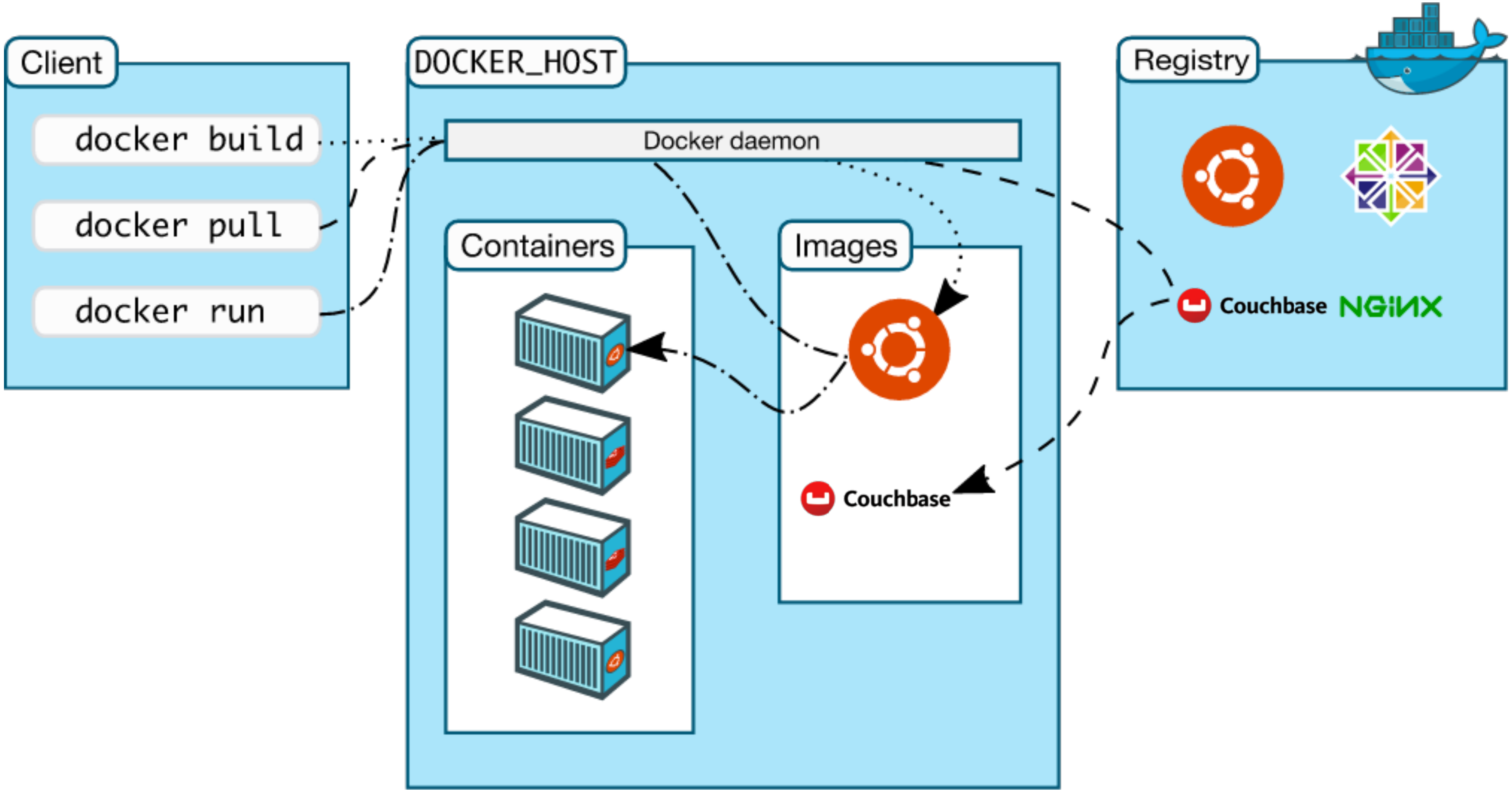
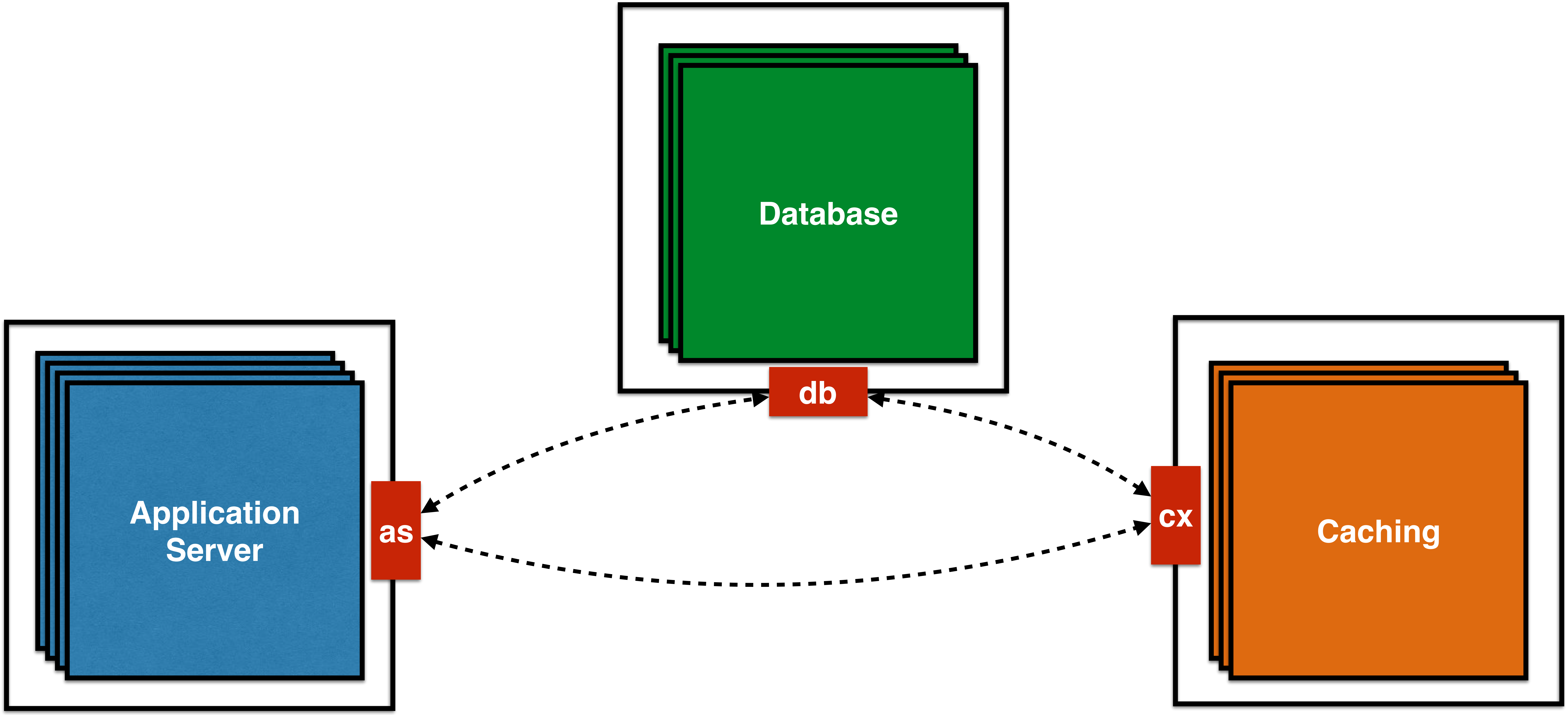
Container Orchestration on Amazon Web Services

