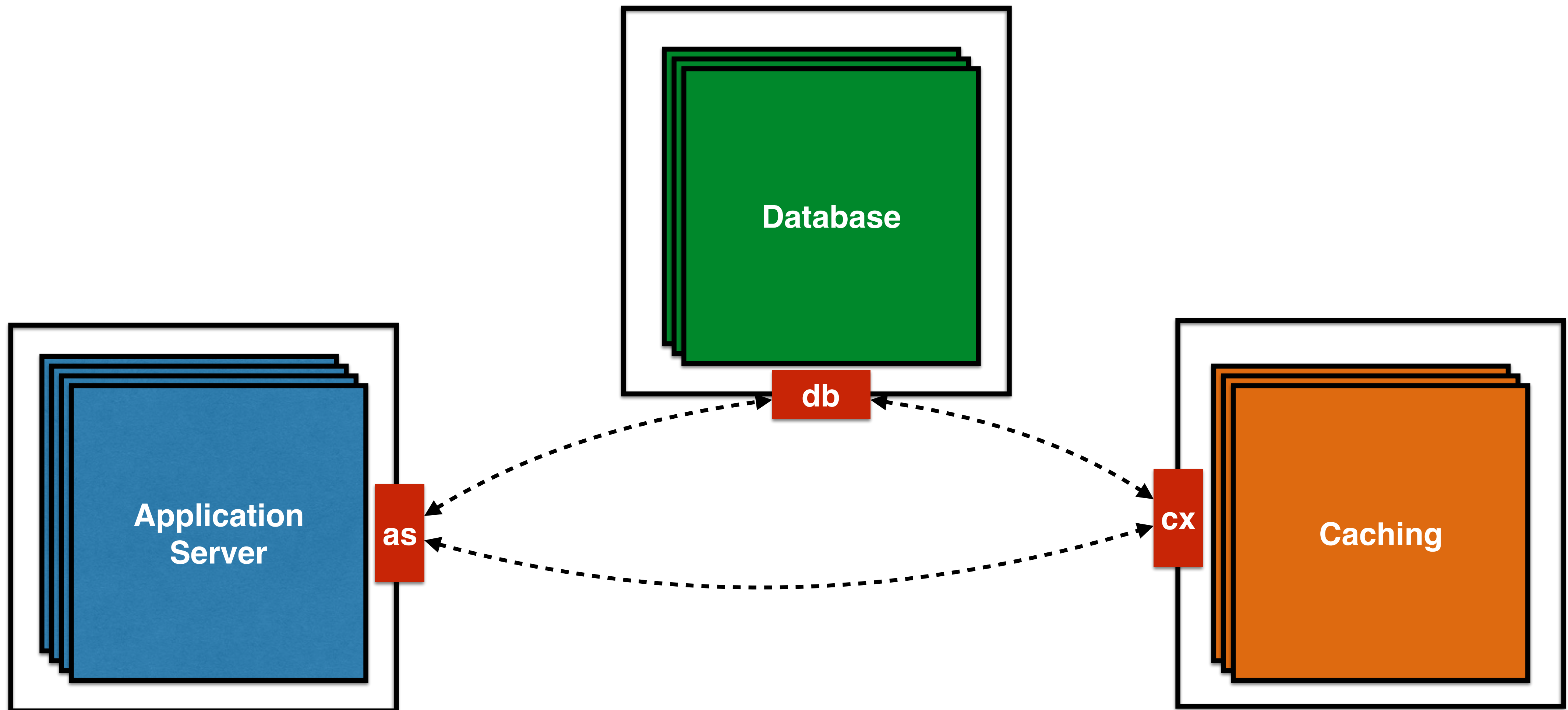




Service Discovery using Docker and Kubernetes

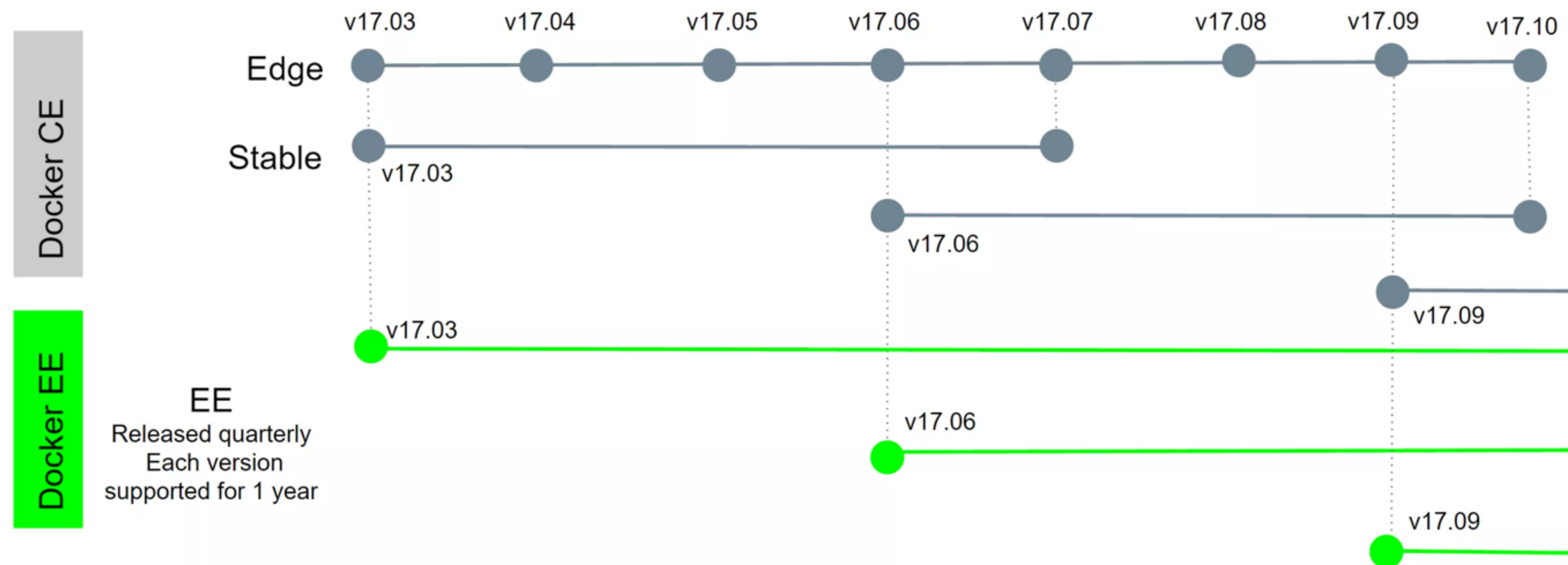
Arun Gupta, @arungupta

Why Service Discovery?

