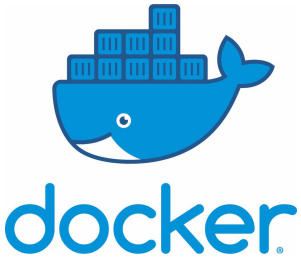
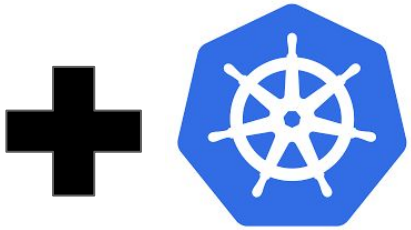
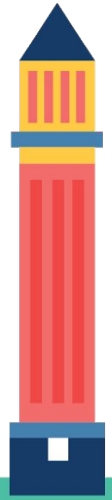
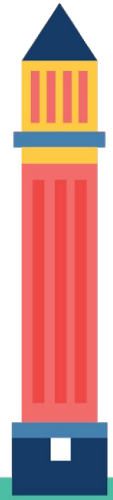




Docker Container Networking

How to Publish and Secure applications in the Docker Enterprise Platform



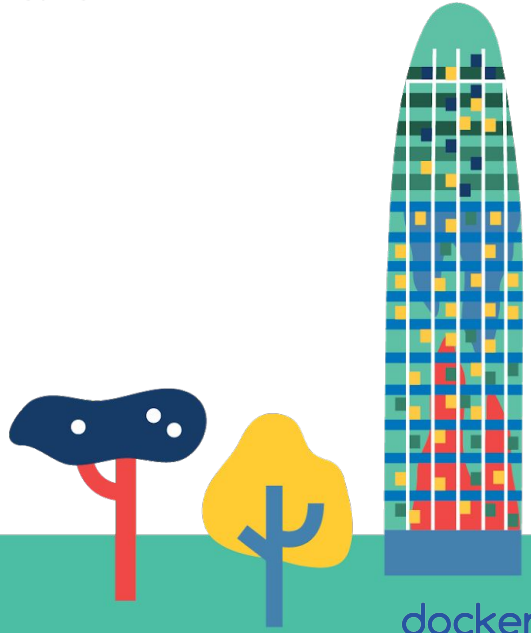


Javier Ramirez

Senior Consultant, Hopla Software
@frjaraur

Guillaume Morini

Solution Engineer, Docker
@GuillaumeMorini

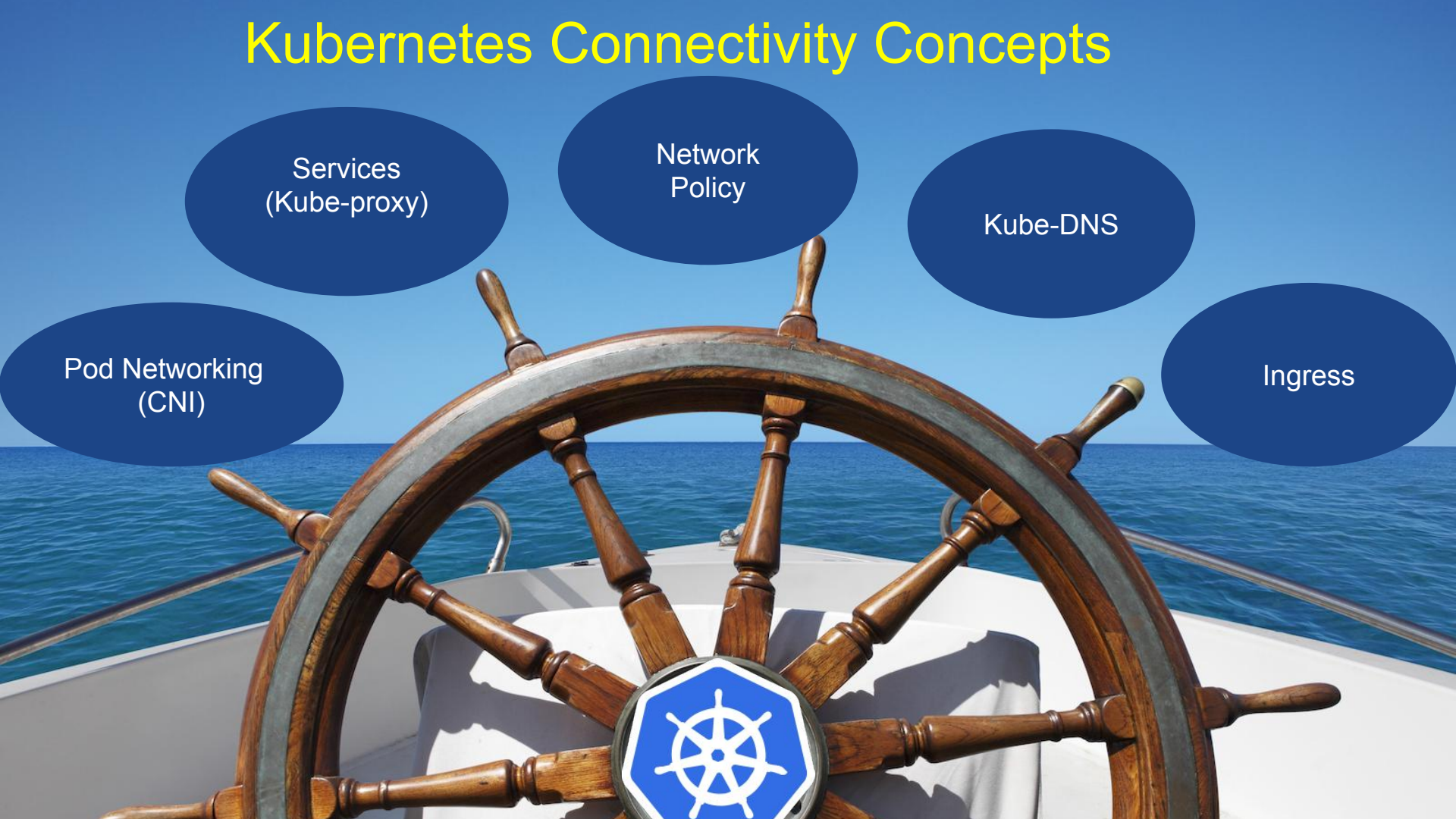


Goal of this Session

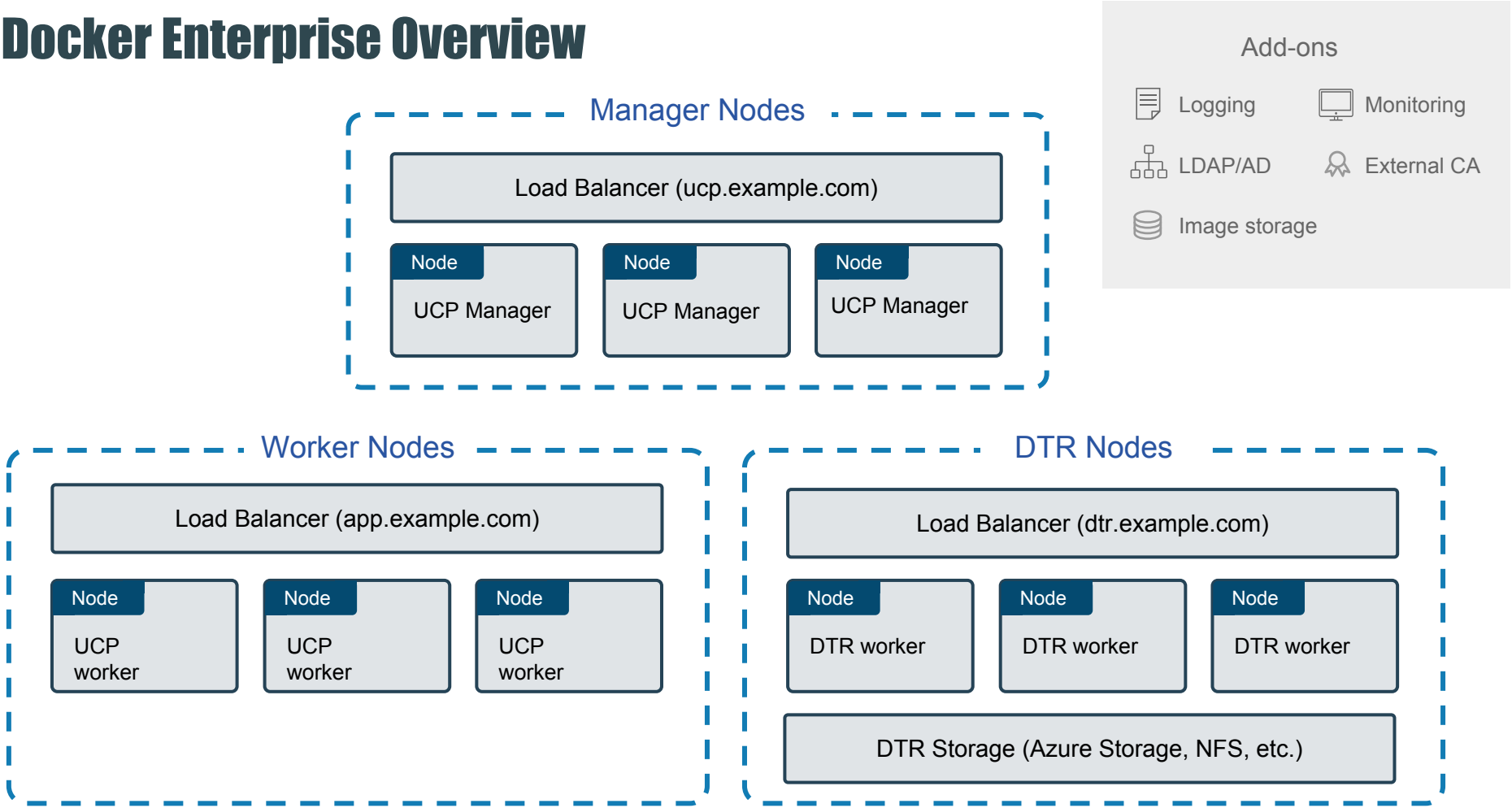
- Basics of K8S Networking
- Docker Enterprise + Calico Integration Overview
