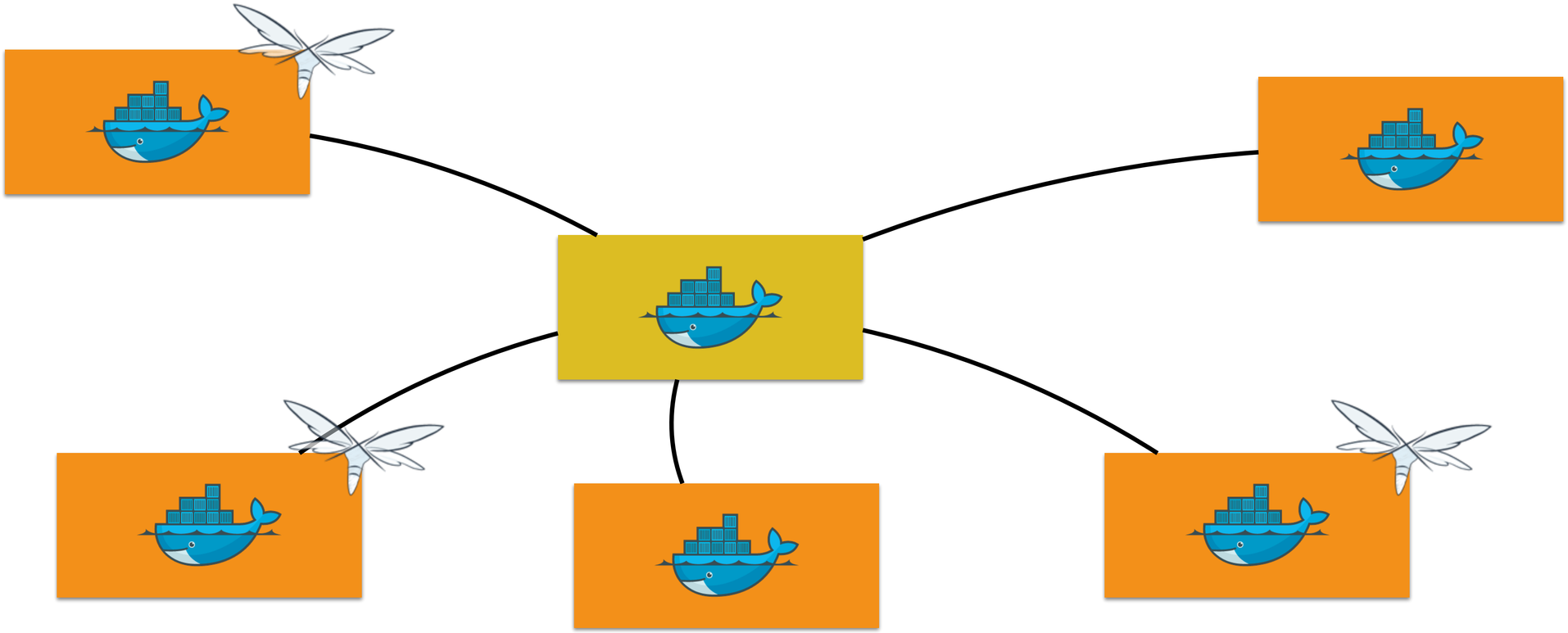
https://blog.online.net/2016/07/29/docker-swarm-an-analysis-of-a-very-large-scale-container-system/

Secure by Default



- Cryptographic node identity
- Automatic encryption and mutual authentication (TLS)
- Automatic cert rotation (90 days, can be up to 30 mins)
- External CA integration

Swarm Mode: Replicated Service

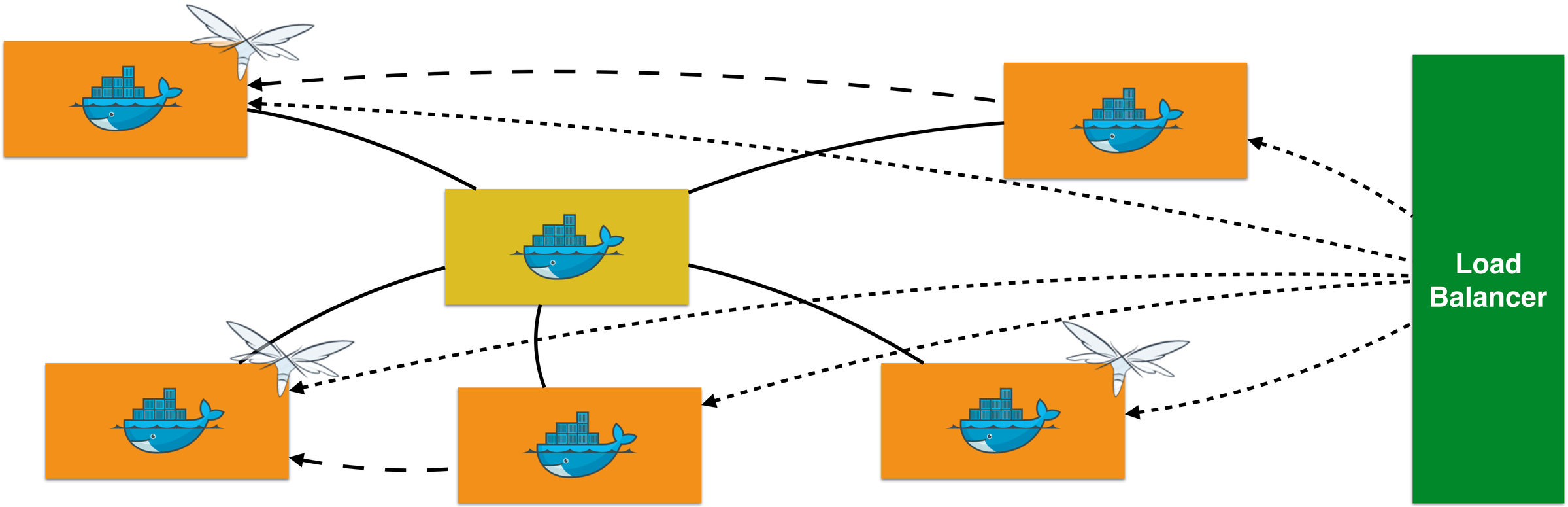


```
docker service create --replicas 3 --name web jboss/wildfly
```

Swarm Mode - Routing Mesh

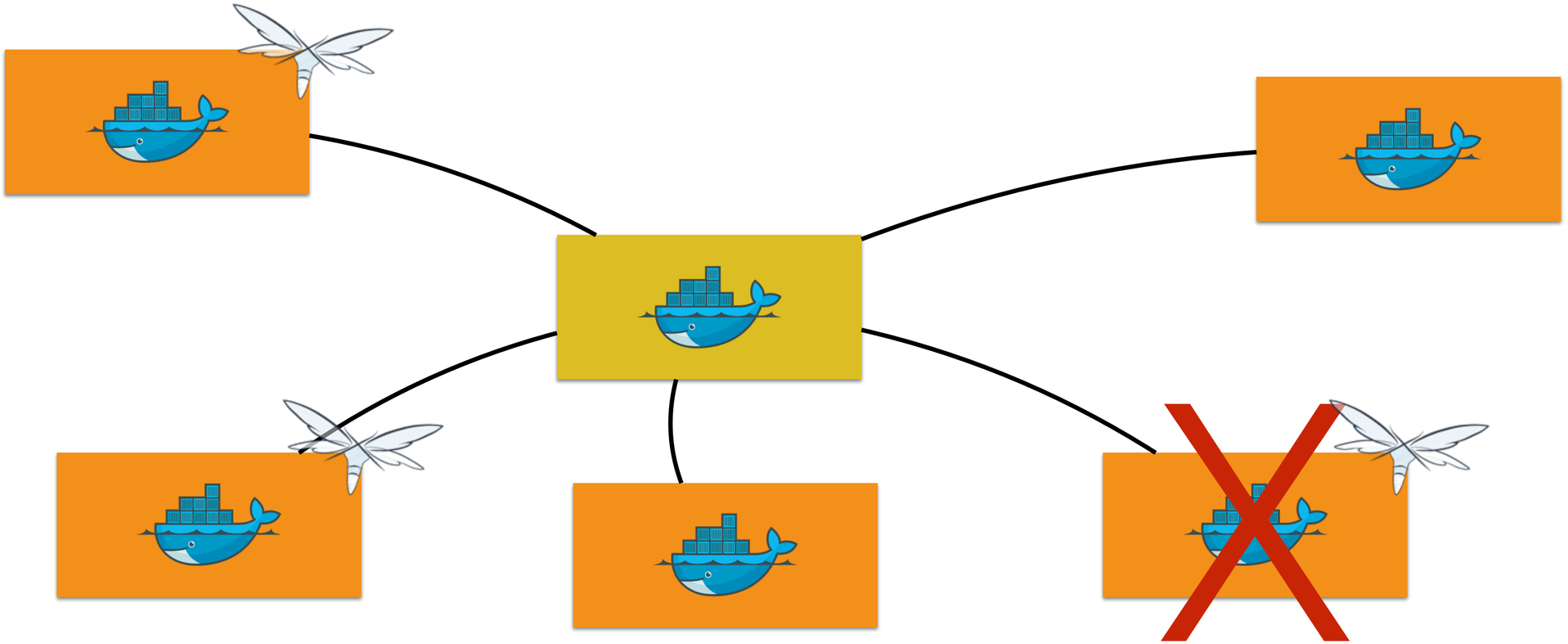
- Load balancers are host-aware, not container-aware
- Swarm mode introduces container-aware routing mesh
- Reroutes traffic from any host to a container
  - Reserves a Swarm-wide ingress port
  - Uses DNS-based service discovery

Swarm Mode: Routing Mesh

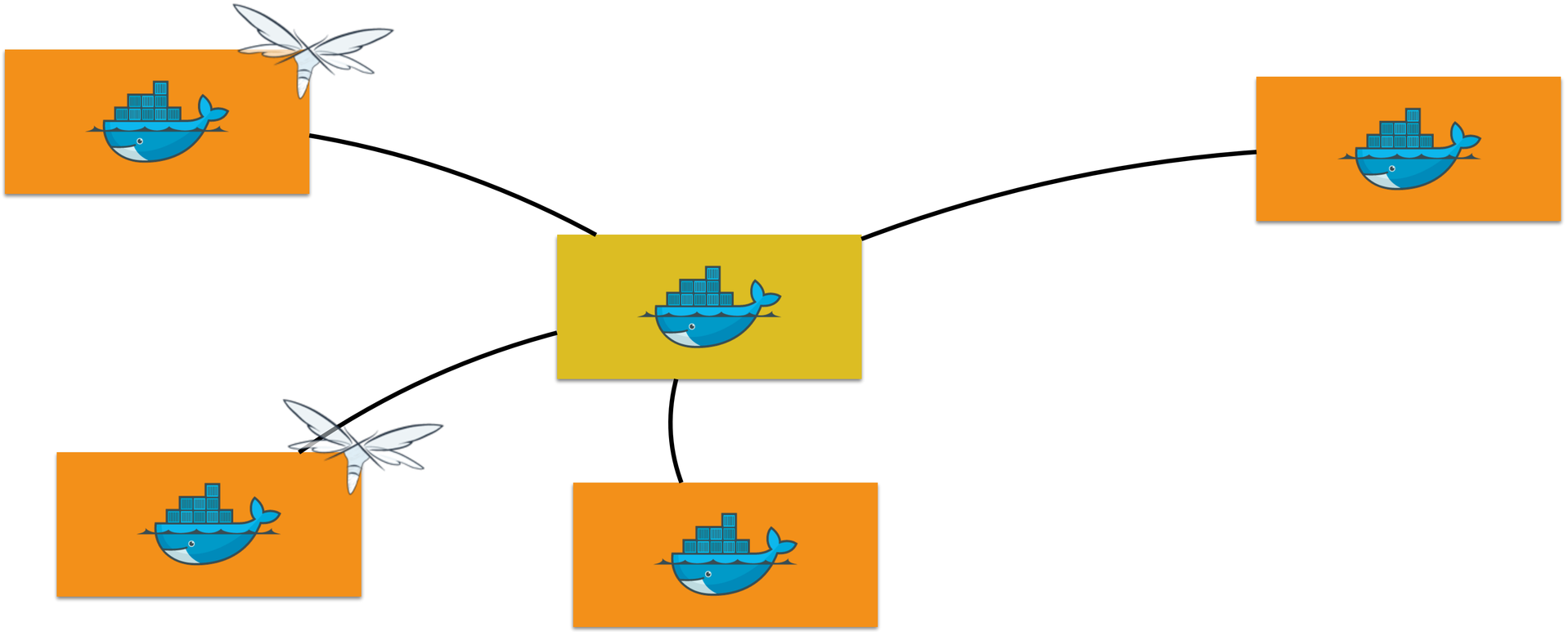



```
docker service create --replicas 3 --name web -p 8080:8080 jboss/
wildfly
```