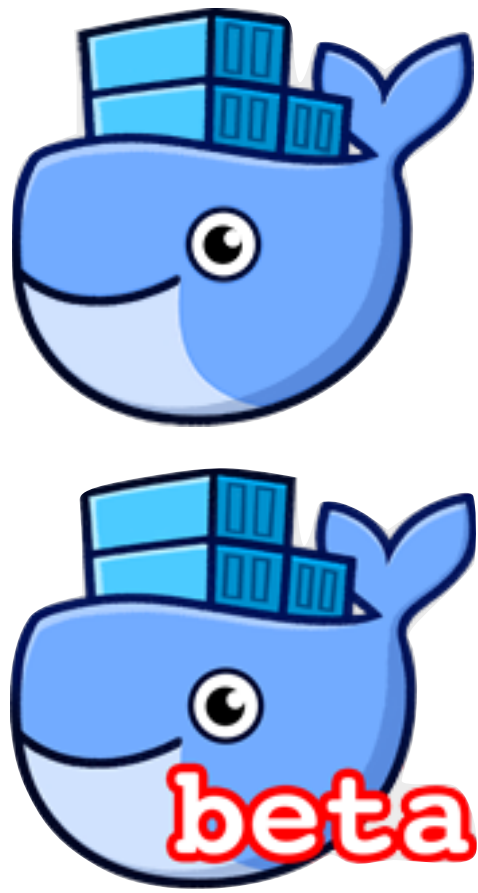


Development using Docker

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



Docker for Mac/Windows



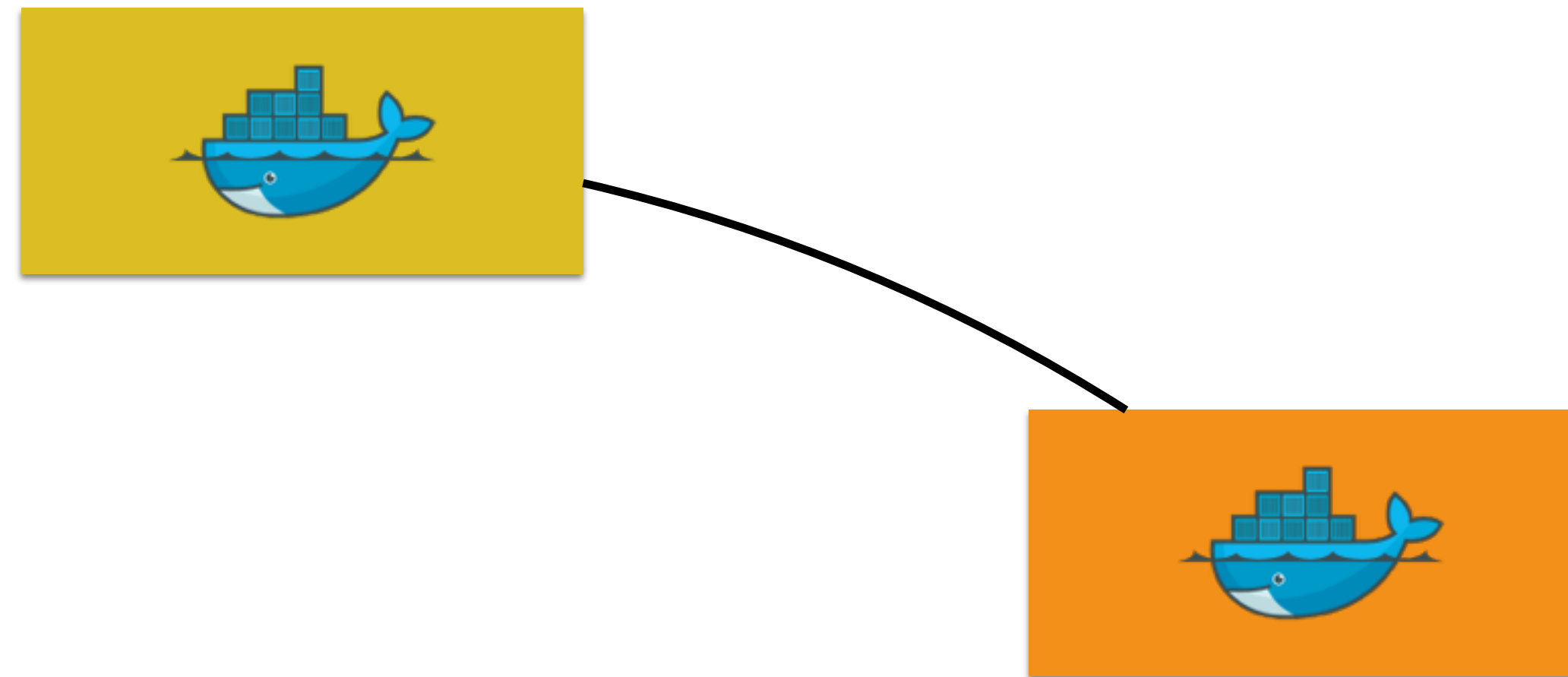
- Native application and UI
- Auto update capability
- No additional software required, e.g. VirtualBox