- Application Publishing with Docker Enterprise
- Takeaways
- Q&A

Let's Start with K8S Connectivity Basics

Kubernetes Connectivity Concepts



Docker Enterprise Overview



TIGERA CALICO: WHY IT'S AWESOME

PROJECT CALICO

Secure networking for the cloud-native era

Open source, maintained by Tigera with hundreds of third party contributors

Batteries-included Container networking for Docker Enterprise Kubernetes

> Scalable, distributed control plane

> Policy-driven network security

> No overlay required

> Integrated with all major cloud platforms

> Widely deployed, proven at scale

Heterogeneous Infrastructure

- → No Underlay Dependency = No Lock-In
- → Simple Zero-Touch Provisioning
- → Any Infrastructure, Any Cloud
  - On-Prem (VM, Bare)
  - Cloud (AWS, Azure, GCP)
  - Hybrid

Google Cloud Platform

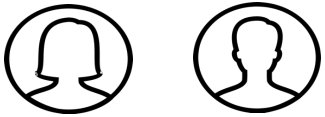
Diverse Application Portfolio

Connectivity Concept	Out-of-the-Box Solution with Docker Enterprise 2.1
Pod - Pod	Calico CNI
Services	ClusterIP NodePort LoadBalancer
Ingress	NGINX Ingress Controller
DNS	kube-dns
K8s Network Policy	Calico

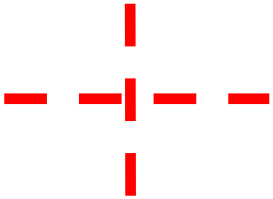
*Tigera CNX builds on Calico with enterprise security features: Hierarchical Policies, Policy RBAC, DevSecOps tools (Audit, Alerting, Compliance), etc.