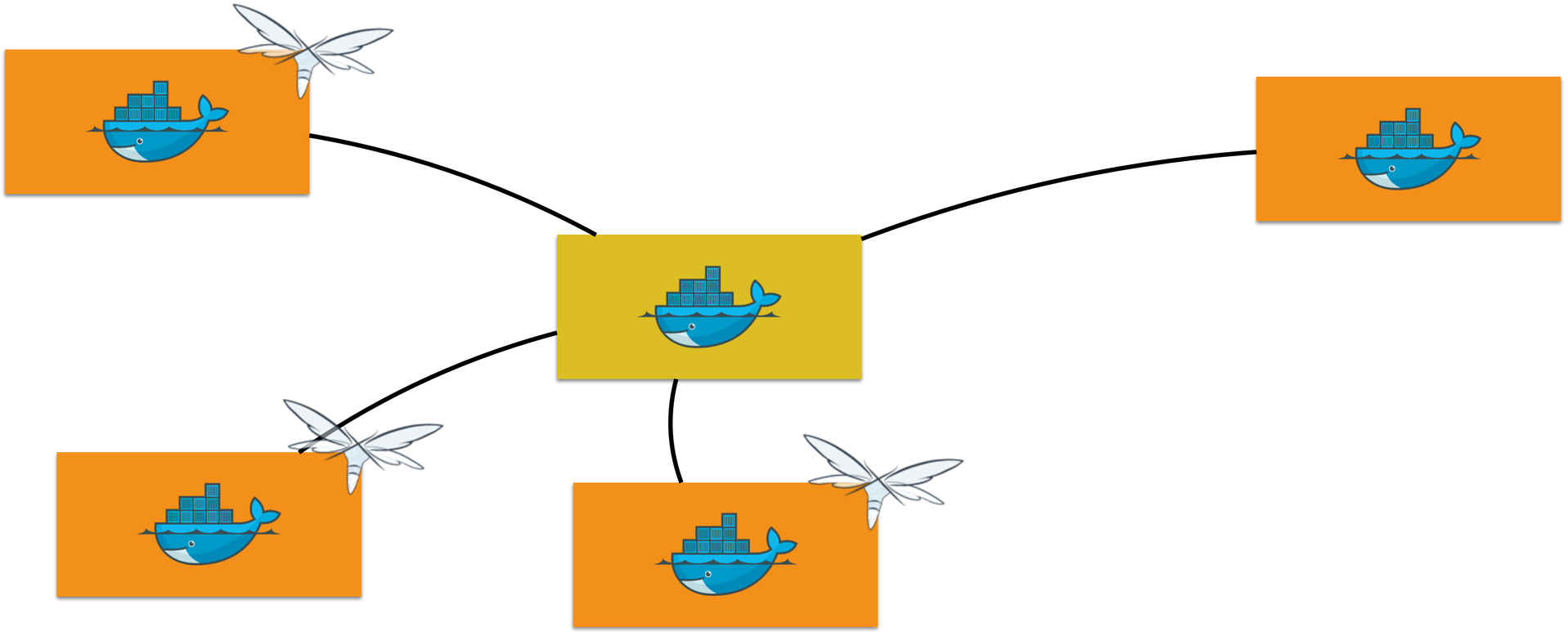
Swarm Mode: Node Failure



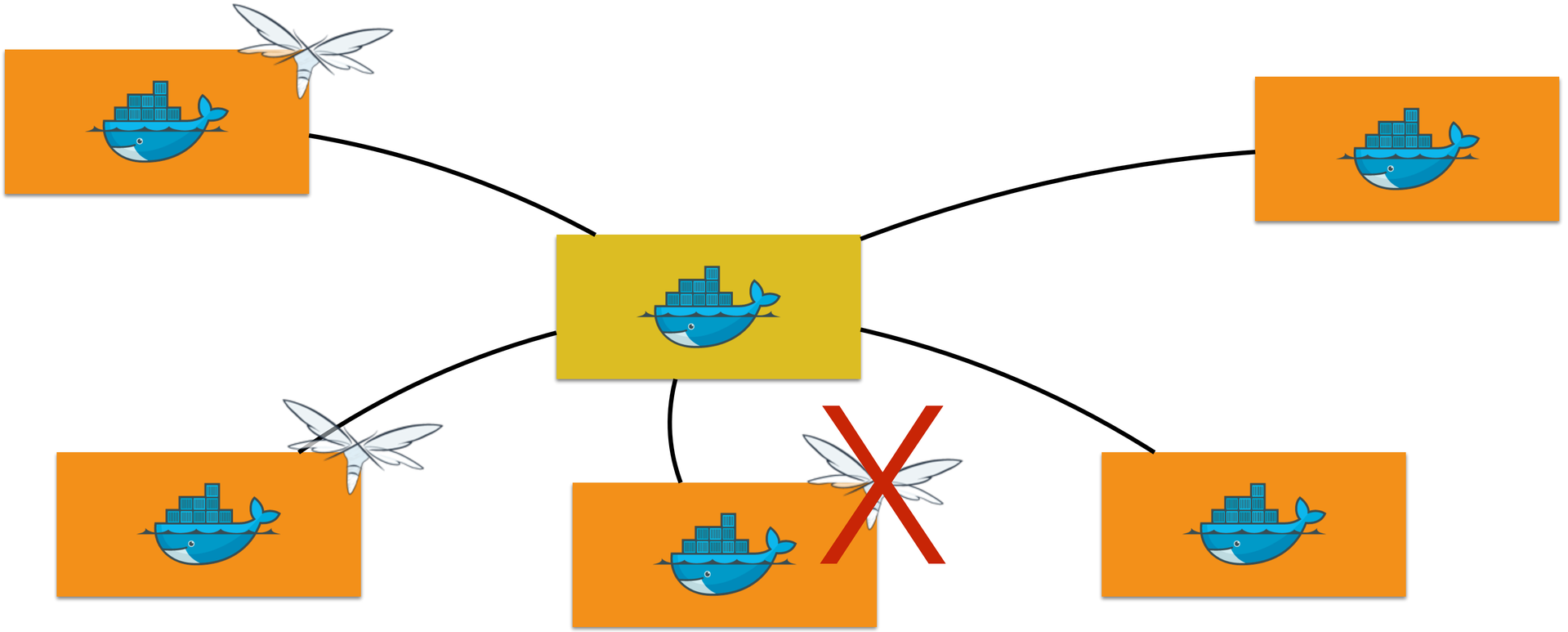
Swarm Mode: Desired != Actual



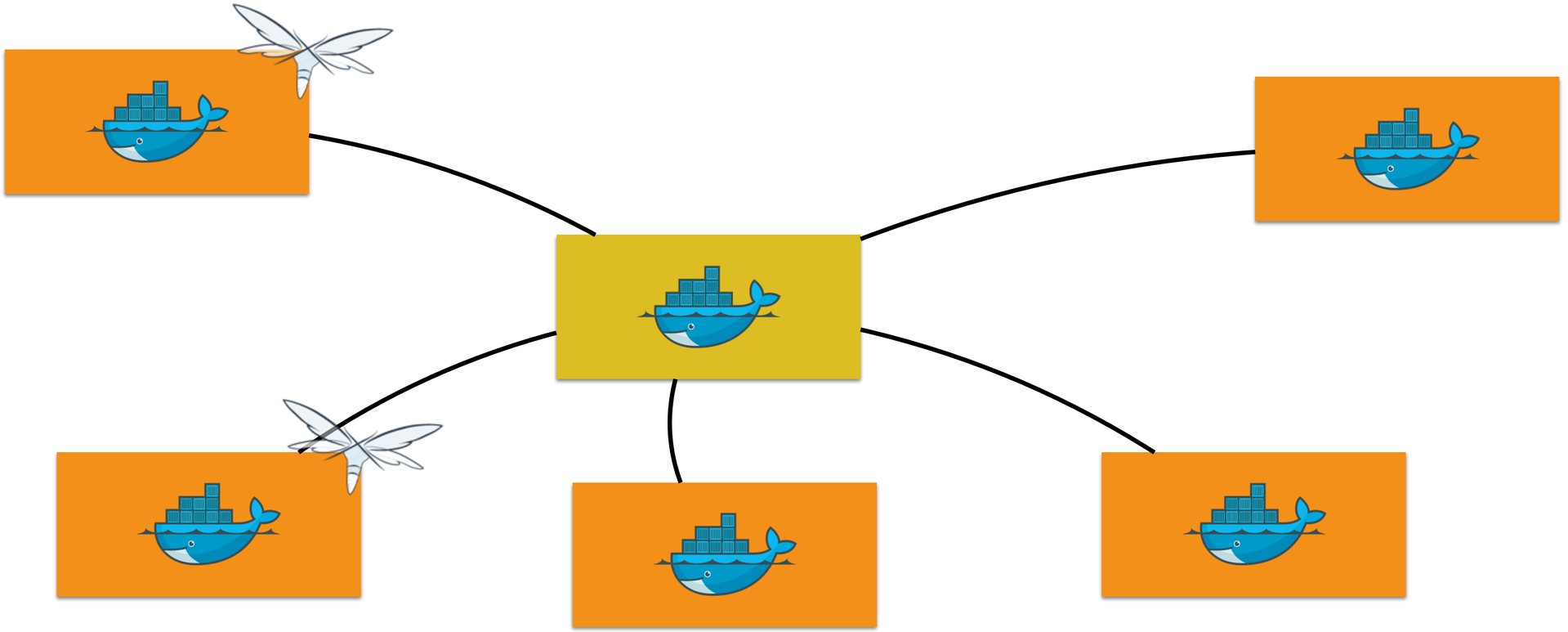
Swarm Mode: Reconcile



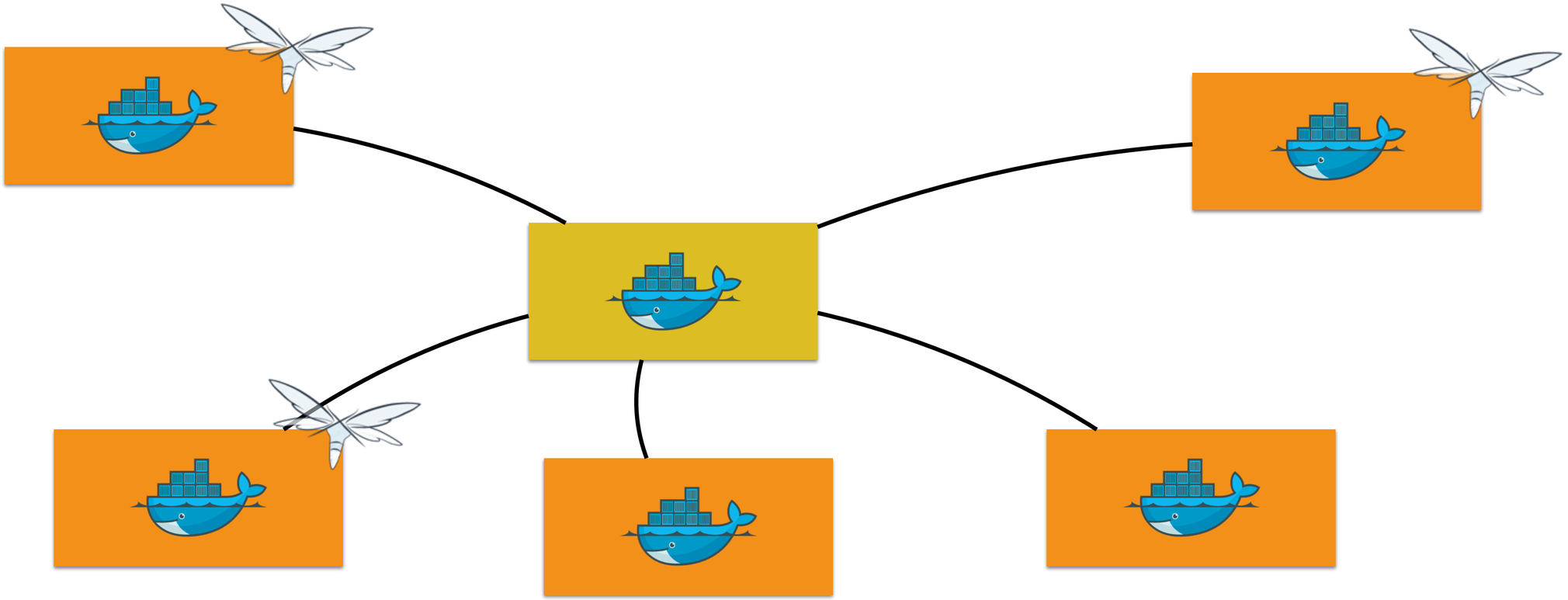
Swarm Mode: Container Failure



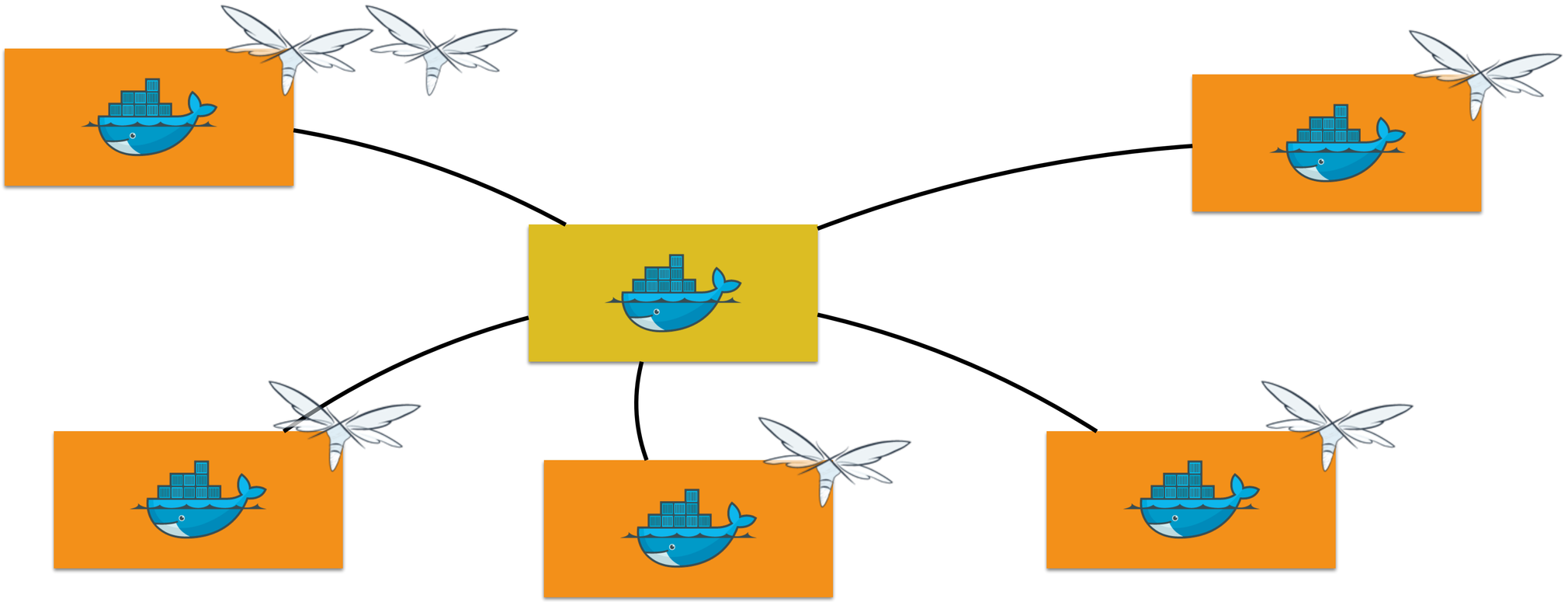
Swarm Mode: Desired != Actual