Arun Gupta, @arungupta

Docker Workflow

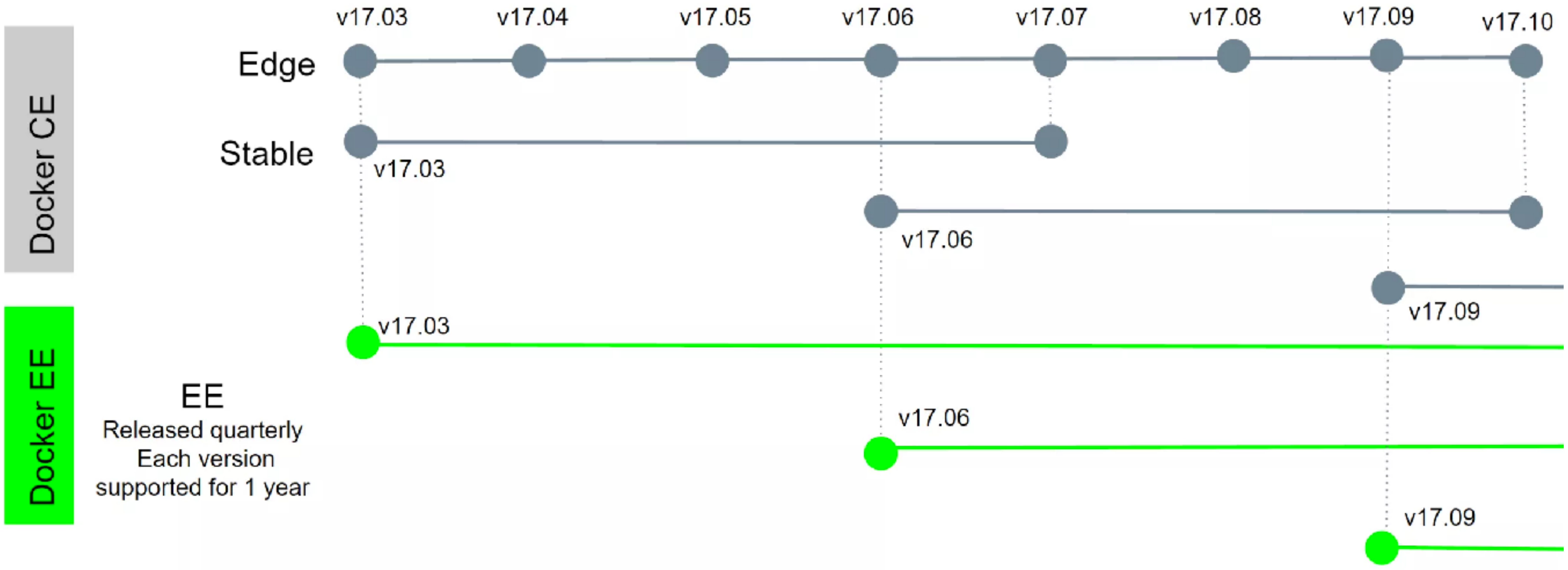


Container Orchestration



Development using Docker

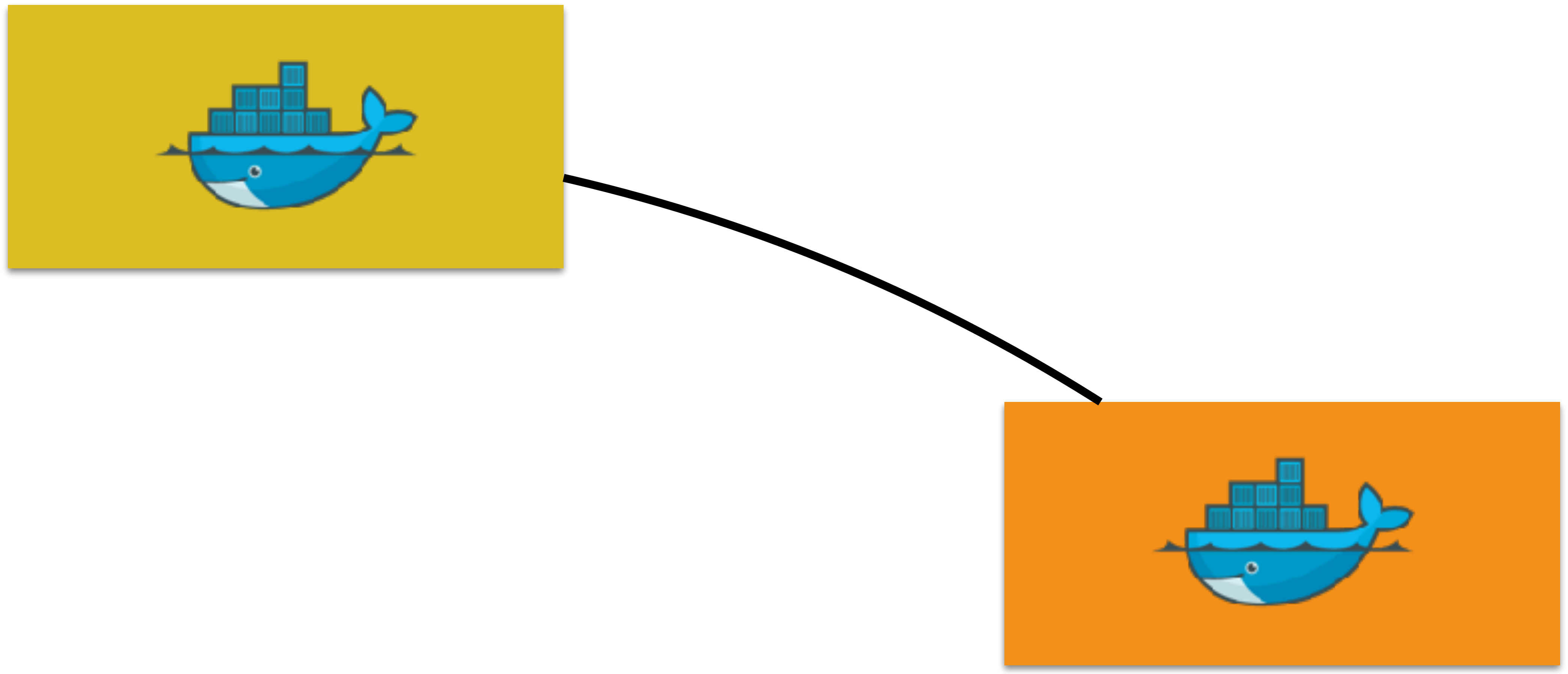
- Docker Community Edition
  - Docker for Mac/Windows/Linux
  - Monthly edge and quarterly stable releases
  - Native desktop or cloud provider experience



Swarm-mode: Initialize

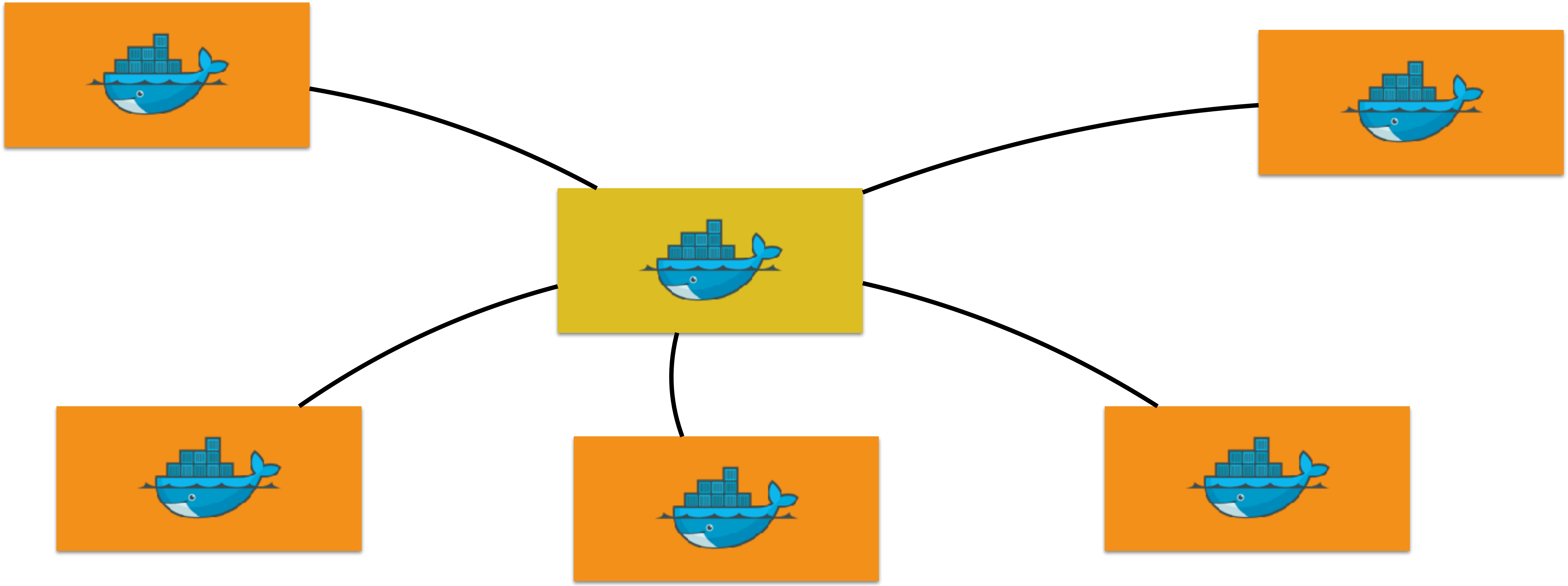
```
docker swarm init --listen-addr <ip>:2377 --secret <SECRET>
```