Zero-Trust Security

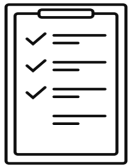
Stage/tier separation



Tenant/namespace isolation



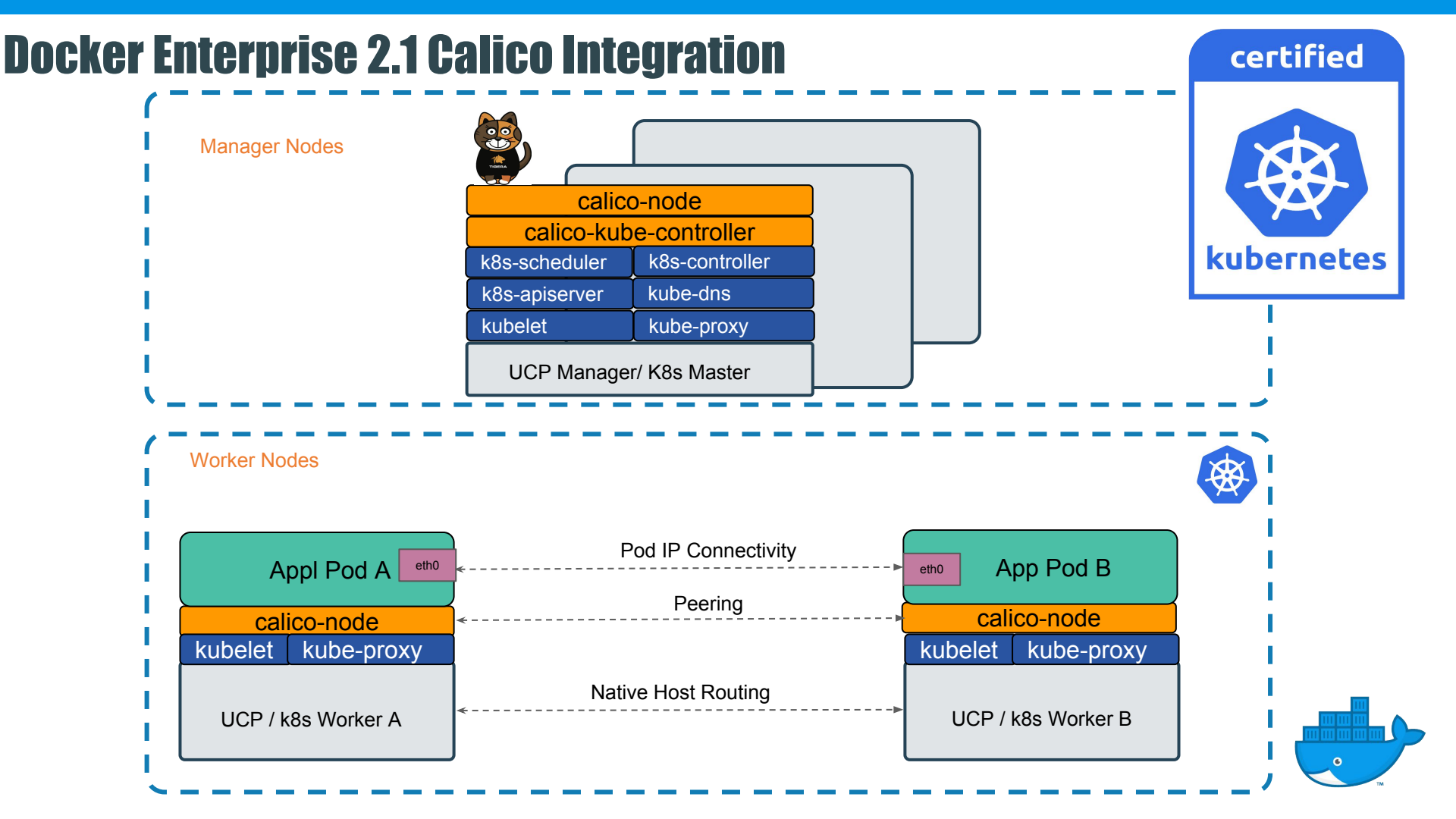
Micro-segmentation