 - OSX: xhyve VM using `Hypervisor.framework`
 - Windows: Hyper-V VM
- Download: docker.com/getdocker
- Requires Yosemite 10.10+ or Windows 10 64-bit

Swarm-mode: Initialize



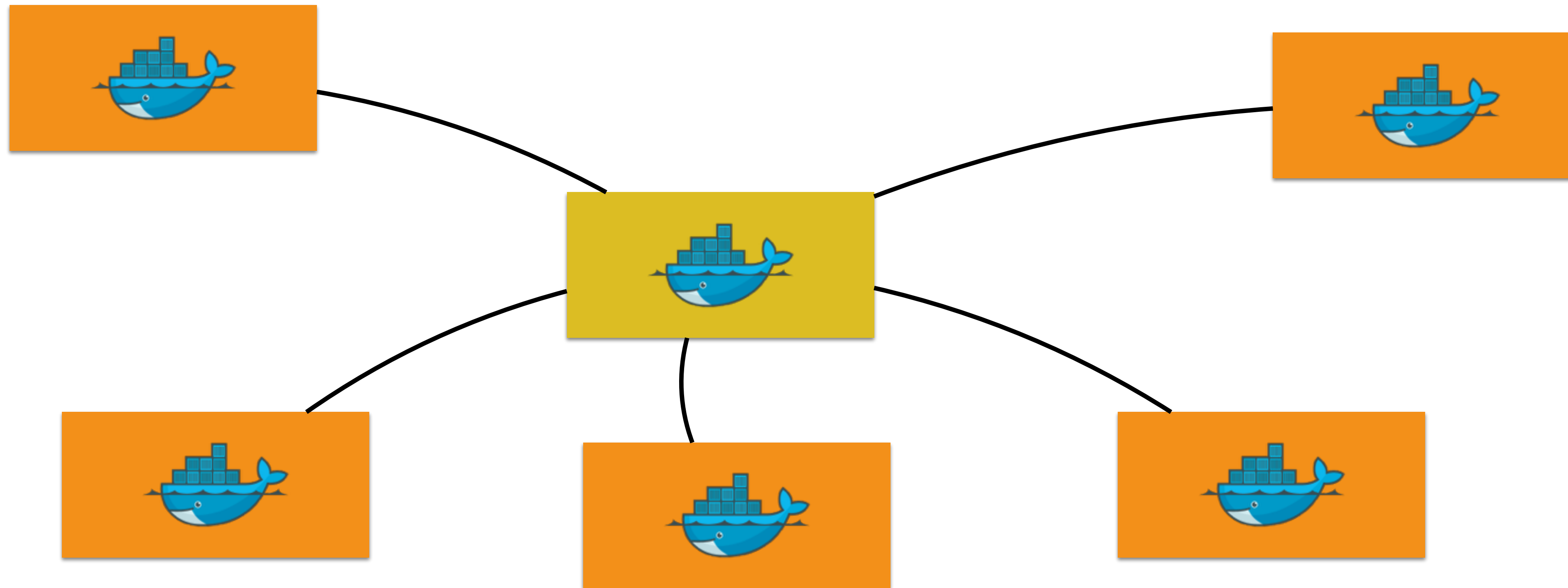
```
docker swarm init --listen-addr <ip>:2377 --secret <SECRET>
```