Swarm-mode: Add Worker



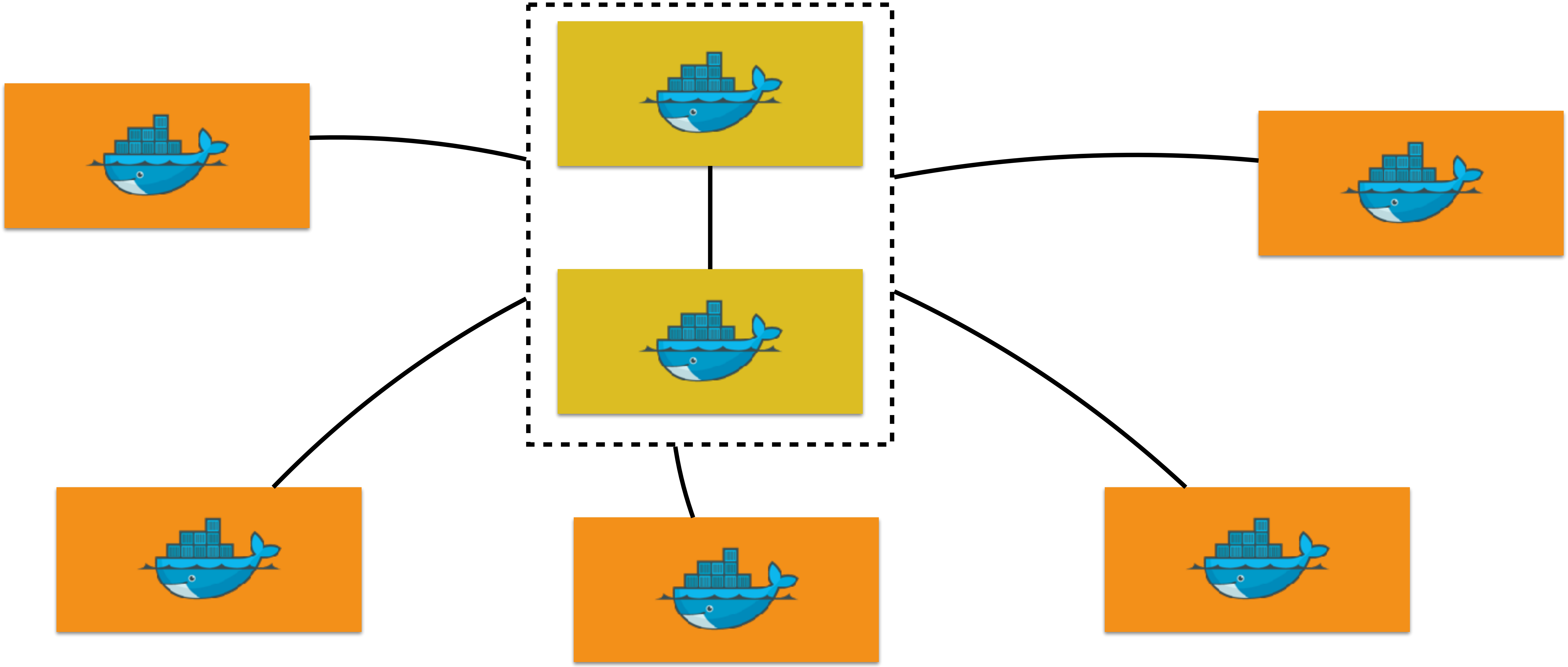
```
docker swarm join --secret <SECRET> <manager>:2377
```

Swarm-mode: Add More Workers



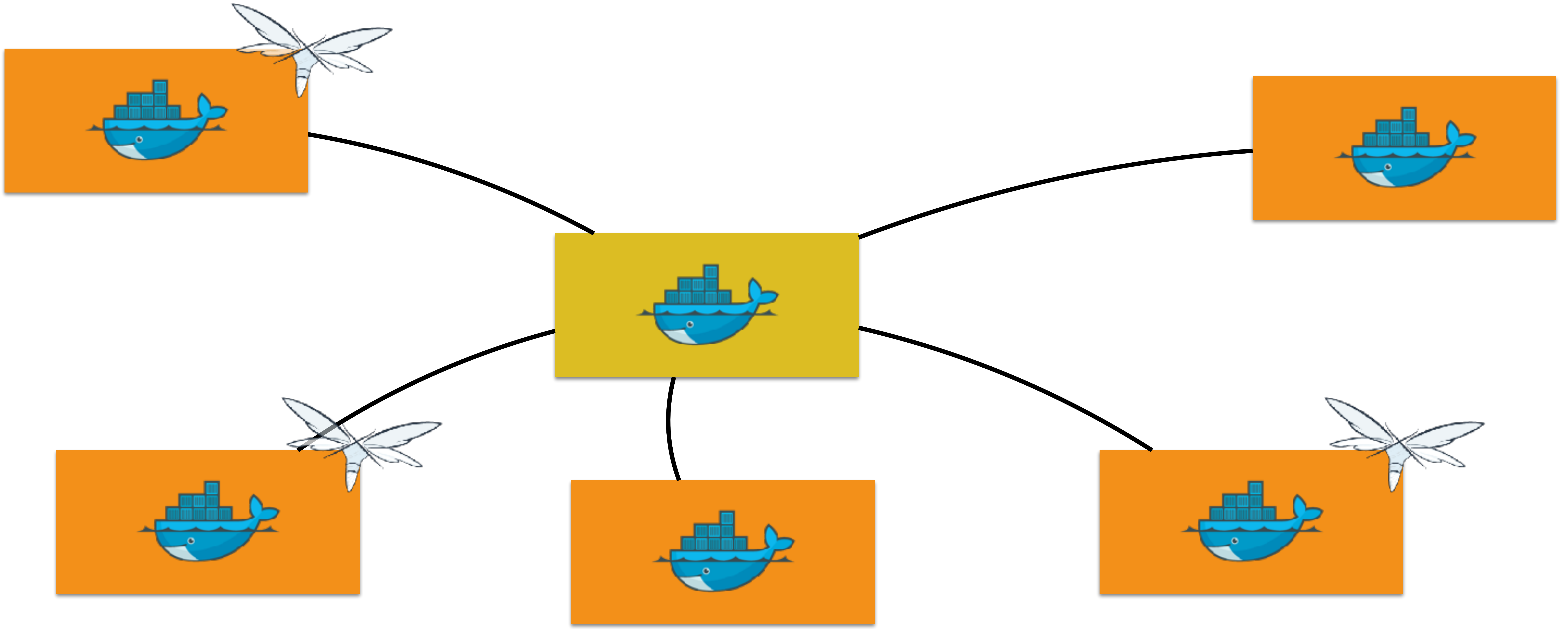
```
docker swarm join --secret <SECRET> <manager>:2377
```

Swarm-mode: Primary/Secondary Master



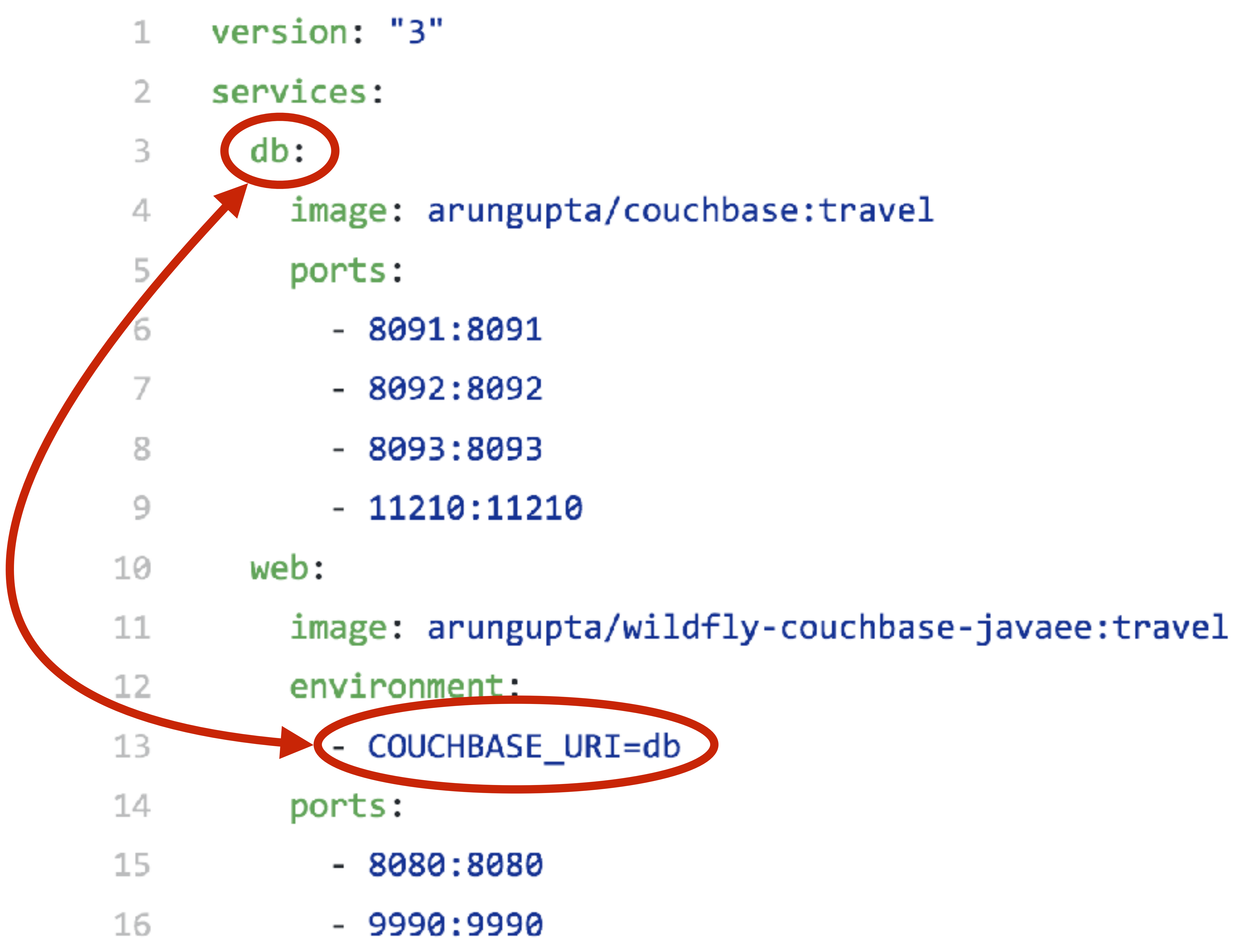
```
docker swarm join --manager --secret <SECRET> --listen-addr
<master2>:2377 <master1>:2377
```