Swarm Mode: Reconcile

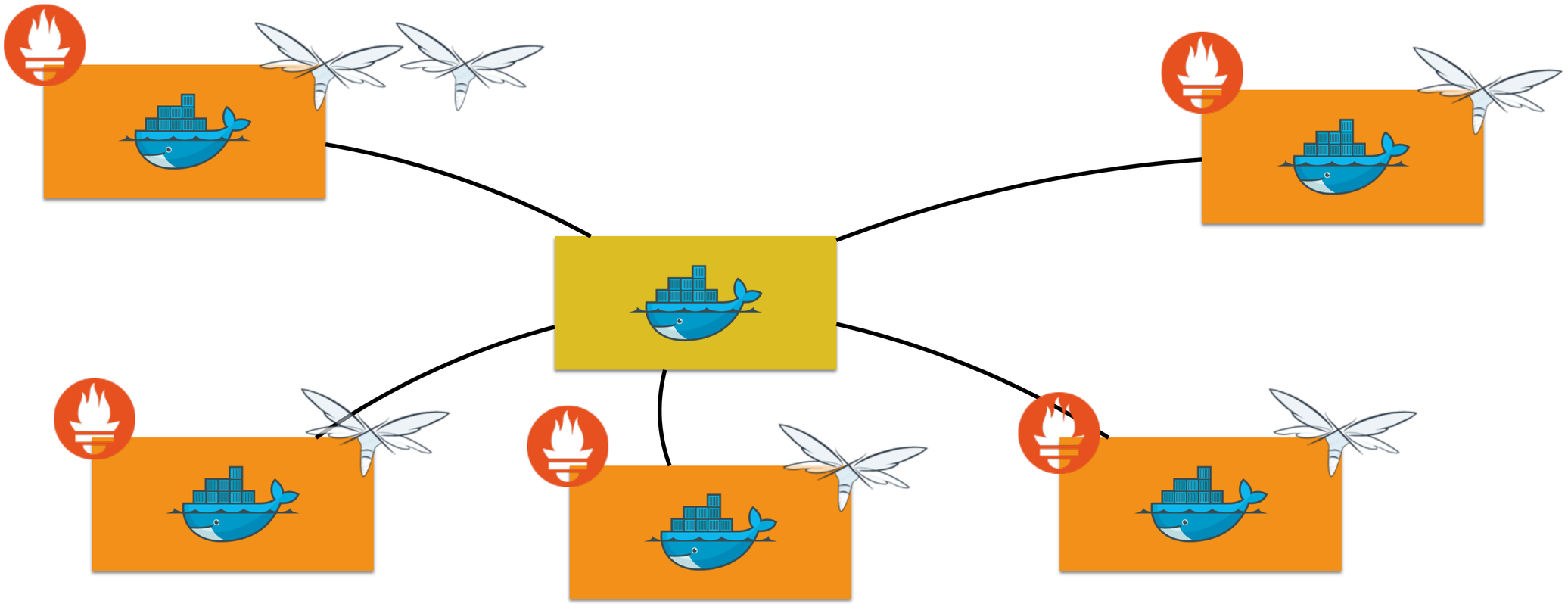


Swarm Mode: Scale



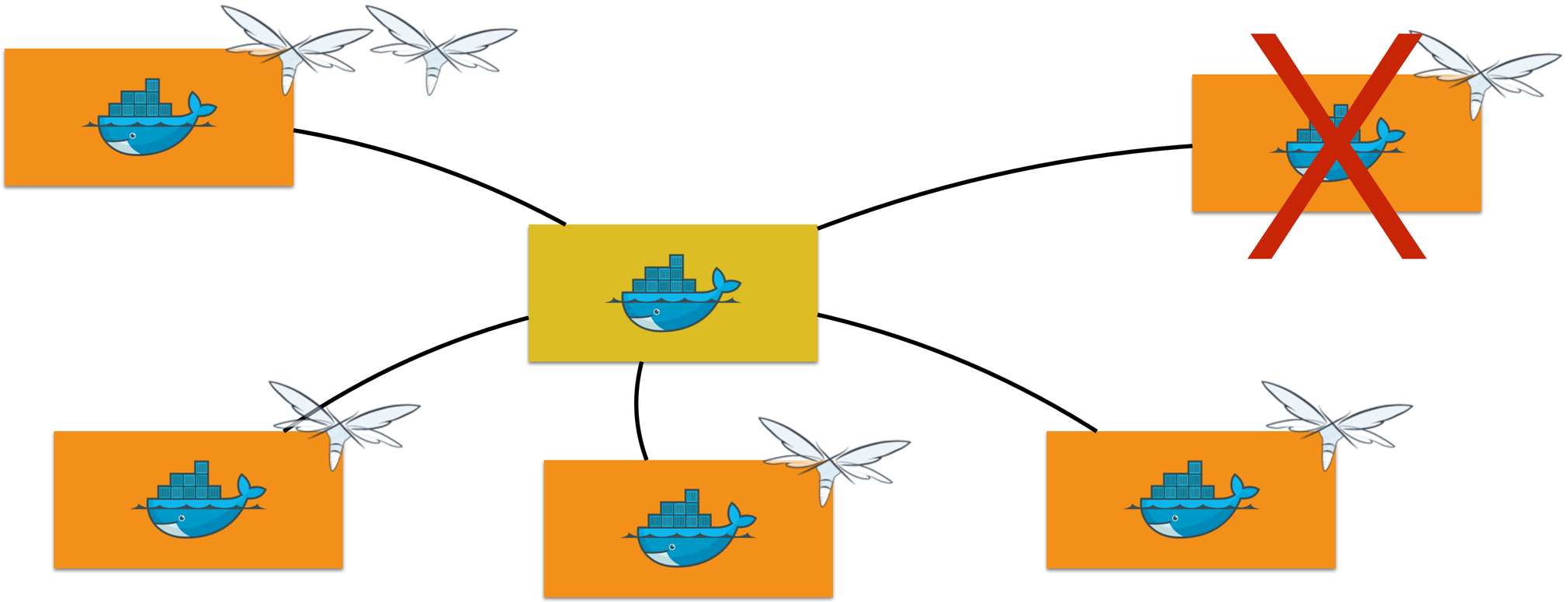
```
docker service scale web=6
```

Swarm Mode: Global Service



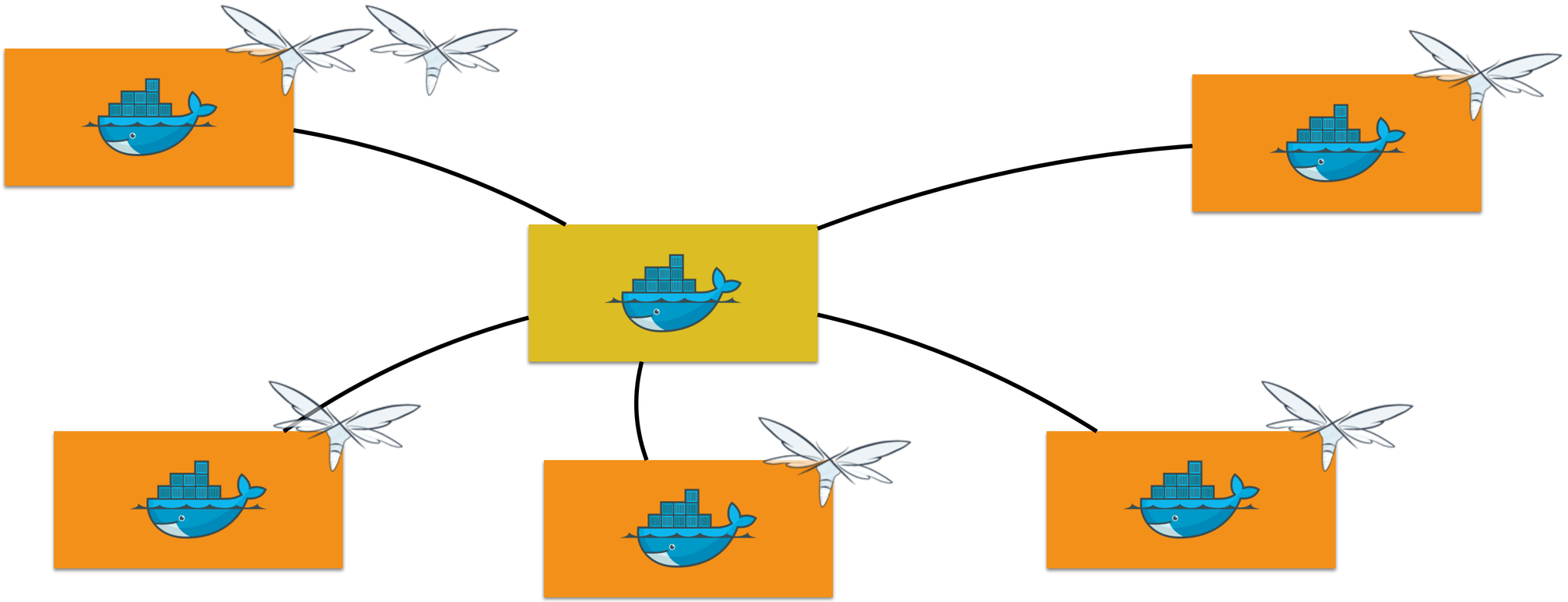
```
docker service create --mode=global --name=prom prom/prometheus
```

Swarm Mode: Pause Node