Swarm-mode: Add Worker



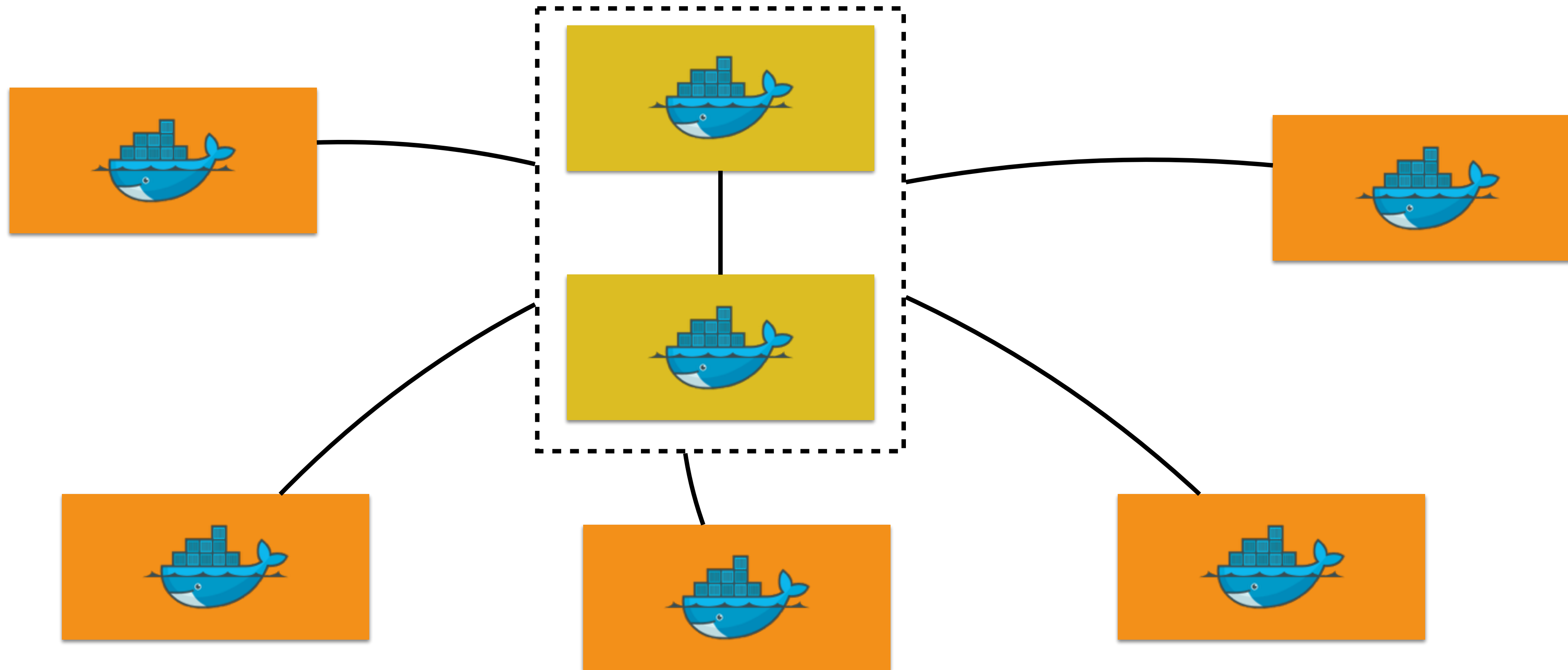
```
docker swarm join --secret <SECRET> <manager>:2377
```