Swarm-mode: Replicated Service



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

Service Discovery with Docker

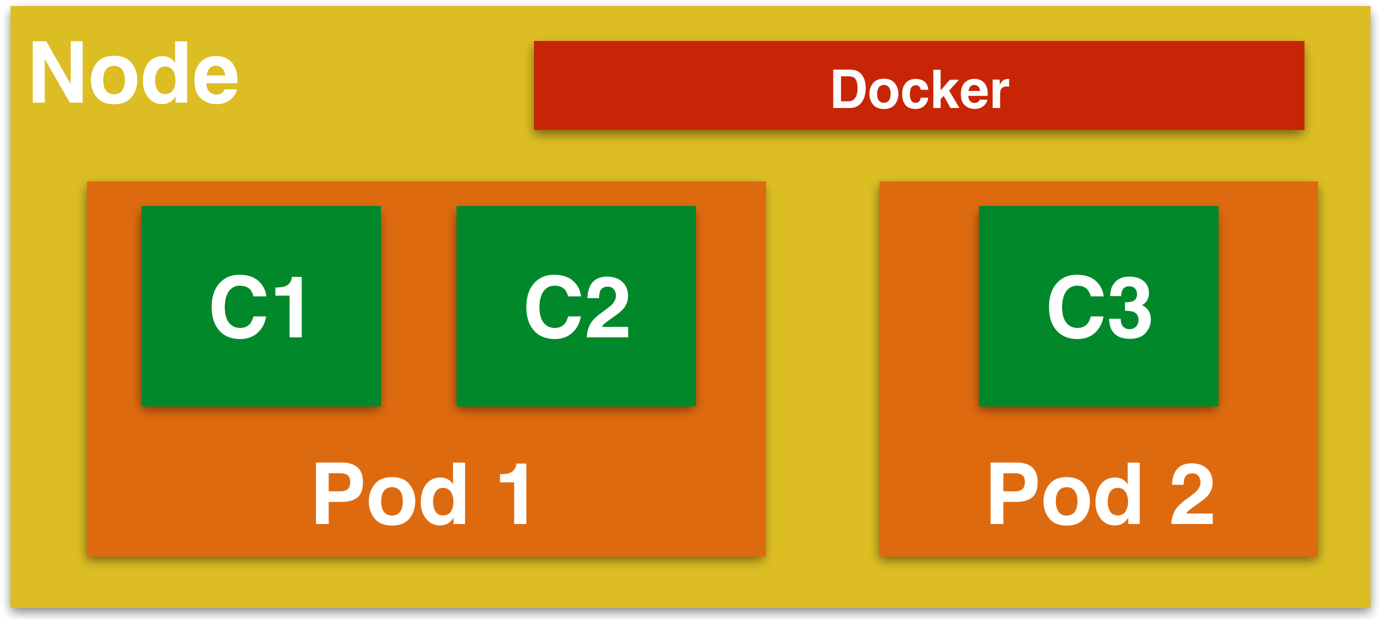


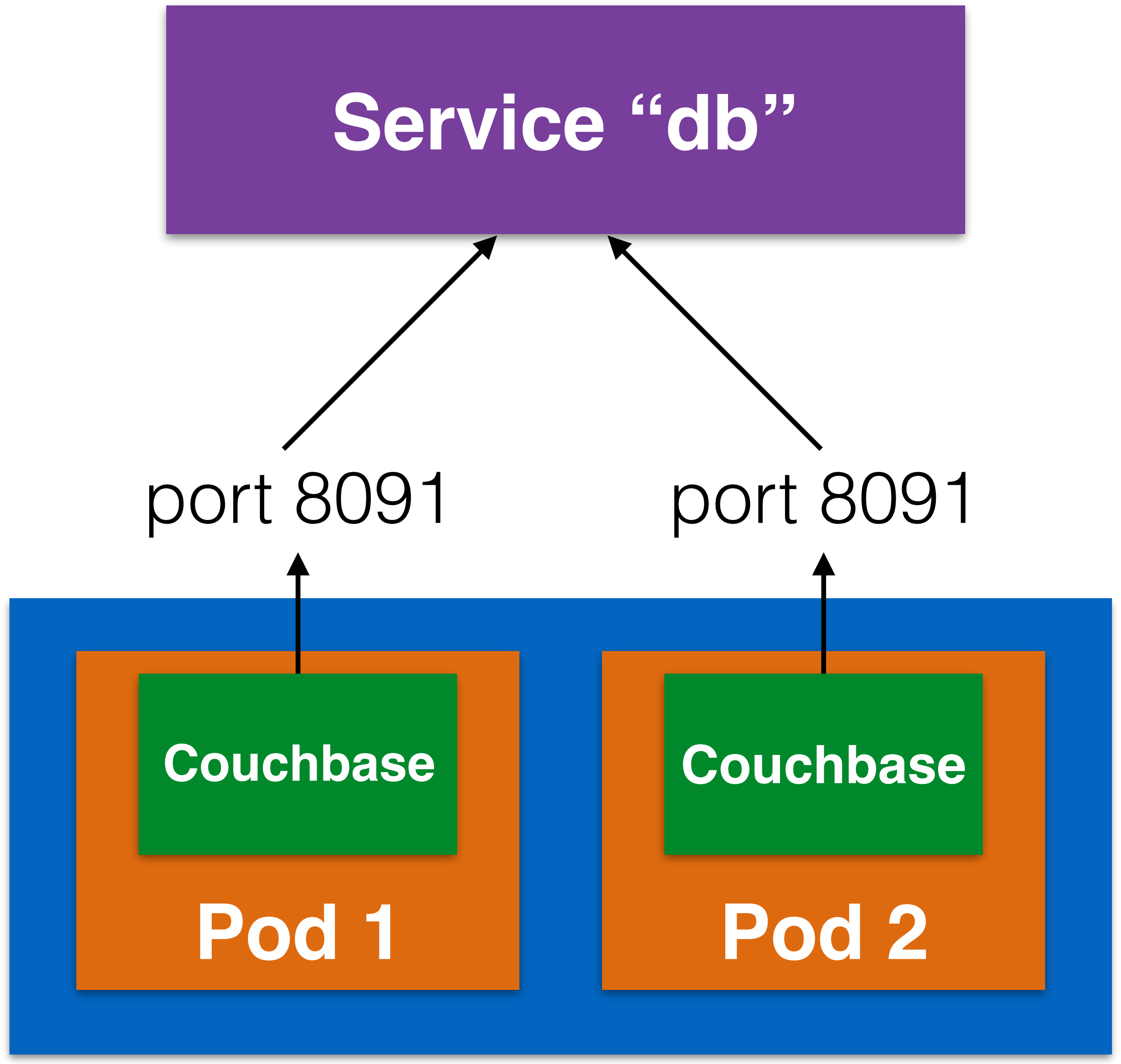
```
version: "3"
services:
  db:
    image: arungupta/couchbase:travel
    ports:
      - 8091:8091
      - 8092:8092
      - 8093:8093
      - 11210:11210
  web:
    image: arungupta/wildfly-couchbase-javaee:travel
    environment:
      - COUCHBASE_URI=db
    ports:
      - 8080:8080
      - 9990:9990
```

```
docker stack deploy --compose-file=docker-compose.yml webapp
```