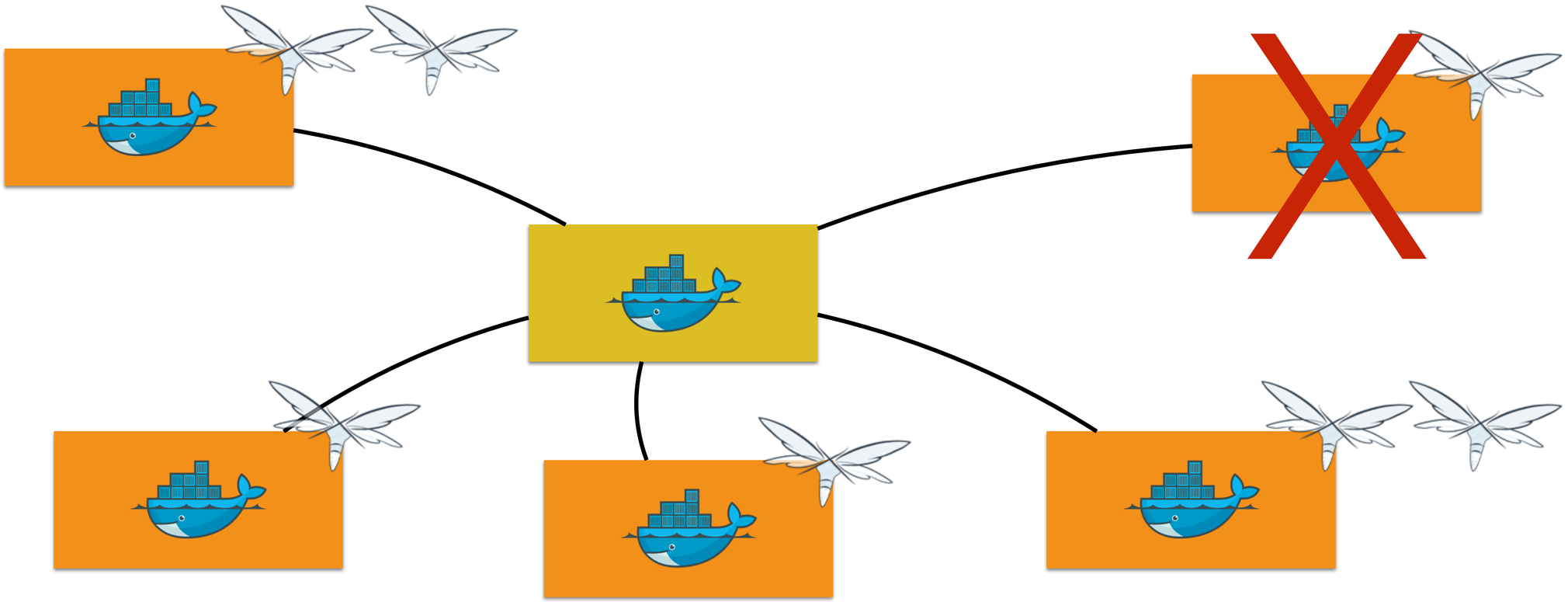
```
docker node update --availability pause <nodename>
```

Swarm Mode: Active Node



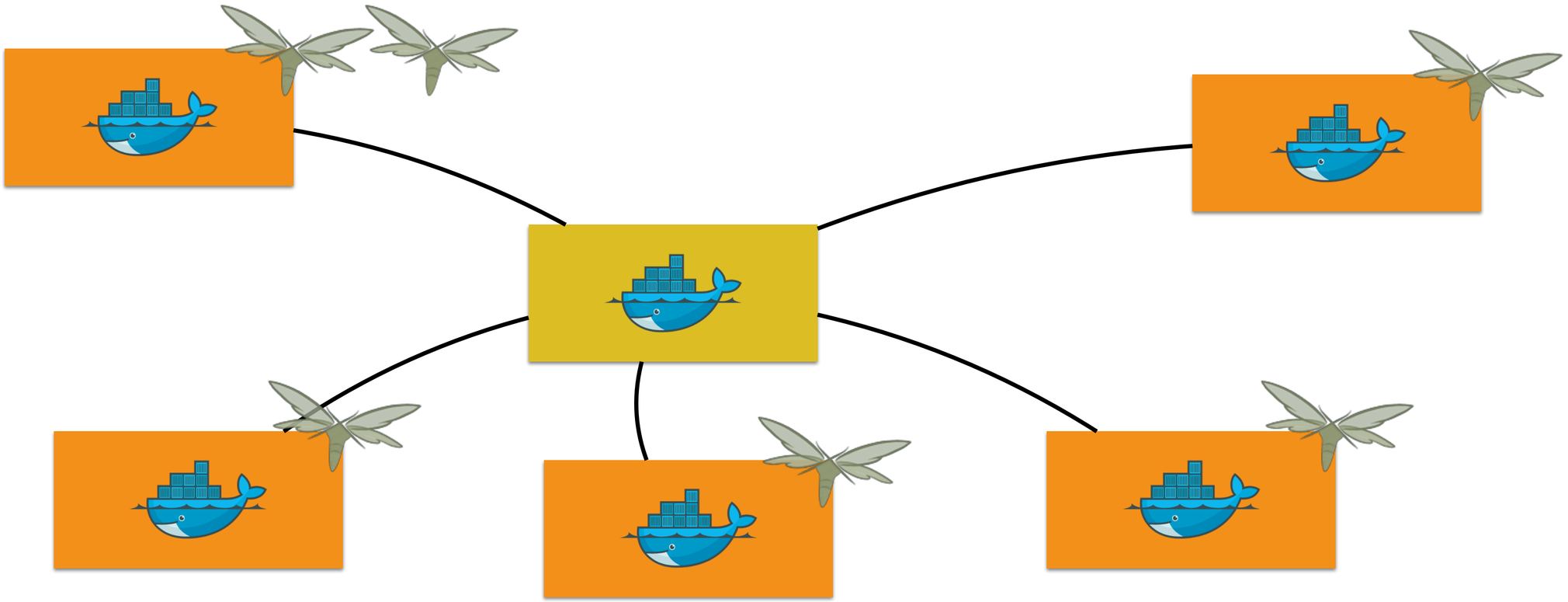
```
docker node update --availability active <nodename>
```

Swarm Mode: Drain Node



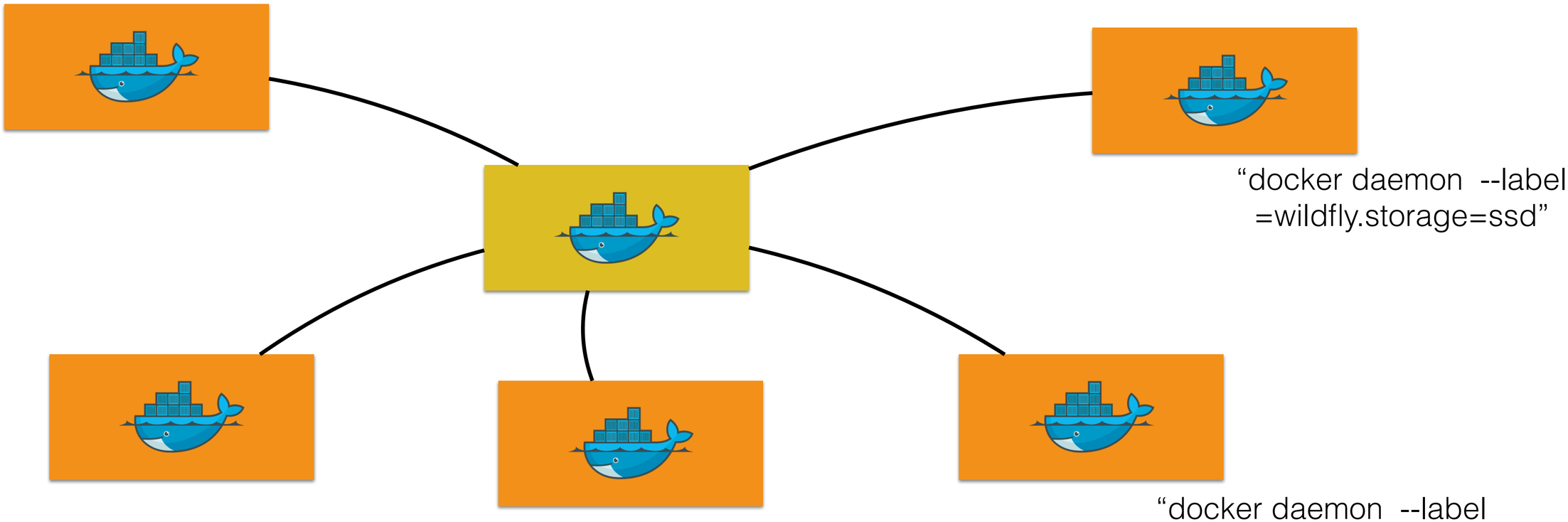
```
docker node update --availability drain <nodename>
```