Swarm-mode: Add More Workers



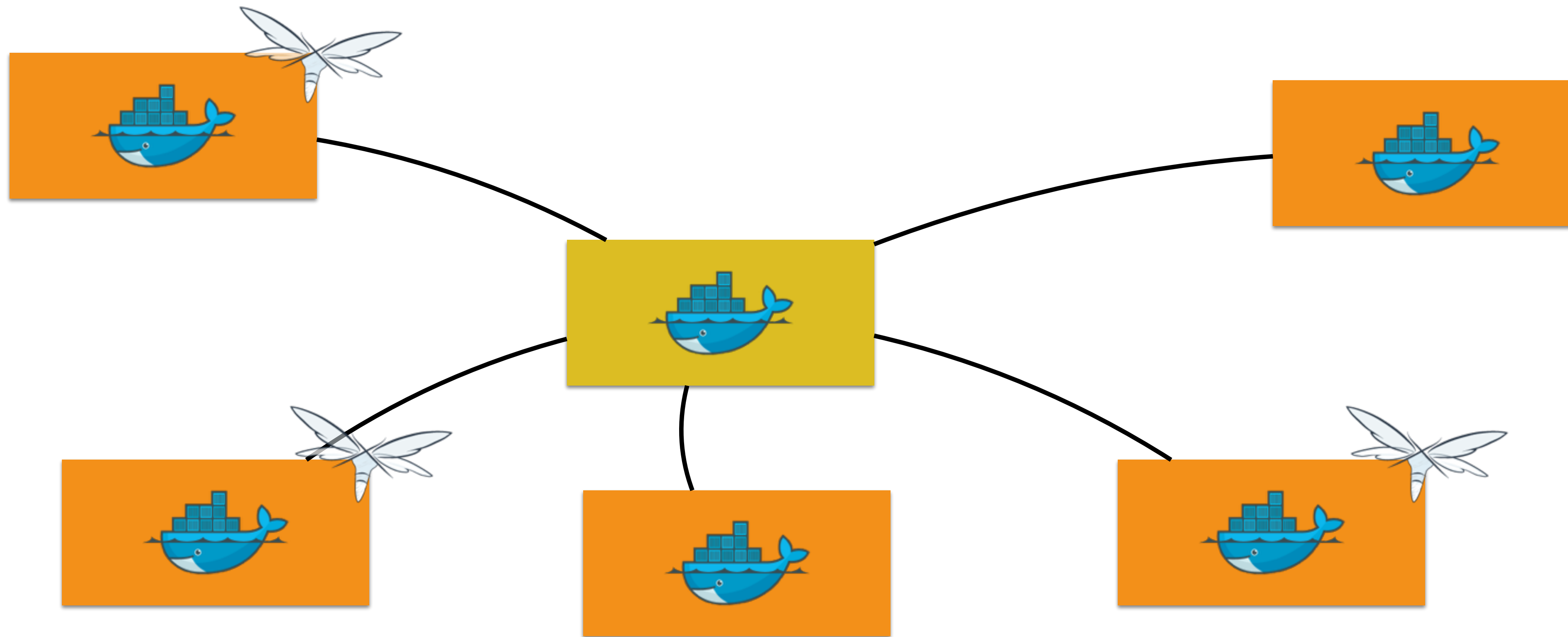
```
docker swarm join --secret <SECRET> <manager>:2377
```

Swarm-mode: Primary/Secondary Master



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


Swarm-mode: Replicated Service

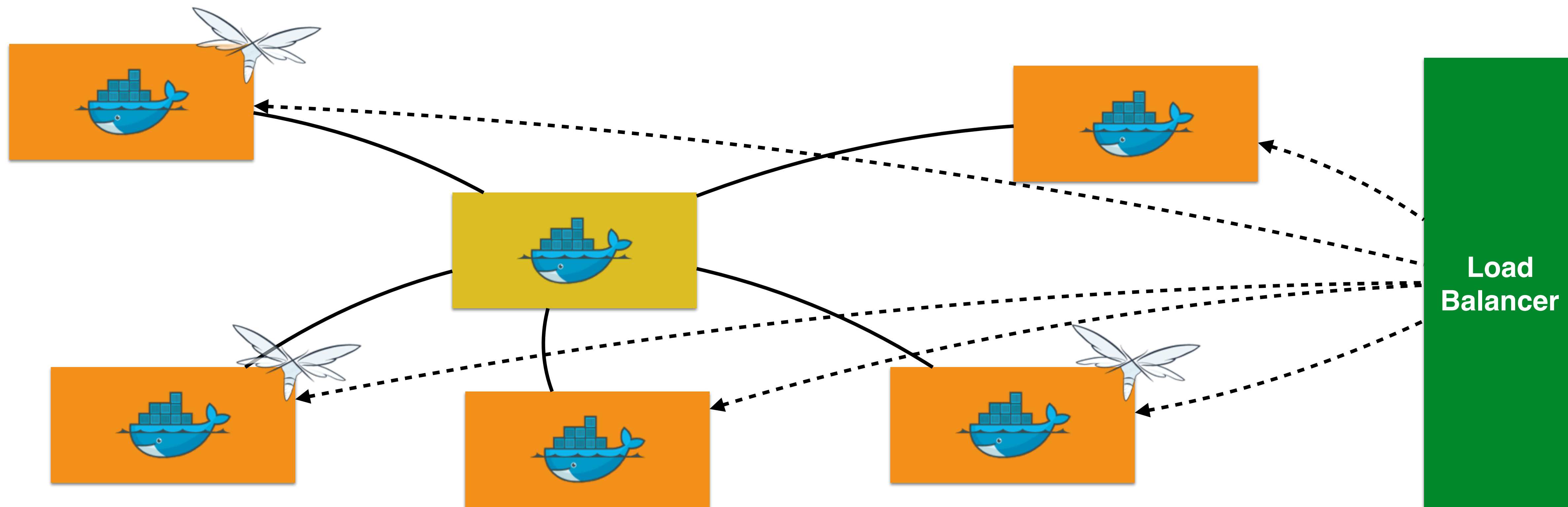


```
docker service create --replicas 3 --name web jboss/wildfly
```