Kubernetes Concepts

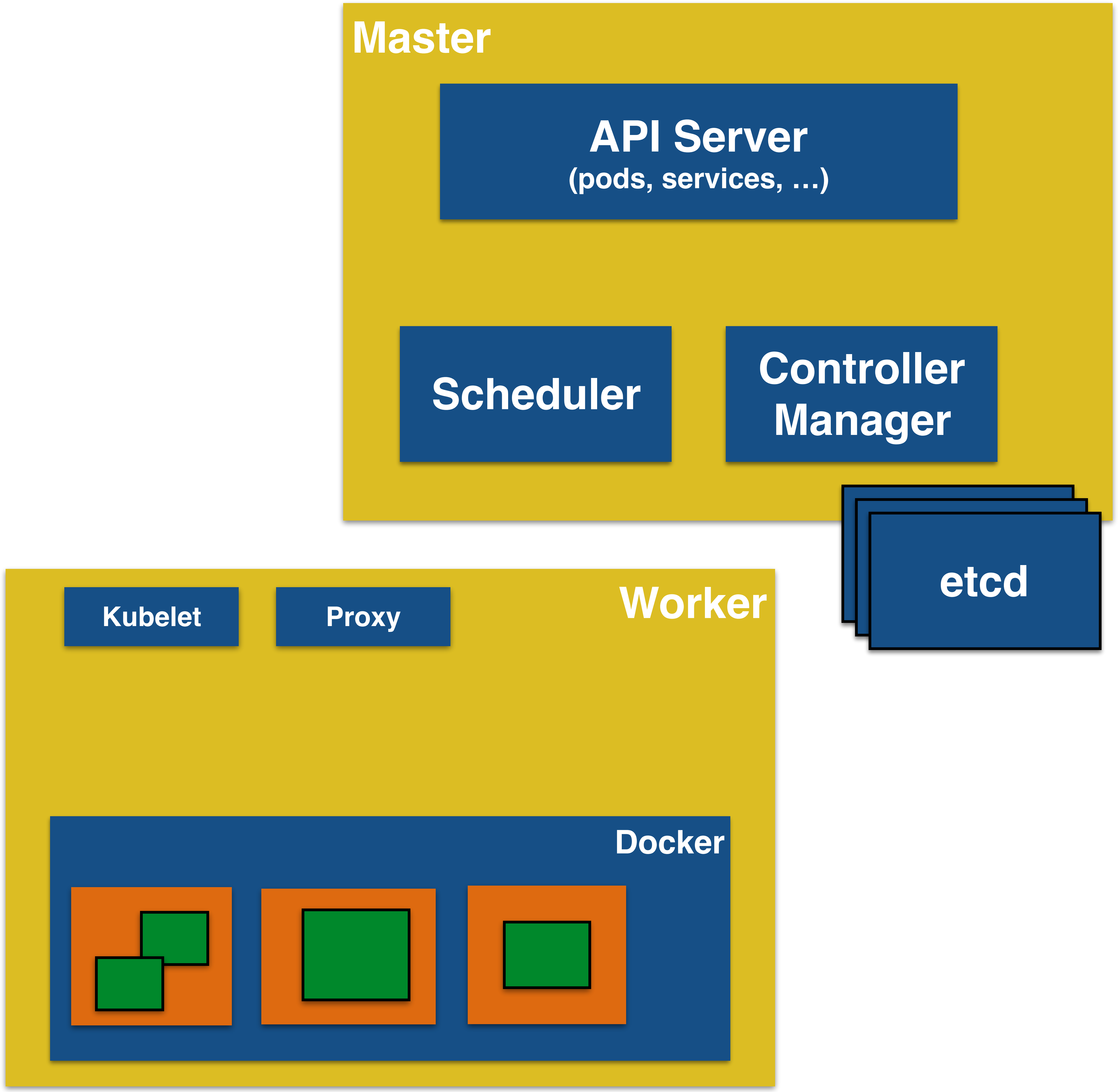
- **Pods**: collocated group of Docker containers that share an IP and storage volume
- **Service**: Single, stable name for a set of pods, also acts as LB
- **Label**: used to organize and select group of objects
- **Replica Set**: manages the lifecycle of pods and ensures specified number are running

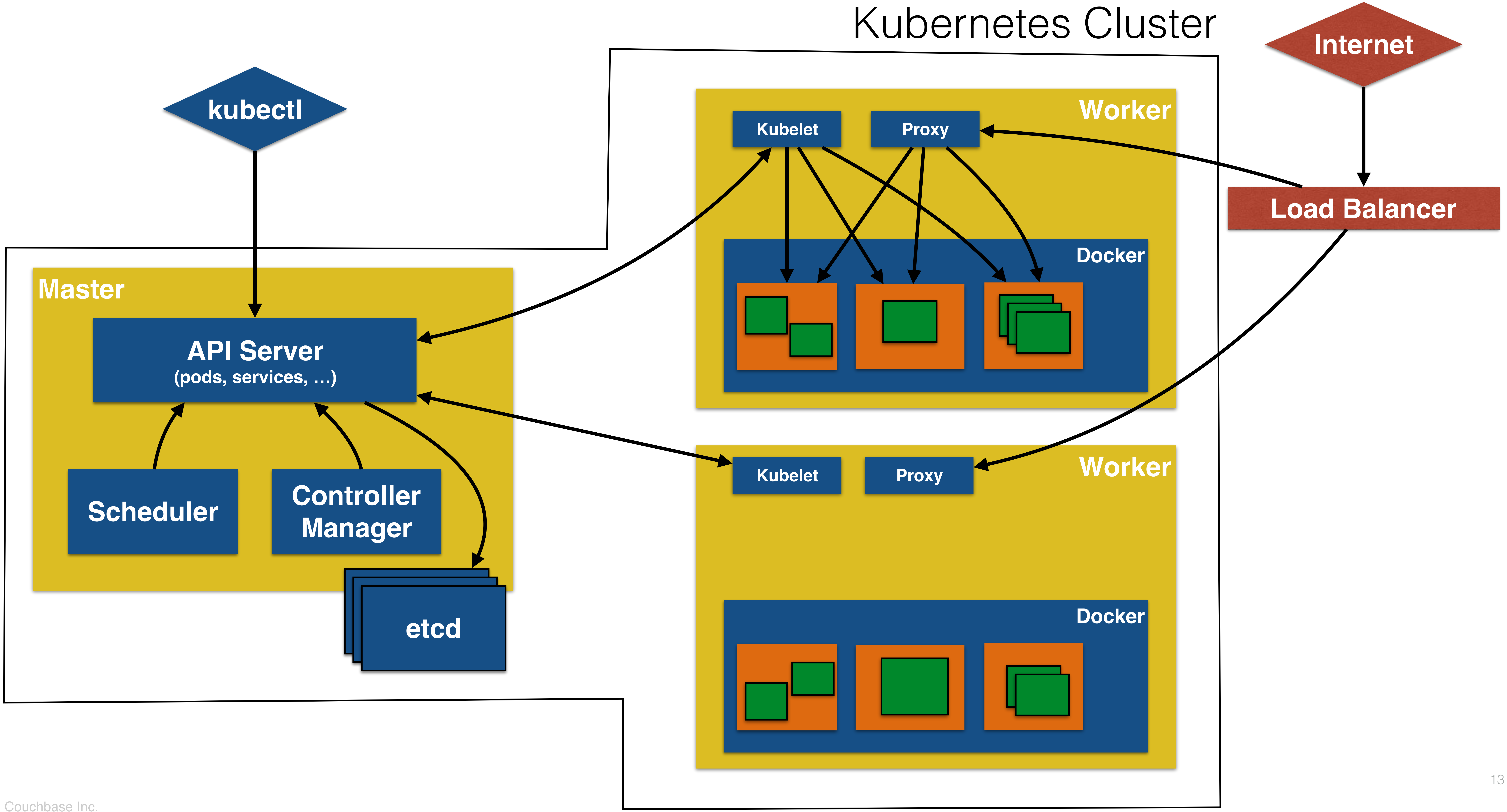




Core Concepts: Kubernetes

- **Node**: Machine or VM in the cluster
- **Master**: Central control plane, provides unified view of the cluster
  - **etcd**: distributed key-value store used to persist Kubernetes system state
- **Worker**: Docker host running *kubelet* (node agent) and *proxy* services
  - Runs pods and containers
  - Monitored by *systemd* (CentOS) or *monit* (Debian)





Kubernetes Cluster

Internet

Load Balancer

kubectl

Master

API Server
(pods, services, …)