Compliance

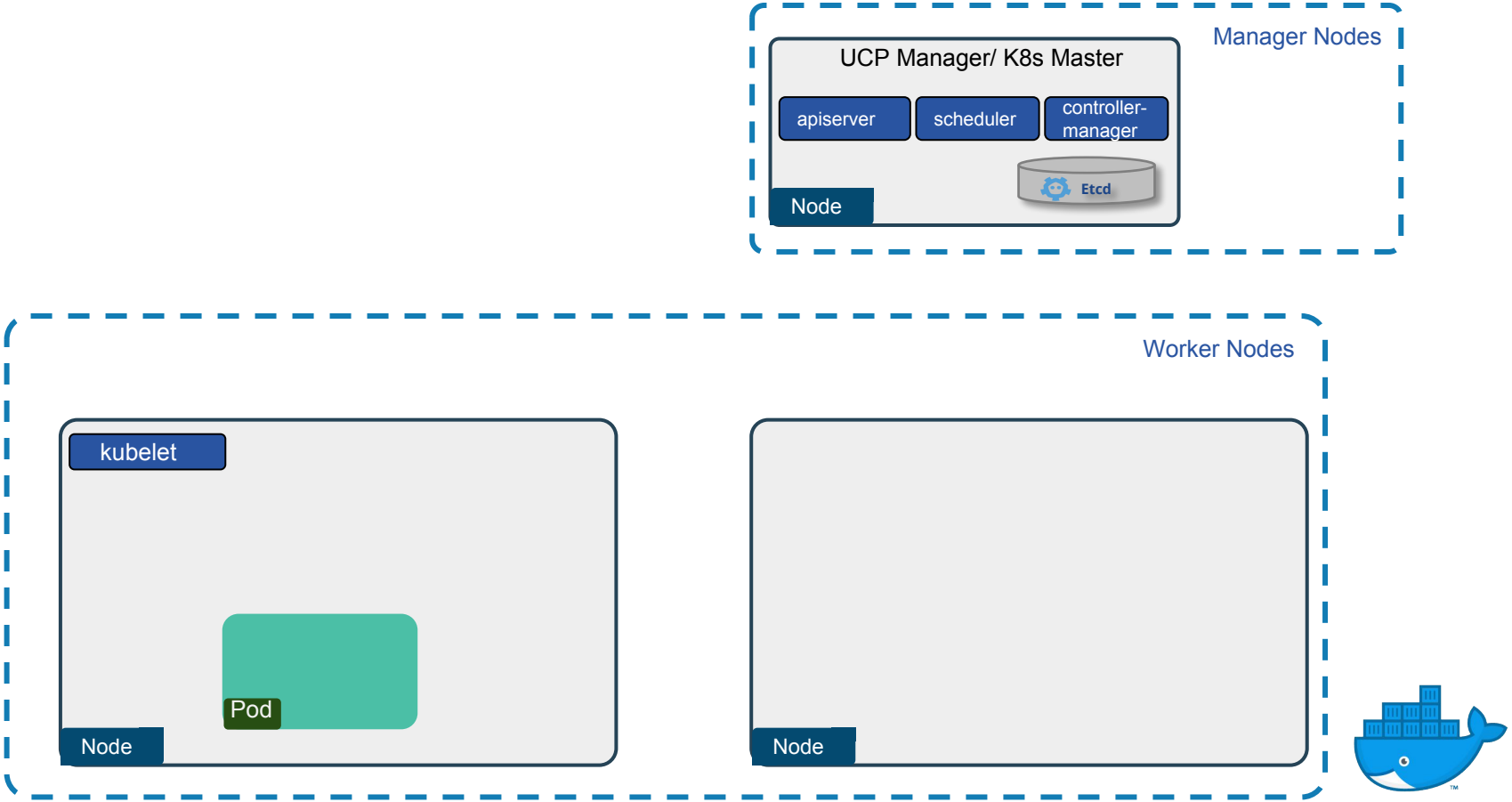
- Declarative policy-driven isolation
- Fine-grained access control
- Dynamic, in lock step with Kubernetes