Swarm Mode: Rolling Updates



```
docker service update web --image wildfly:2 --update-parallelism
2 --update-delay 10s
```

Swarm Mode: Label

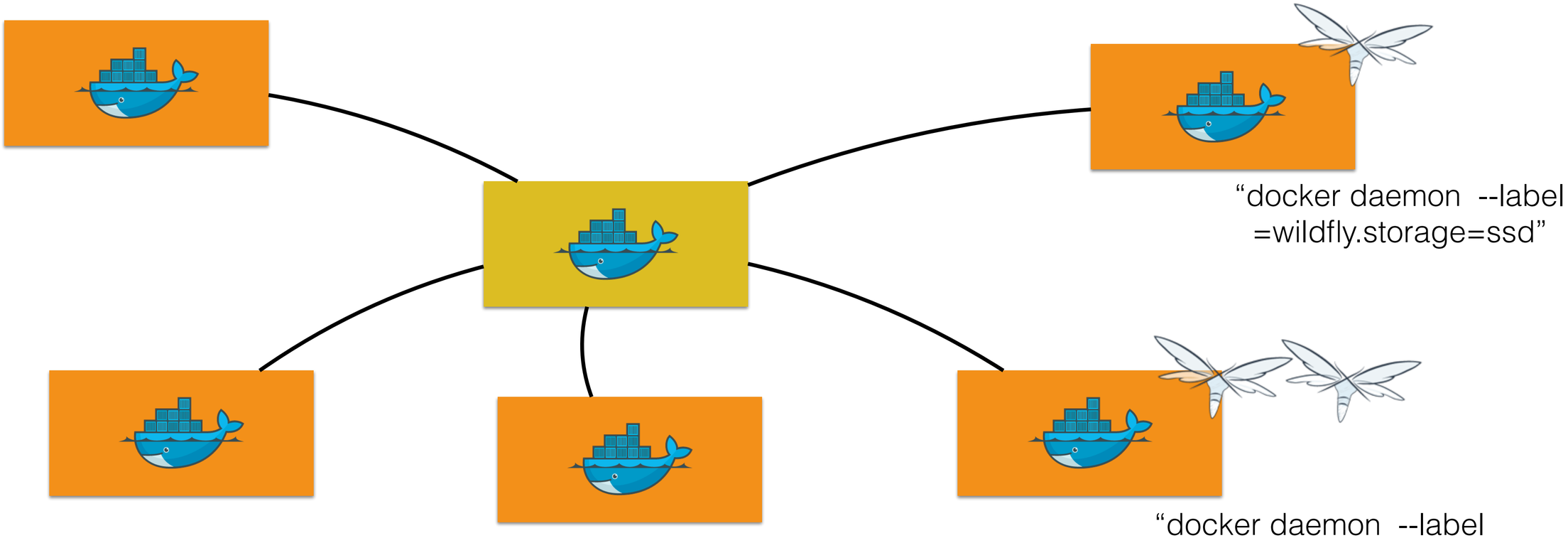


“docker daemon --label =wildfly.storage=ssd”

“docker daemon --label =wildfly.storage=ssd”

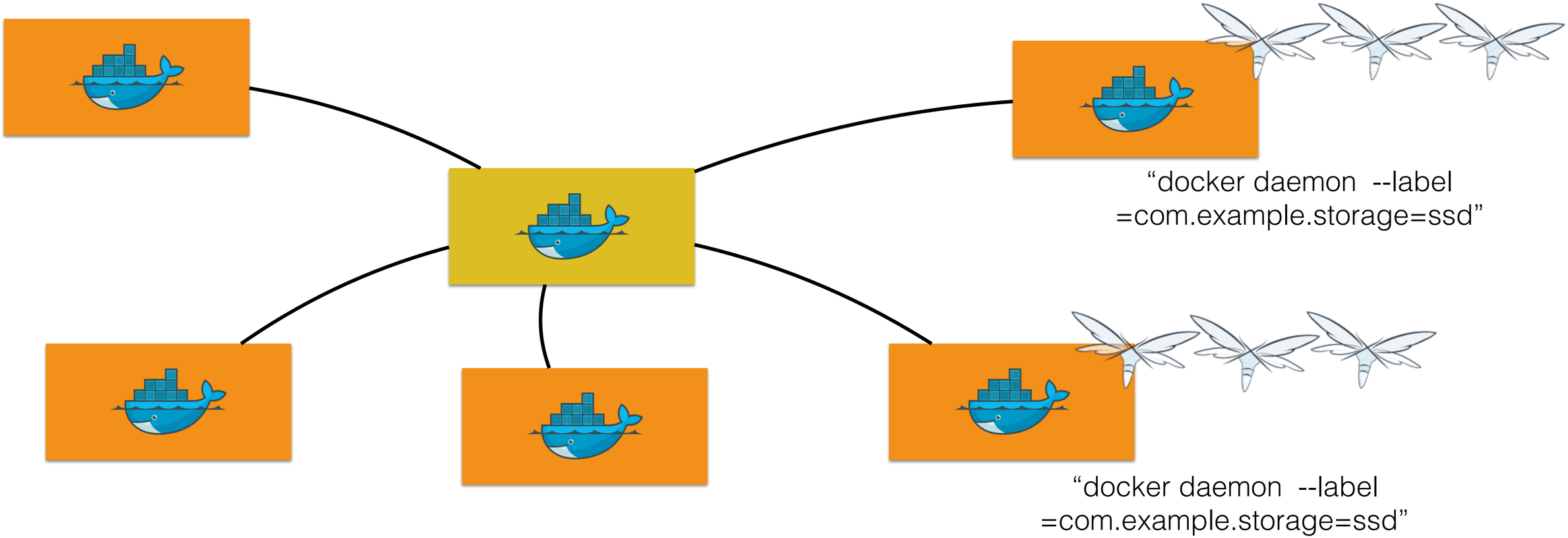
```
DOCKER_OPTS="--label=wildfly.storage=ssd"
```

Swarm Mode: Constraints



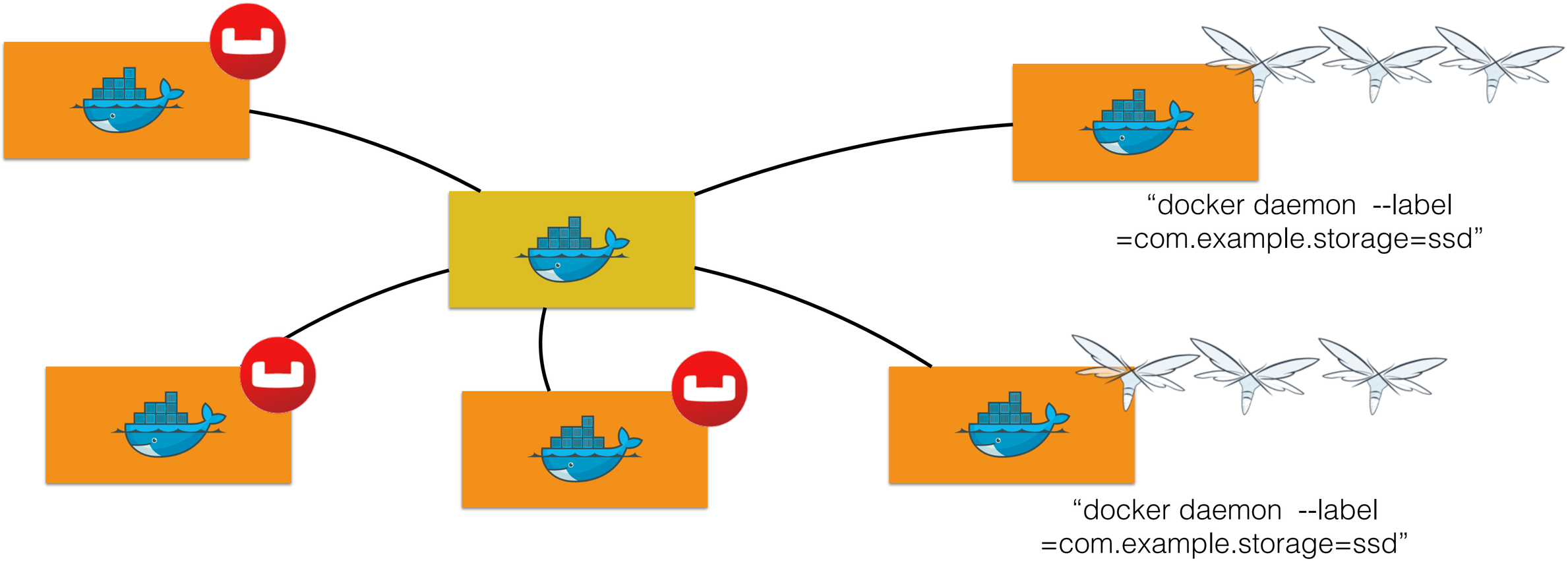

```
docker service create --replicas=3 --name=web --constraint
engine.labels.wildfly.storage==ssd jboss/wildfly
```