Scheduler

Controller Manager

etcd

Worker

Kubelet

Proxy

Docker

Worker

Kubelet

Proxy

Docker

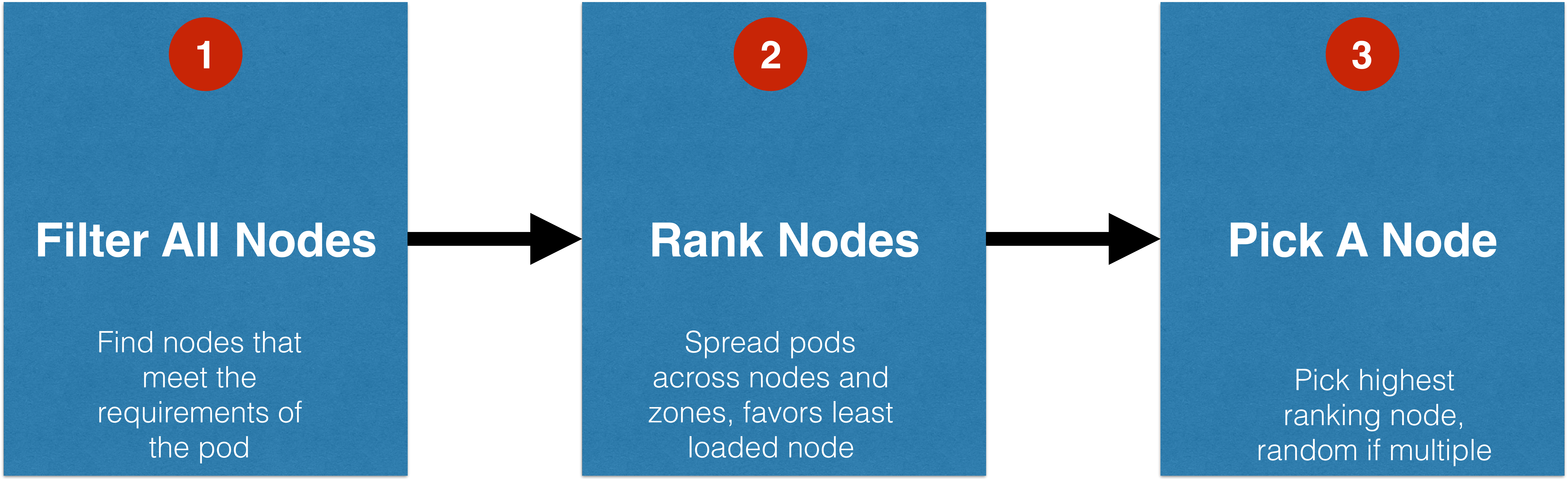
Development using Kubernetes

- Single node cluster
  - minikube
- Multi-node cluster
  - kops
  - kube-aws (CoreOS + AWS)
  - kube-up (deprecated)
  - Google Cloud, Azure, Tectonic, …

kubectl

- Controls the Kubernetes cluster manager
- `kubectl get pods` or `minions`
- `kubectl create -f <filename>`
- `kubectl update` or `delete`
- `kubectl resize –replicas=3 replicaset <name>`

Kubernetes Scheduling Algorithm



1

Filter All Nodes

Find nodes that meet the requirements of the pod

2

Rank Nodes

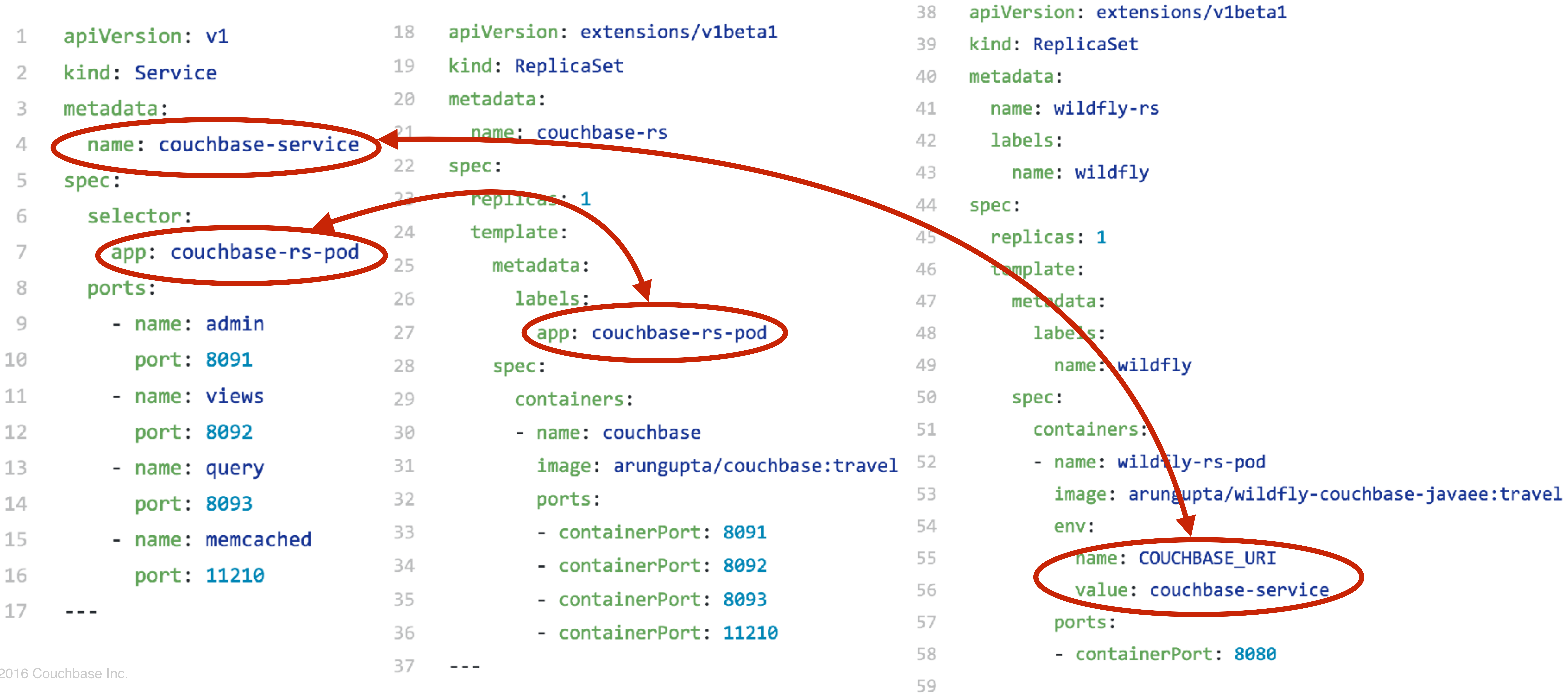
Spread pods across nodes and zones, favors least loaded node

3

Pick A Node

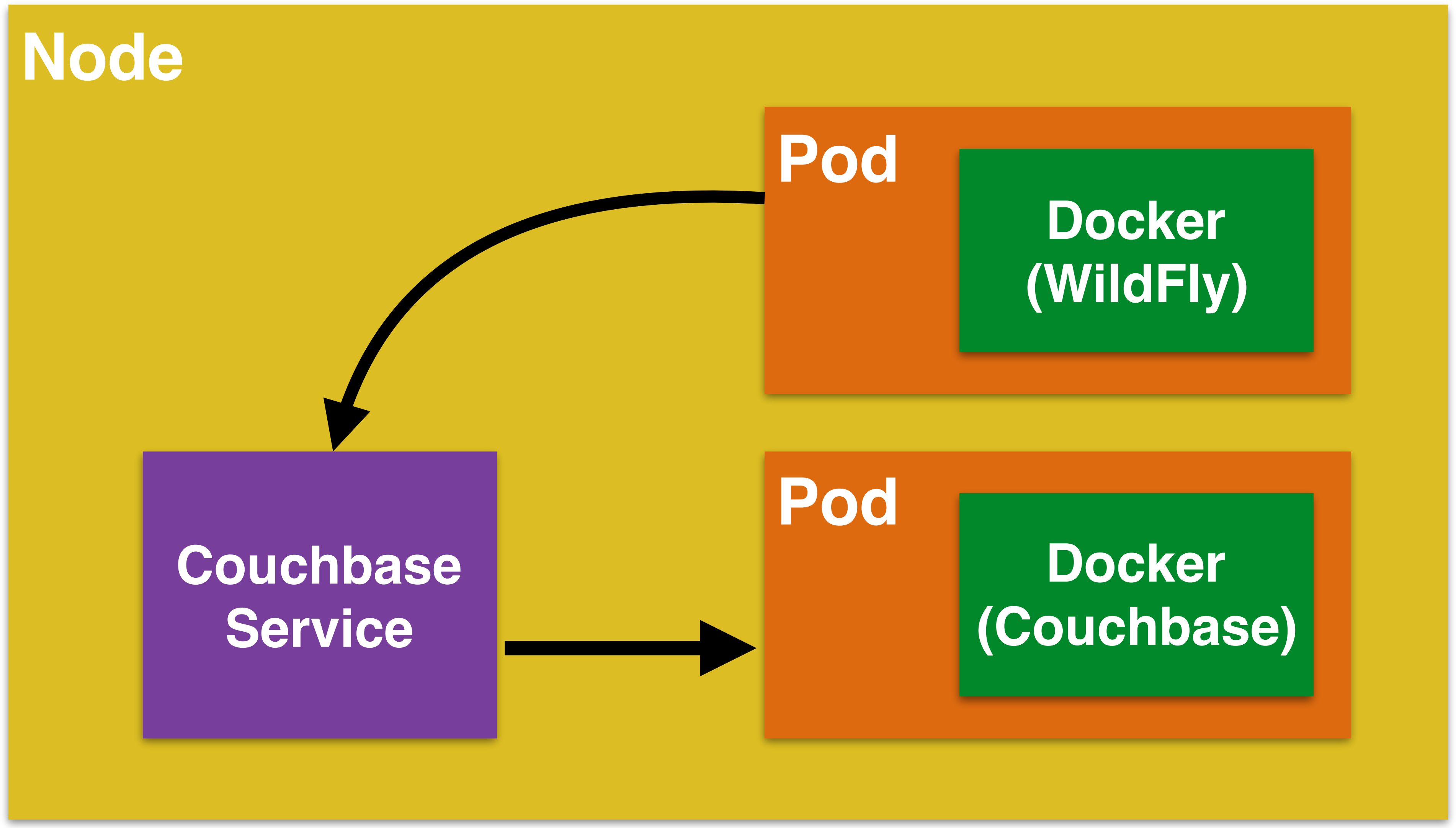
Pick highest ranking node, random if multiple