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
```

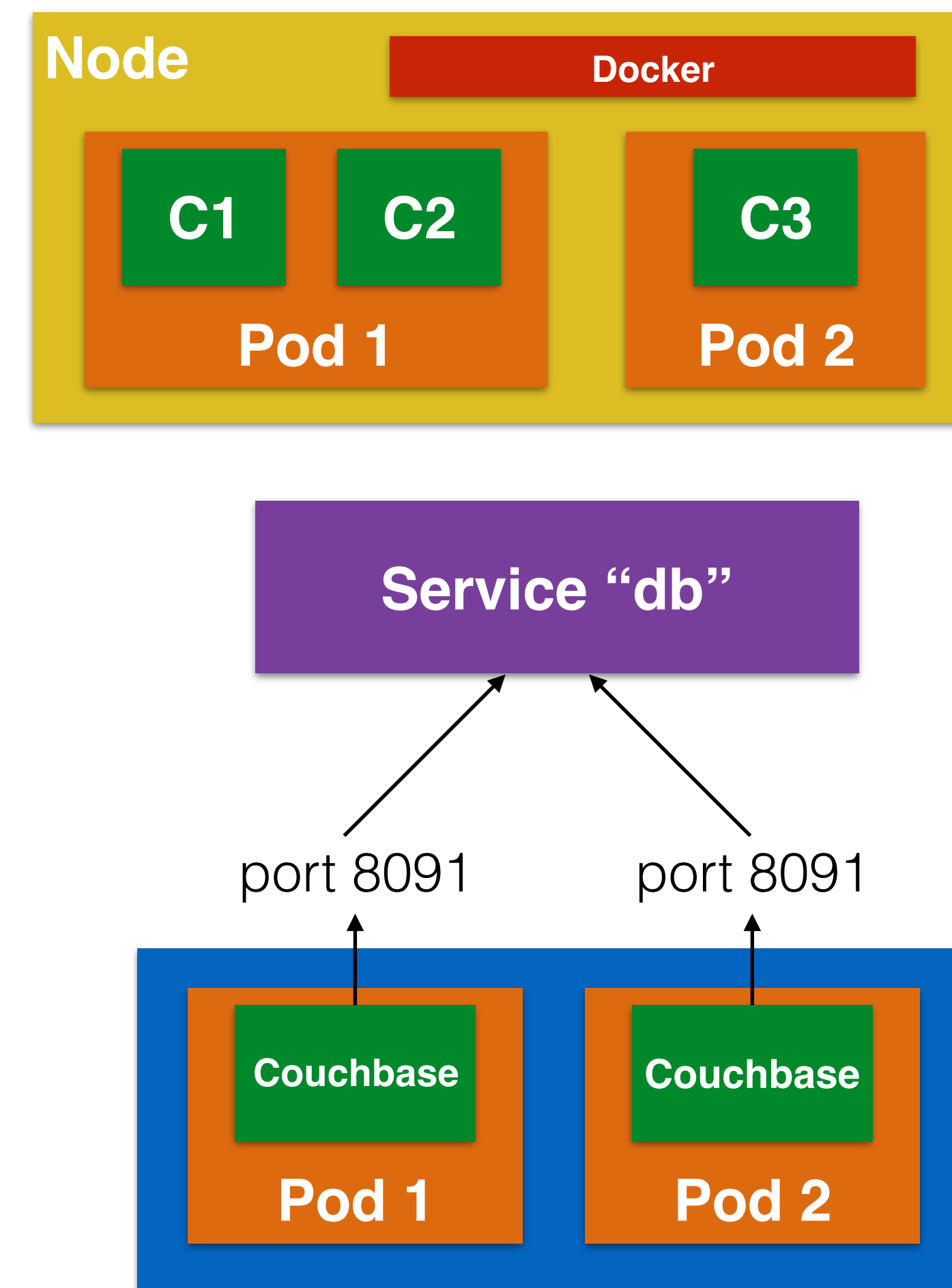
Service Discovery with Docker

```
1  version: "3"
2  services:
3    db:
4      image: arungupta/couchbase:travel
5      ports:
6        - 8091:8091
7        - 8092:8092
8        - 8093:8093
9        - 11210:11210
10   web:
11     image: arungupta/wildfly-couchbase-javaee:travel
12     environment:
13       - COUCHBASE_URI=db
14     ports:
15       - 8080:8080
16       - 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