Swarm Mode: Constraints




```
docker service scale web=6
```

Swarm Mode: Constraints




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

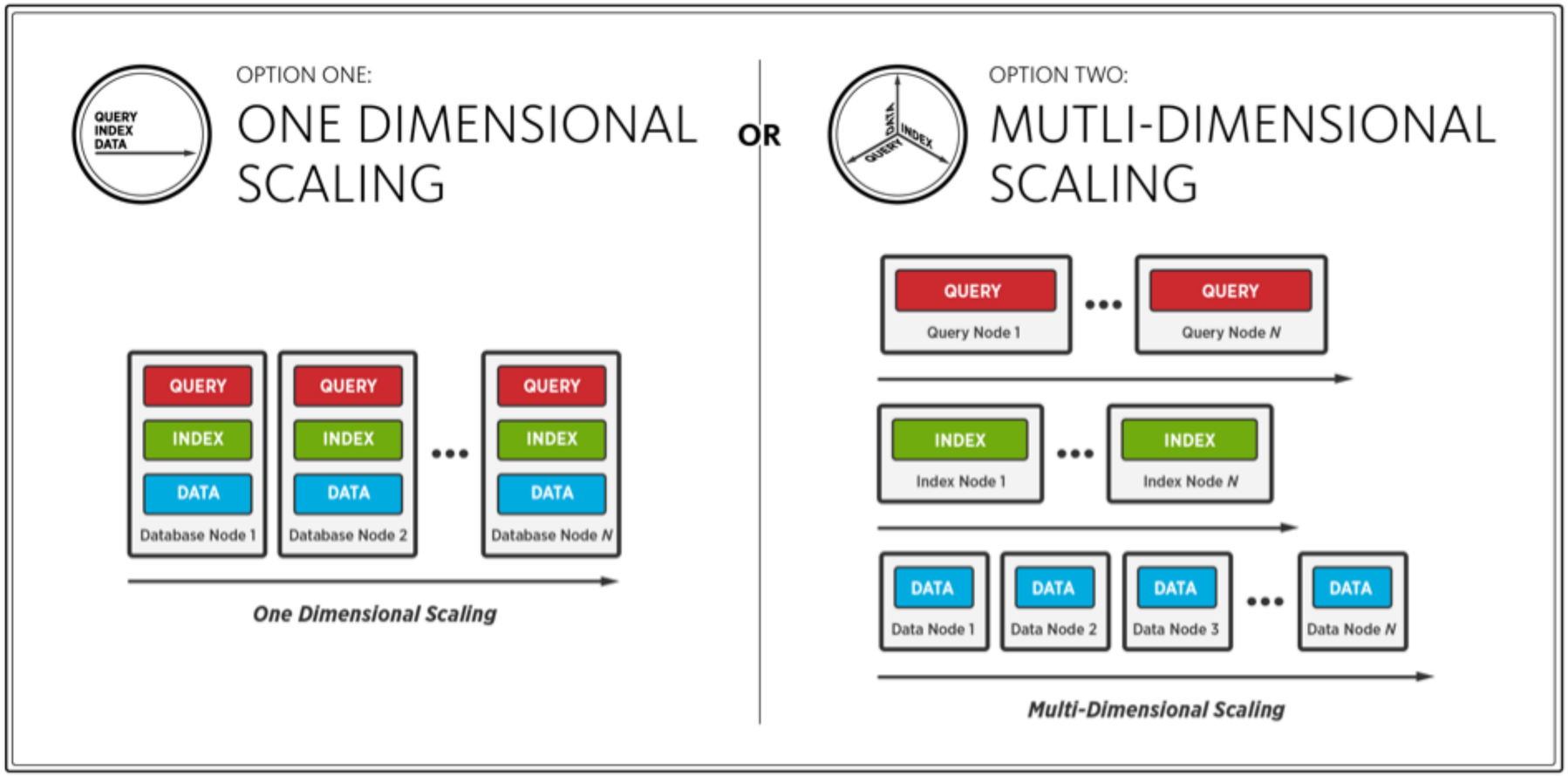
Scheduling Backends using Filters

- **Label**: Metadata attached to Docker Daemon
- **Filters**: Used by Docker Swam scheduler to create and run container

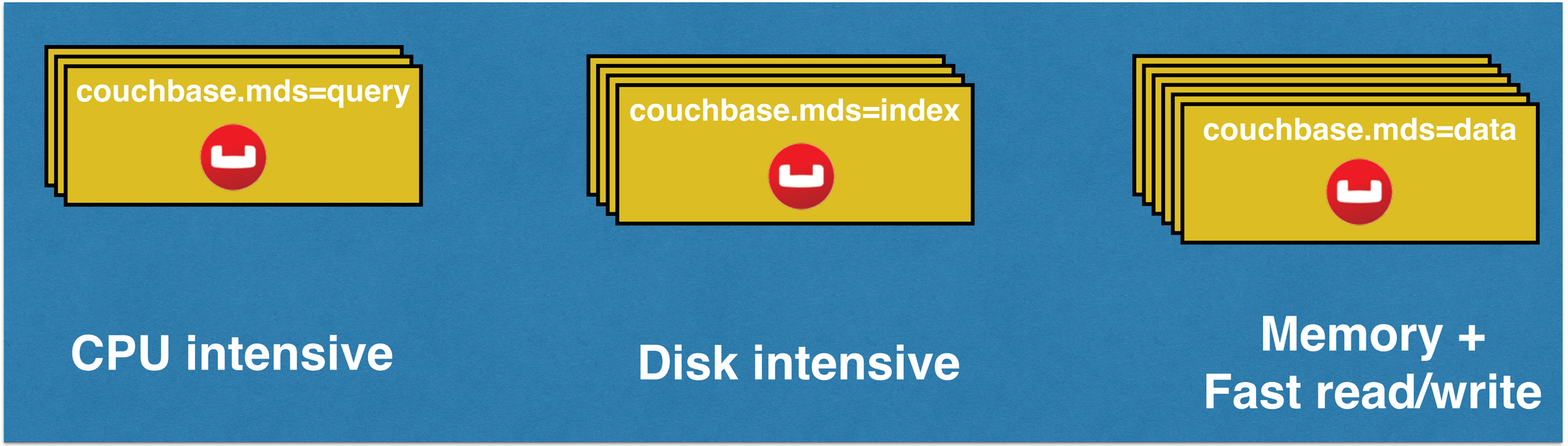
Node		
Constraint	Default or custom tags	`node`, `operatinsystem`, `kernelversion`, …
Health	Schedule containers on healthy nodes only	
Container Slots	Maximum number of containers on a node	`--labels containerslots=3`
Container		
Affinity	“Attraction” between containers	`-e affinity:container=<name>/<id>`, `image`, …
Dependency	Dependent containers on same node	`--volumes-from=<id>`, `--net=container:<id>`, …
Port	Port availability on the host	`-p <host>:<container>`

Couchbase Multi Dimensional Scaling

Only Couchbase gives customers two options for scaling: Standard One-Dimensional Scaling and New Multi-Dimensional Scaling.

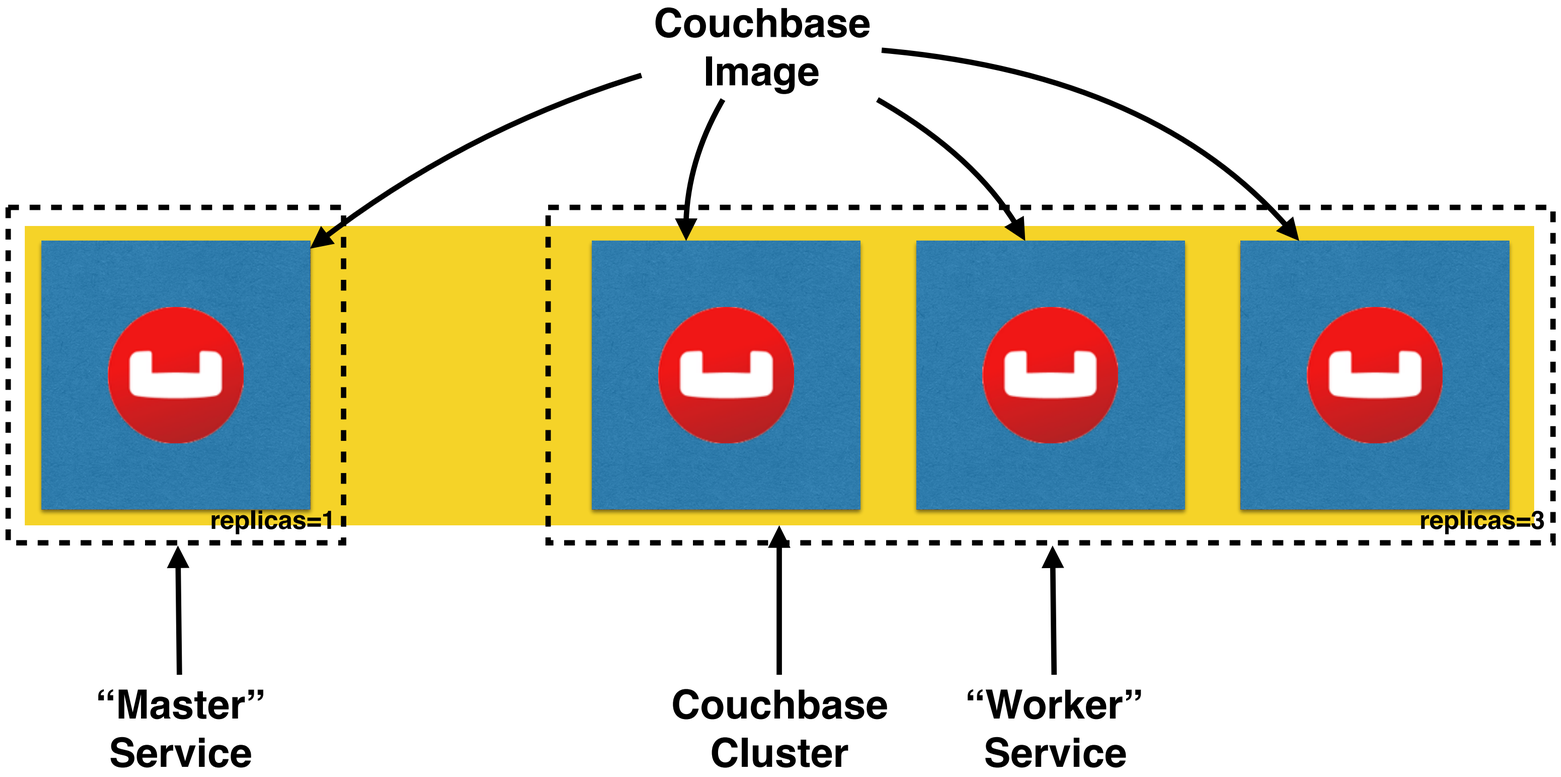


Optimal Utilization of Resources



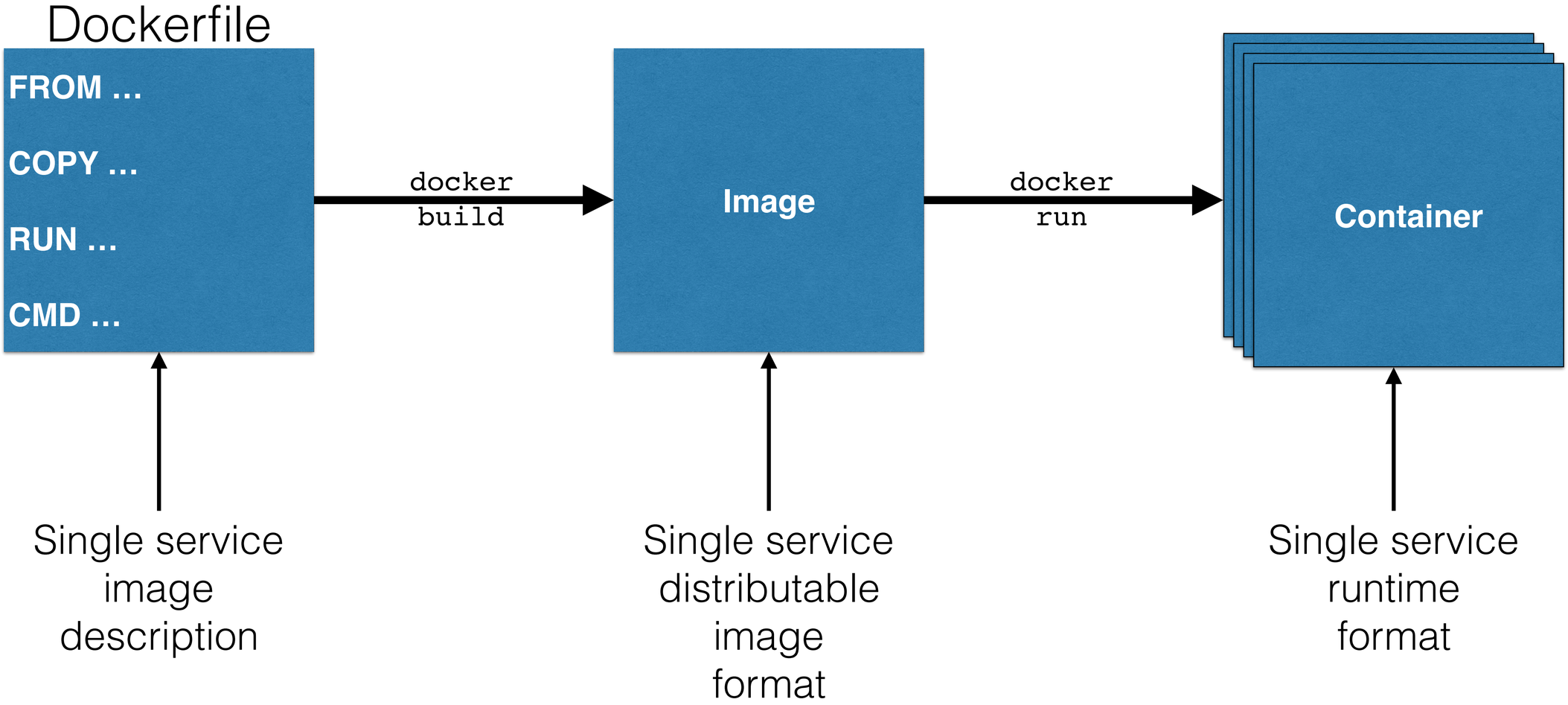
- **Attach labels**: `DOCKER_OPTS="--label.couchbase.mds=data"`
- **Run Containers**: `docker service create --constraint engine.labels.couchbase.mds==index couchbase`