Kubernetes Configuration File

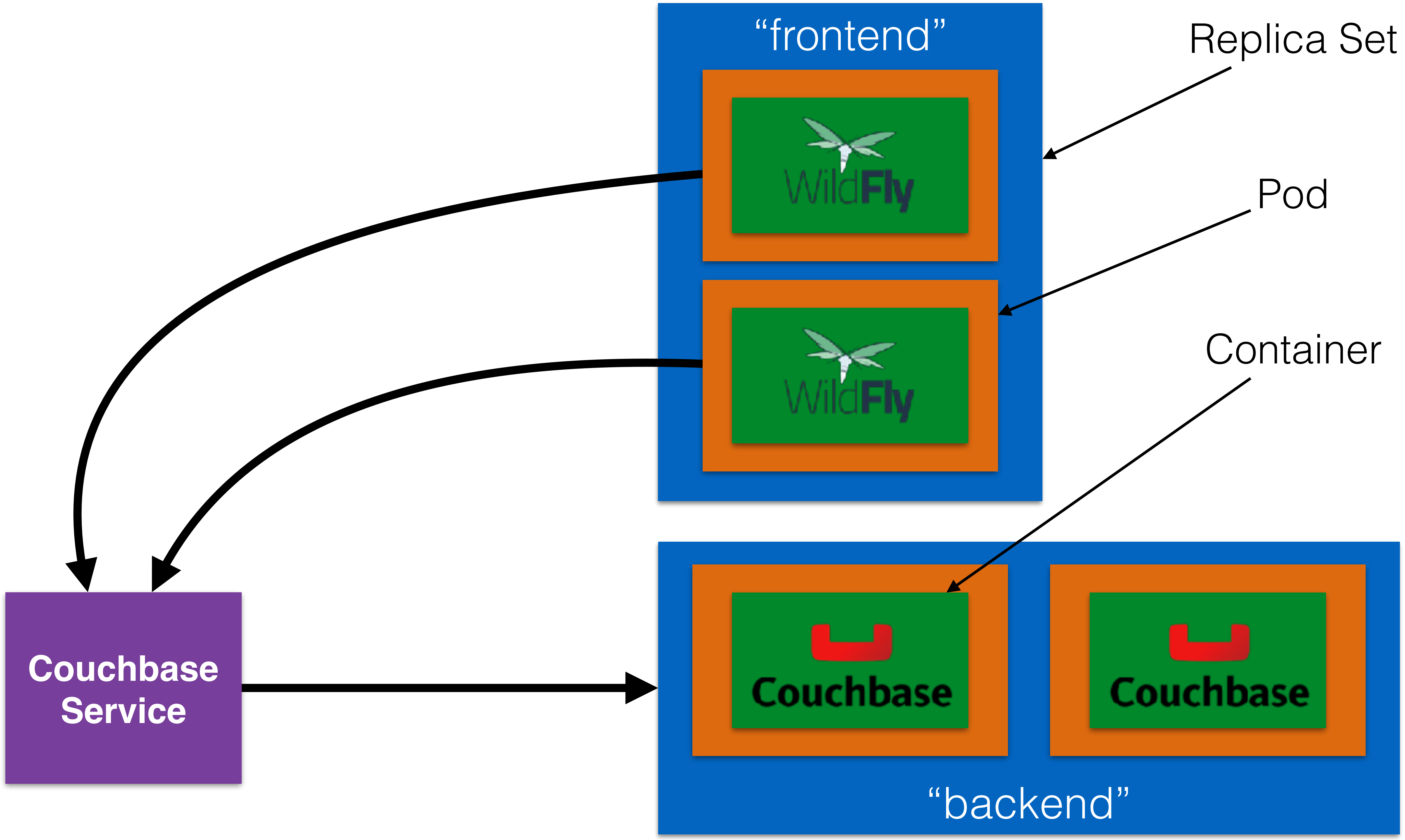


```yaml
apiVersion: v1
kind: Service
metadata:
  name: couchbase-service
spec:
  selector:
    app: couchbase-rs-pod
  ports:
    - name: admin
      port: 8091
    - name: views
      port: 8092
    - name: query
      port: 8093
    - name: memcached
      port: 11210
---
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: couchbase-rs
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: couchbase-rs-pod
    spec:
      containers:
      - name: couchbase
        image: arungupta/couchbase:travel
        ports:
        - containerPort: 8091
        - containerPort: 8092
        - containerPort: 8093
        - containerPort: 11210
---
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: wildfly-rs
  labels:
    name: wildfly
spec:
  replicas: 1
  template:
    metadata:
      labels:
        name: wildfly
    spec:
      containers:
      - name: wildfly-rs-pod
        image: arungupta/wildfly-couchbase-javaee:travel
        env:
        - name: COUCHBASE_URI
          value: couchbase-service
        ports:
        - containerPort: 8080
```

Services


Node
Pod
Docker (WildFly)
Couchbase Service
Pod
Docker (Couchbase)

Services



"frontend"
Replica Set
Pod
Container
WildFly
WildFly
Couchbase Service
Couchbase
Couchbase
"backend"

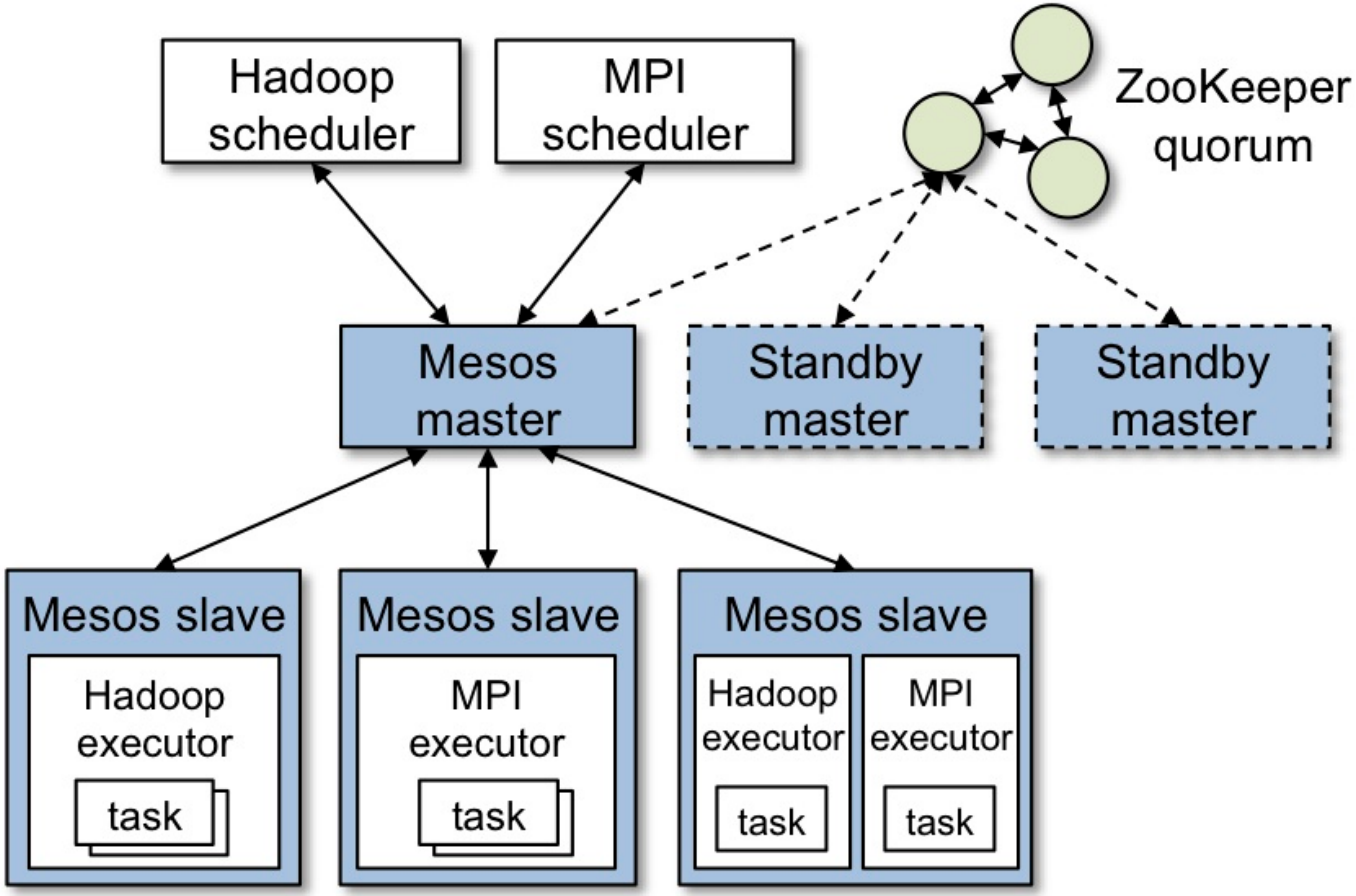
Mesos

- Open source cluster manager
- Developed at UC Berkeley
- Provides resource isolation and sharing across distributed applications
- Run distributed systems on the same pool of nodes
  - Hadoop, Spark, Jenkins, Couchbase, …
- Cluster monitoring
- Tasks isolated via Linux containers