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>`

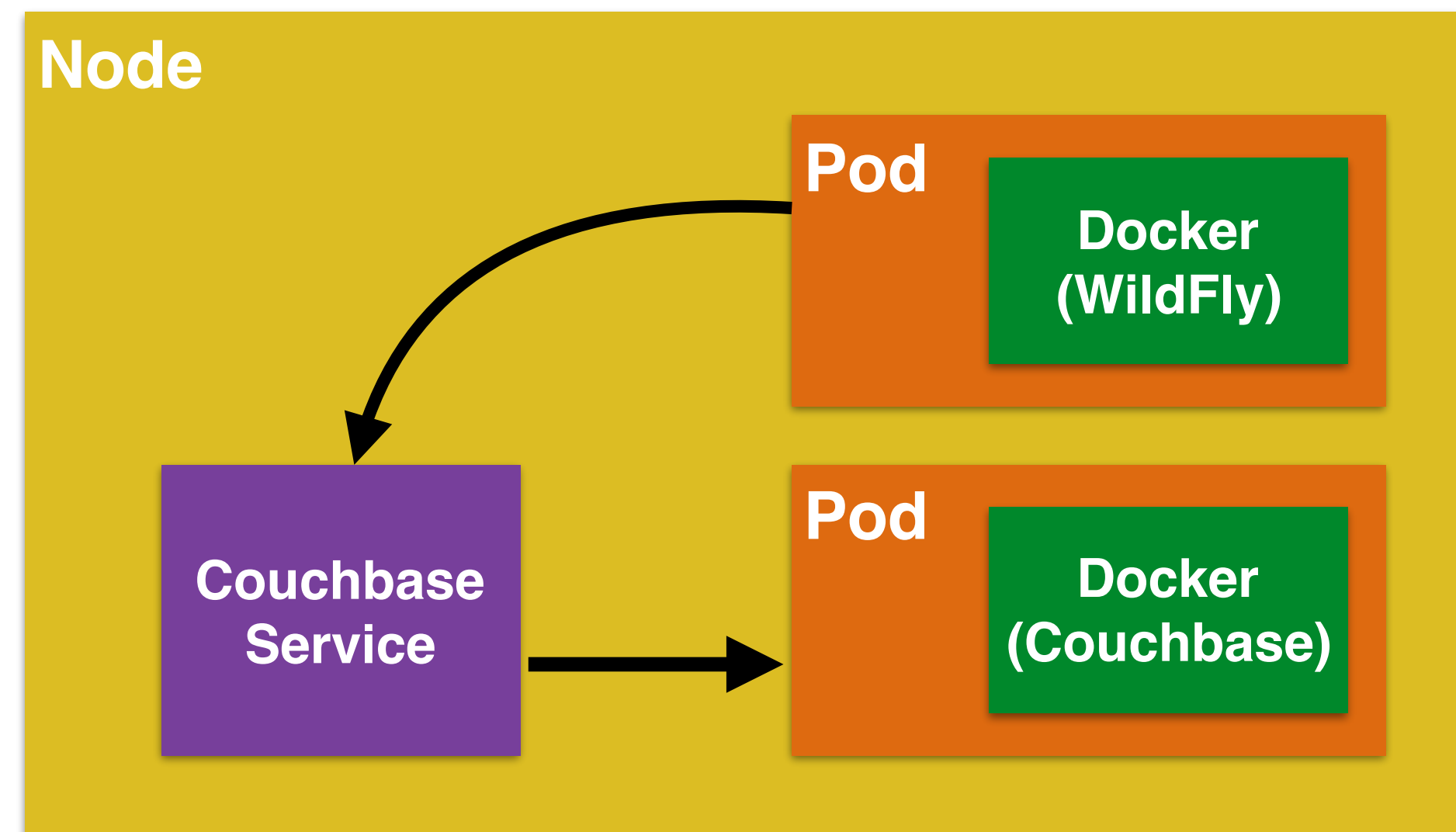
Kubernetes Configuration File

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4  name: couchbase-service
5  spec:
6  selector:
7  app: couchbase-rs-pod
8  ports:
9    - name: admin
10     port: 8091
11    - name: views
12     port: 8092
13    - name: query
14     port: 8093
15    - name: memcached
16     port: 11210
17  ---
18  apiVersion: extensions/v1beta1
19  kind: ReplicaSet
20  metadata:
21  name: couchbase-rs
22  spec:
23  replicas: 1
24  template:
25    metadata:
26    labels:
27    app: couchbase-rs-pod
28  spec:
29    containers:
30    - name: couchbase
31      image: arungupta/couchbase:travel
32      ports:
33        - containerPort: 8091
34        - containerPort: 8092
35        - containerPort: 8093
36        - containerPort: 11210
37  ---
38  apiVersion: extensions/v1beta1
39  kind: ReplicaSet
40  metadata:
41  name: wildfly-rs
42  labels:
43  name: wildfly
44  spec:
45  replicas: 1
46  template:
47    metadata:
48    labels:
49    name: wildfly
50  spec:
51    containers:
52    - name: wildfly-rs-pod
53      image: arungupta/wildfly-couchbase-javaee:travel
54      env:
55      name: COUCHBASE_URI
56      value: couchbase-service
57    ports:
58    - containerPort: 8080
59  ---
```

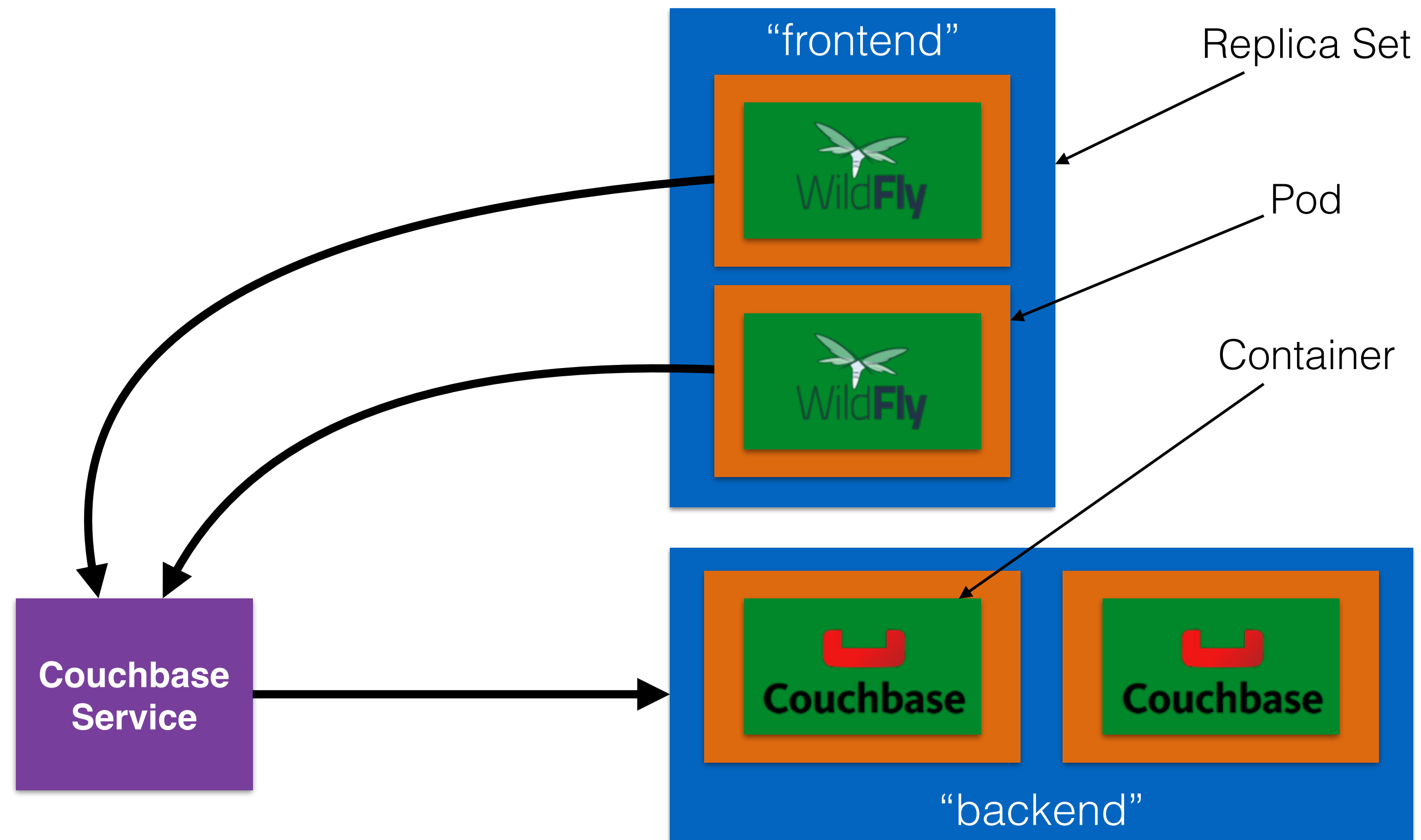
Diagram illustrating the configuration dependencies between the three Kubernetes manifests:

- The `Service` manifest (lines 1-17) defines `name: couchbase-service` (line 4).
- The `ReplicaSet` manifest for `couchbase-rs` (lines 18-37) uses `app: couchbase-rs-pod` (line 27) in its template labels, which selects the `Service` defined in the first manifest.
- The `ReplicaSet` manifest for `wildfly-rs` (lines 38-59) uses `name: COUCHBASE_URI` (line 55) with `value: couchbase-service` (line 56) in its environment variables, which refers to the `Service` defined in the first manifest.

Services



Services



References

- Docker: github.com/docker/labs/tree/master/developer-tools/java
- Kubernetes: github.com/arun-gupta/kubernetes-java-sample

ONLINE COURSE



Arun Gupta

KUBERNETES FOR JAVA DEVELOPERS

September 23 | 10AM-12PM

bit.ly/kubejava

O'REILLY®

Docker for Java Developers

Package, Deploy, and Scale with Ease



Arun Gupta

bit.ly/dockerjava



Thanks!

Arun Gupta, @arungupta

github.com/javaee-samples/docker-java/tree/master/slides