Couchbase Cluster using Docker Services



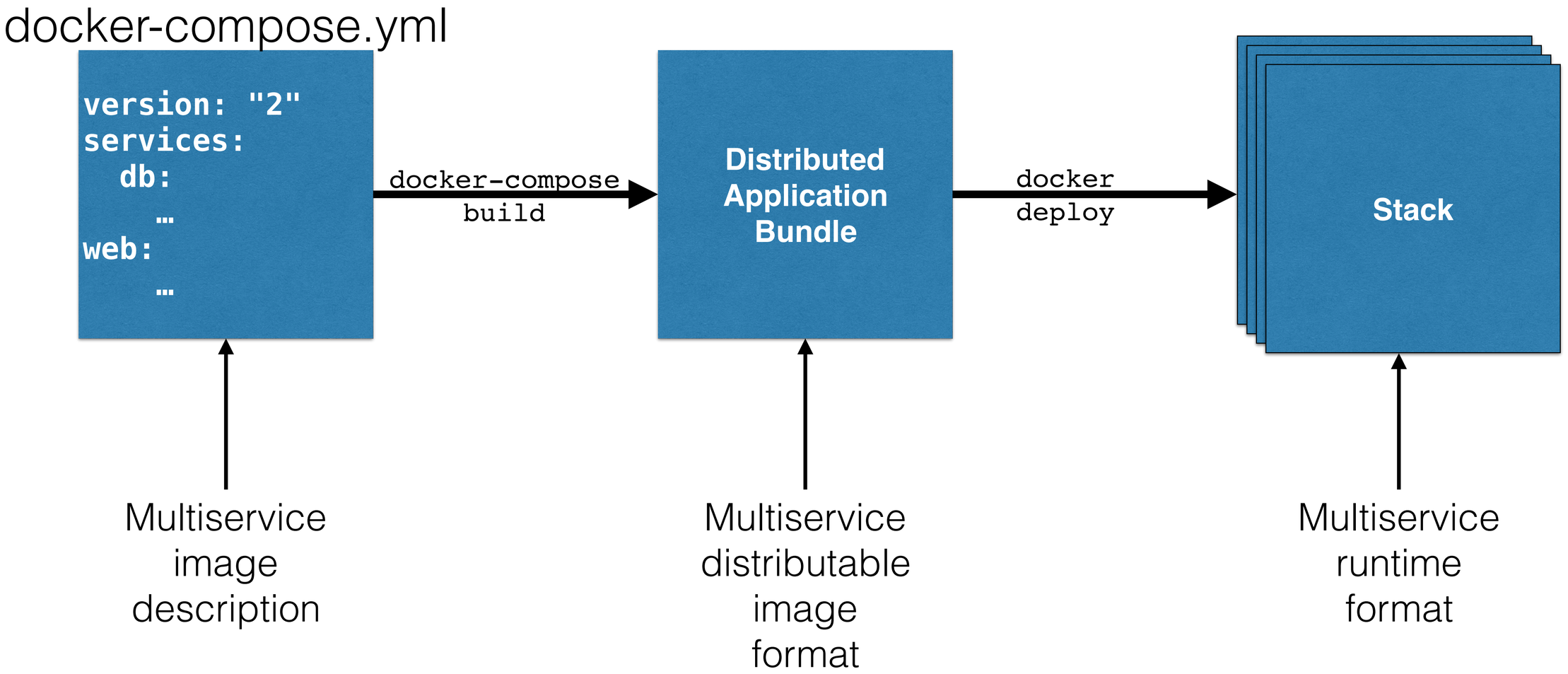
http://blog.couchbase.com/2016/september/docker-service-swarm-mode-couchbase-cluster

Docker Lifecycle



Experimental feature

Distributed Application Bundle



NOTE: Docker Engine and Registry do not support distribution of DABs

Persistent Storage

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

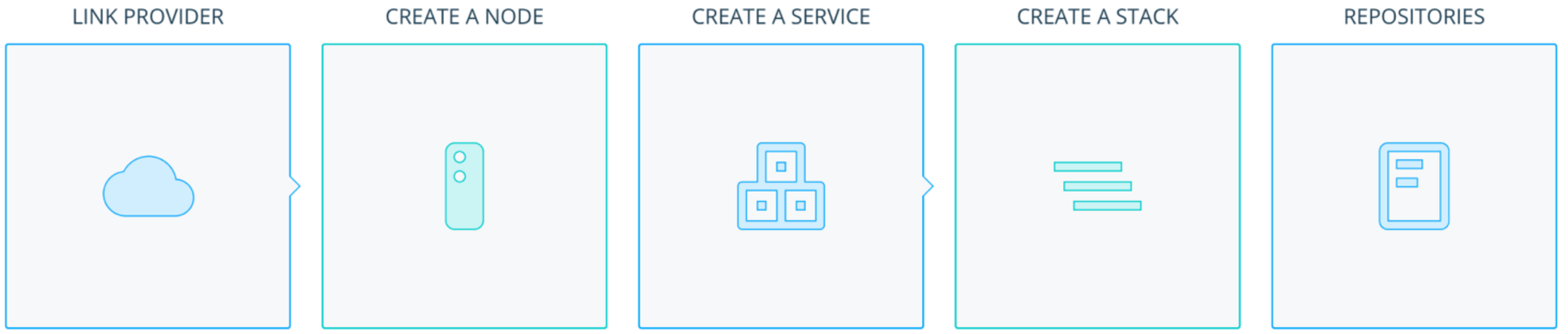
```
docker volume create –name=data data
```

Run a container with the volume

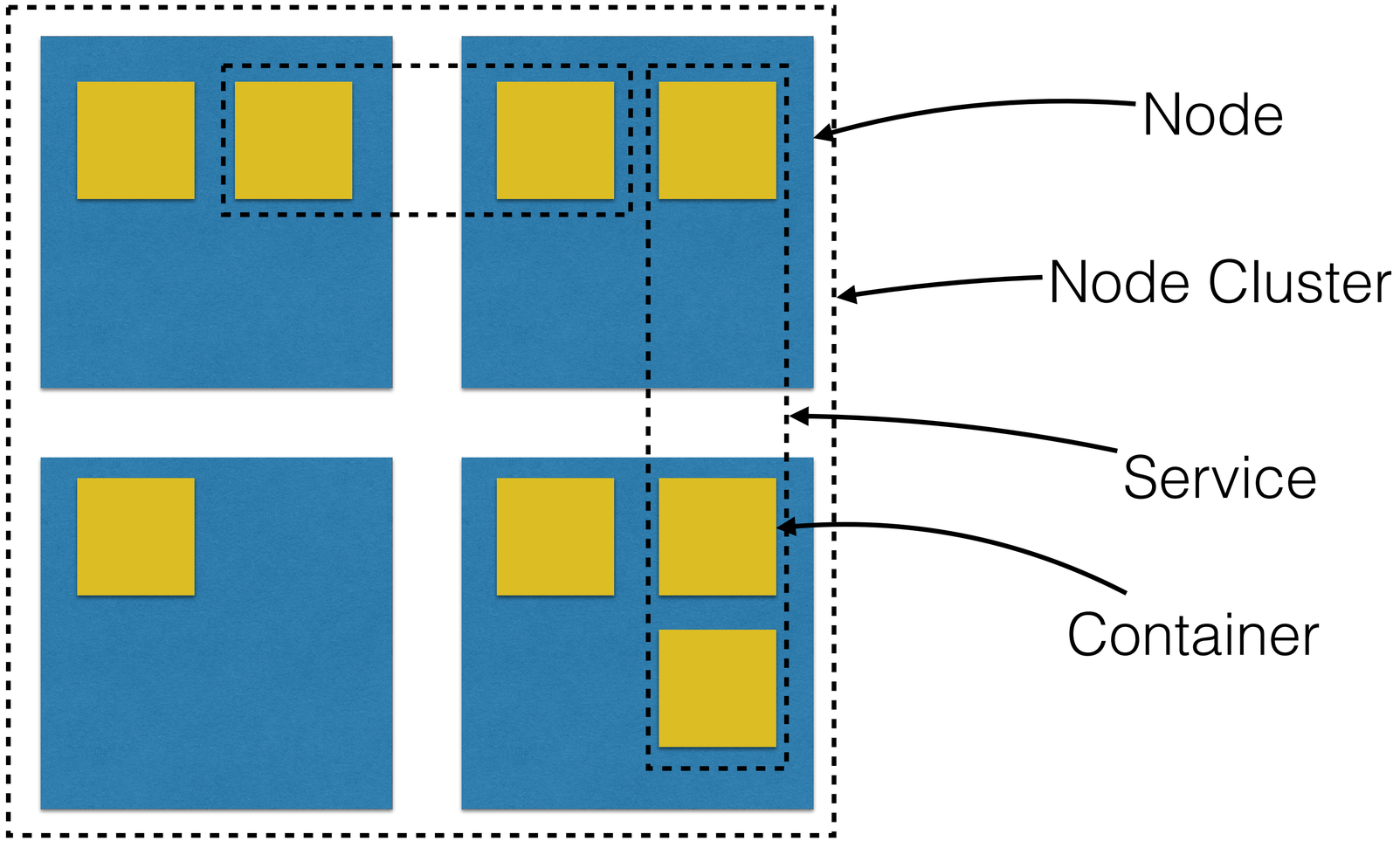
```
docker run -it -v data:/opt/couchbase/var couchbase
```

Docker Cloud

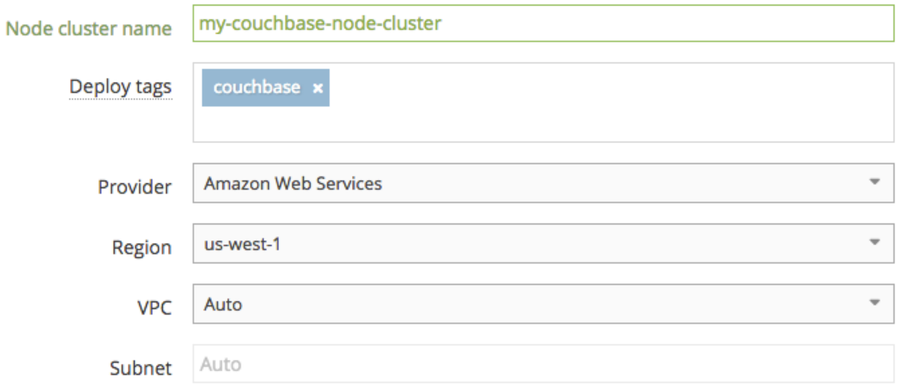
- SaaS
- Build, deploy and manage your images across different clouds
  - Amazon, Digital Ocean, Azure, BYON