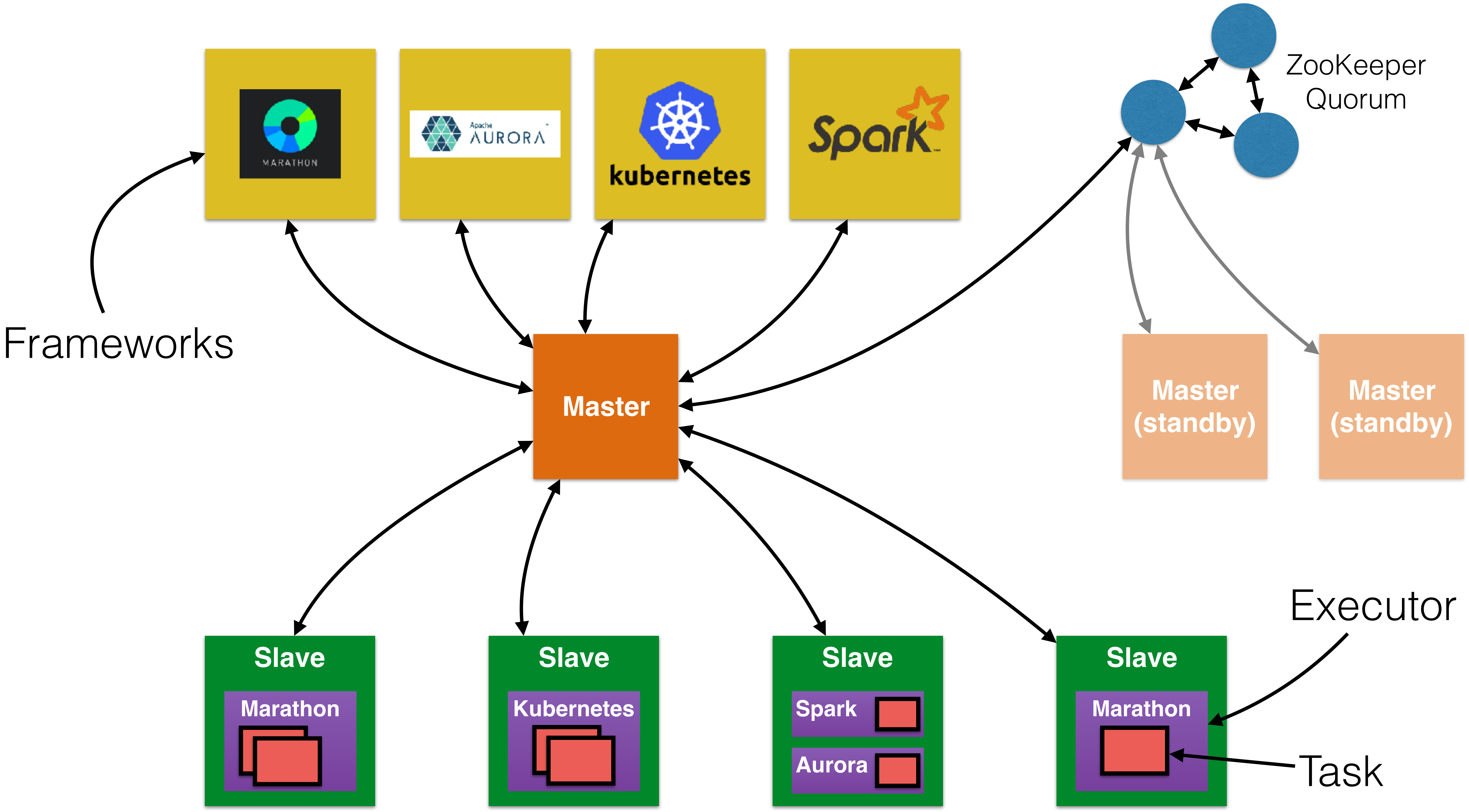
Mesos

- Master Slave architecture
- Fault tolerant
  - Leader election via ZooKeeper
- Multi platform
  - Ubuntu, Mac OS, CentOS

Mesos Architecture



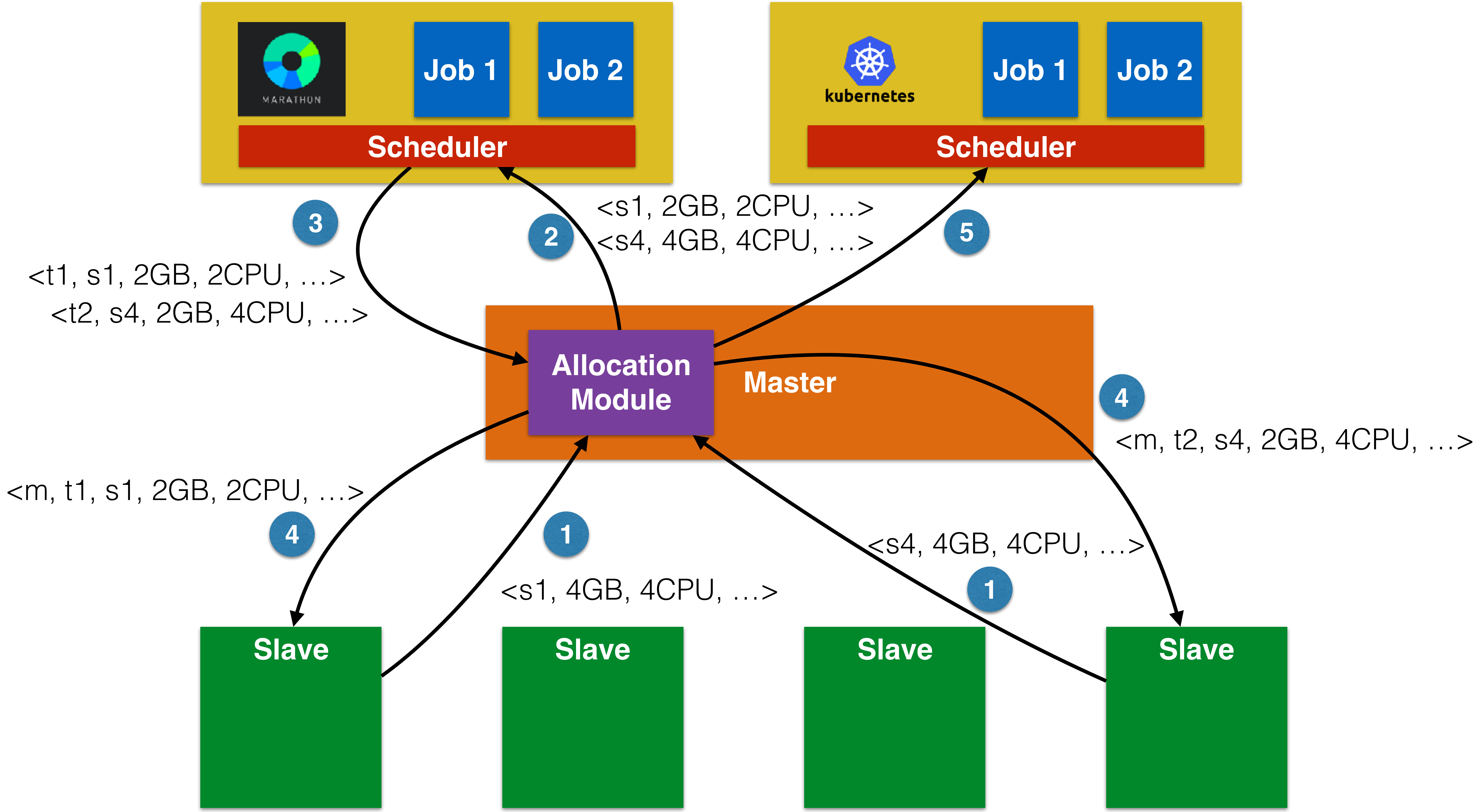
http://mesos.apache.org/documentation/latest/architecture/



http://mesos.apache.org/documentation/latest/architecture/

Frameworks

- Frameworks are targeted at a use case and domain-specific
  - Master node “offers” resources to each framework
  - Framework “accepts” the offer and execute applications
- Framework has “scheduler” and “executor”
  - Scheduler registers with the master for “offer”
  - Executor launched on slave nodes to run the task
    - Passes a description of the task to run



http://mesos.apache.org/documentation/latest/architecture/

EC2 Container Service

Thanks!

Arun Gupta, @arungupta
github.com/javaee-samples/docker-java/tree/master/slides