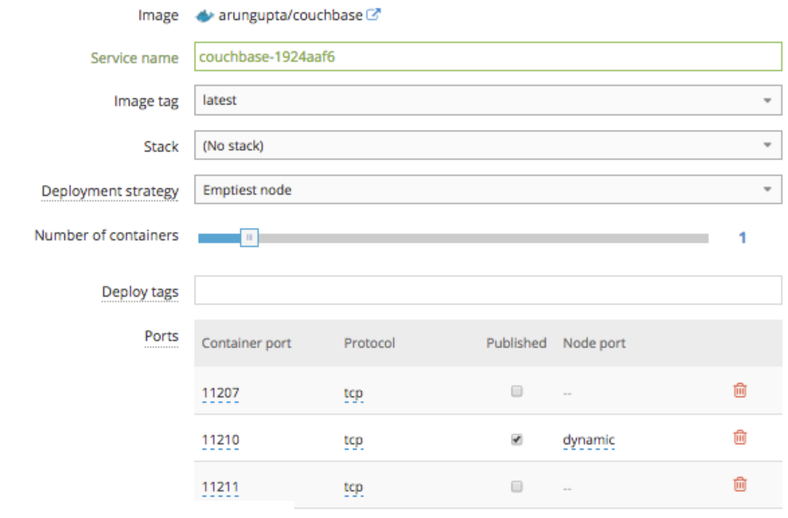


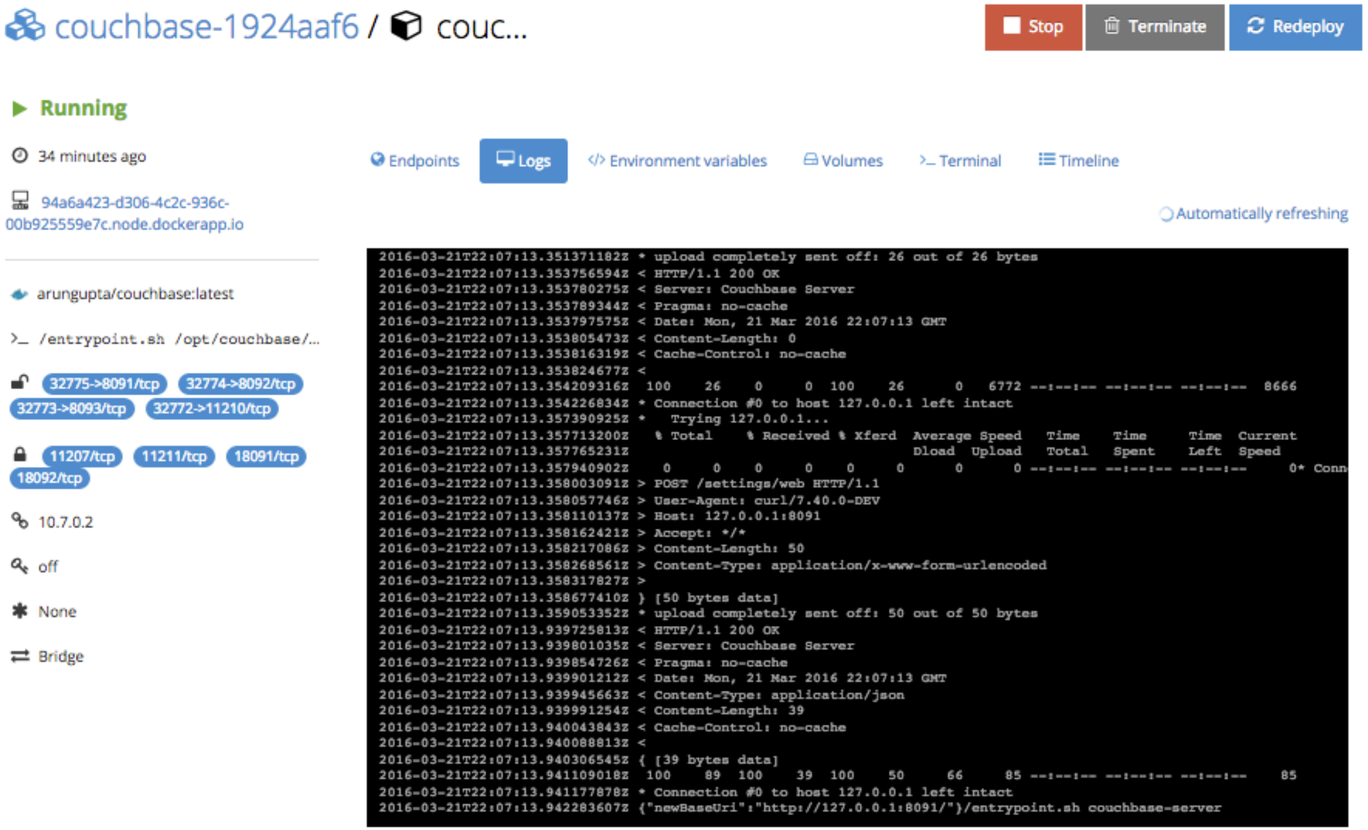
Docker Cloud



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

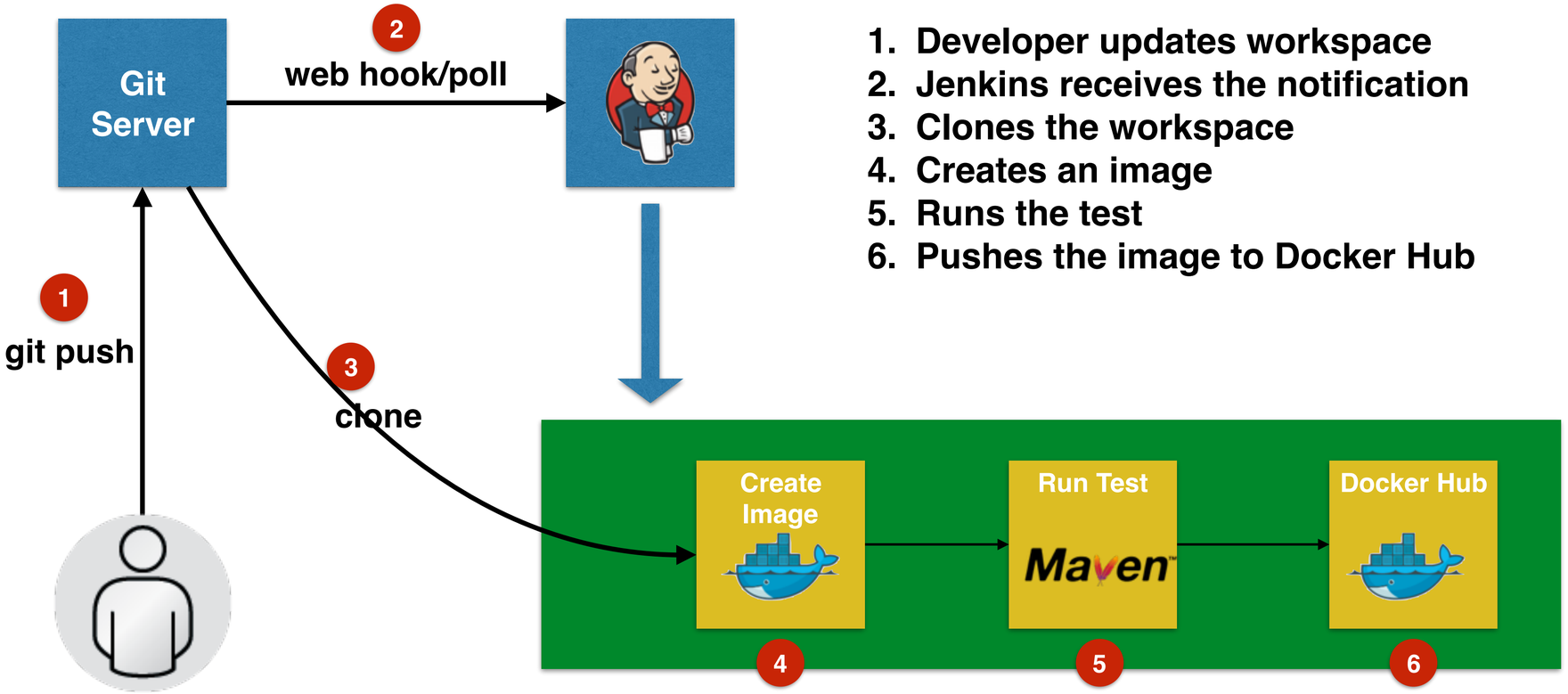
Docker Registry

- 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 Primary Usage

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

CI/CD with Docker + Jenkins

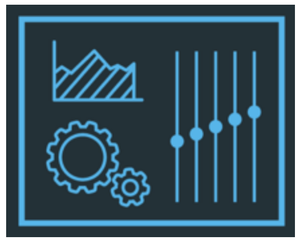


1. Developer updates workspace
2. Jenkins receives the notification
3. Clones the workspace
4. Creates an image
5. Runs the test
6. Pushes the image to Docker Hub

http://blog.couchbase.com/2016/september/deployment-pipeline-docker-jenkins-java-couchbase

Monitoring Docker Containers

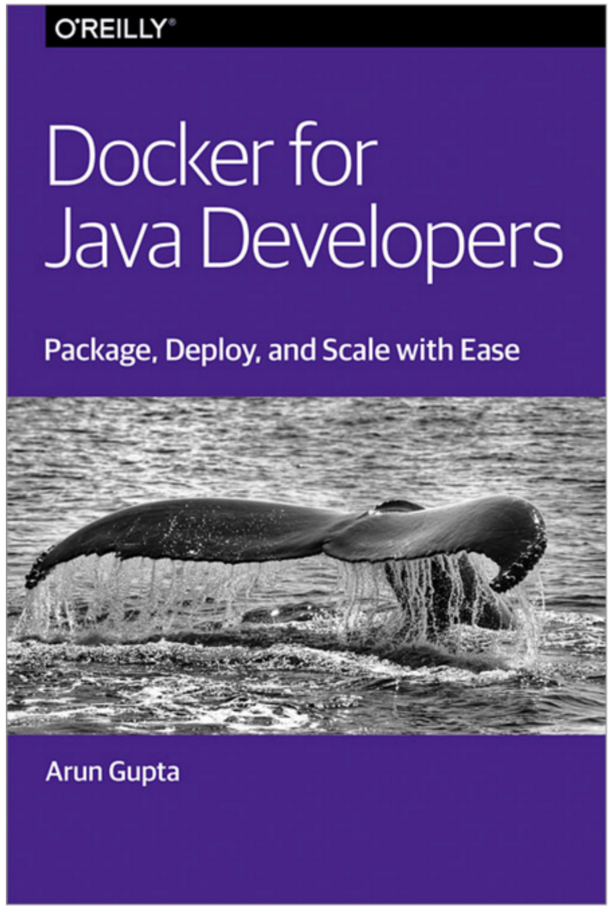
- `docker stats` command
  - LogEntries
- Docker Remote API: `/container/{container-name|cid}/stats`
- Docker Universal Control Plane
- cAdvisor
  - Prometheus
  - InfluxDB



Docker Support in Java IDEs

https://github.com/arun-gupta/docker-java-ides

bit.ly/dockerjava



bit.ly/kubejava

References

- Slides: github.com/docker/labs/tree/master/slides
- Workshop: github.com/docker/labs/tree/master/java
- Docs: docs.docker.com