Docker Enterprise 2.1 Calico Integration

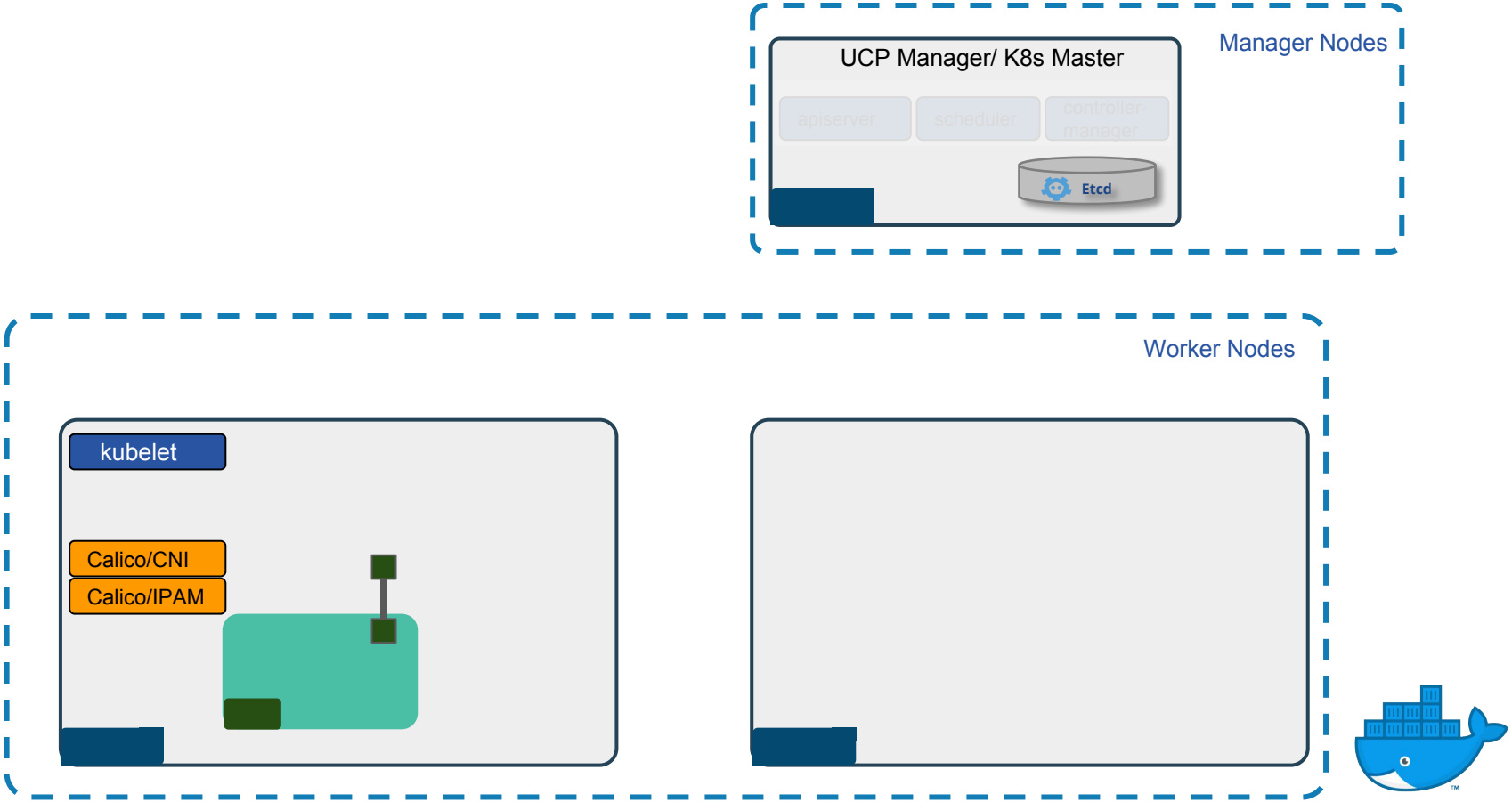


Let's dig inside each component

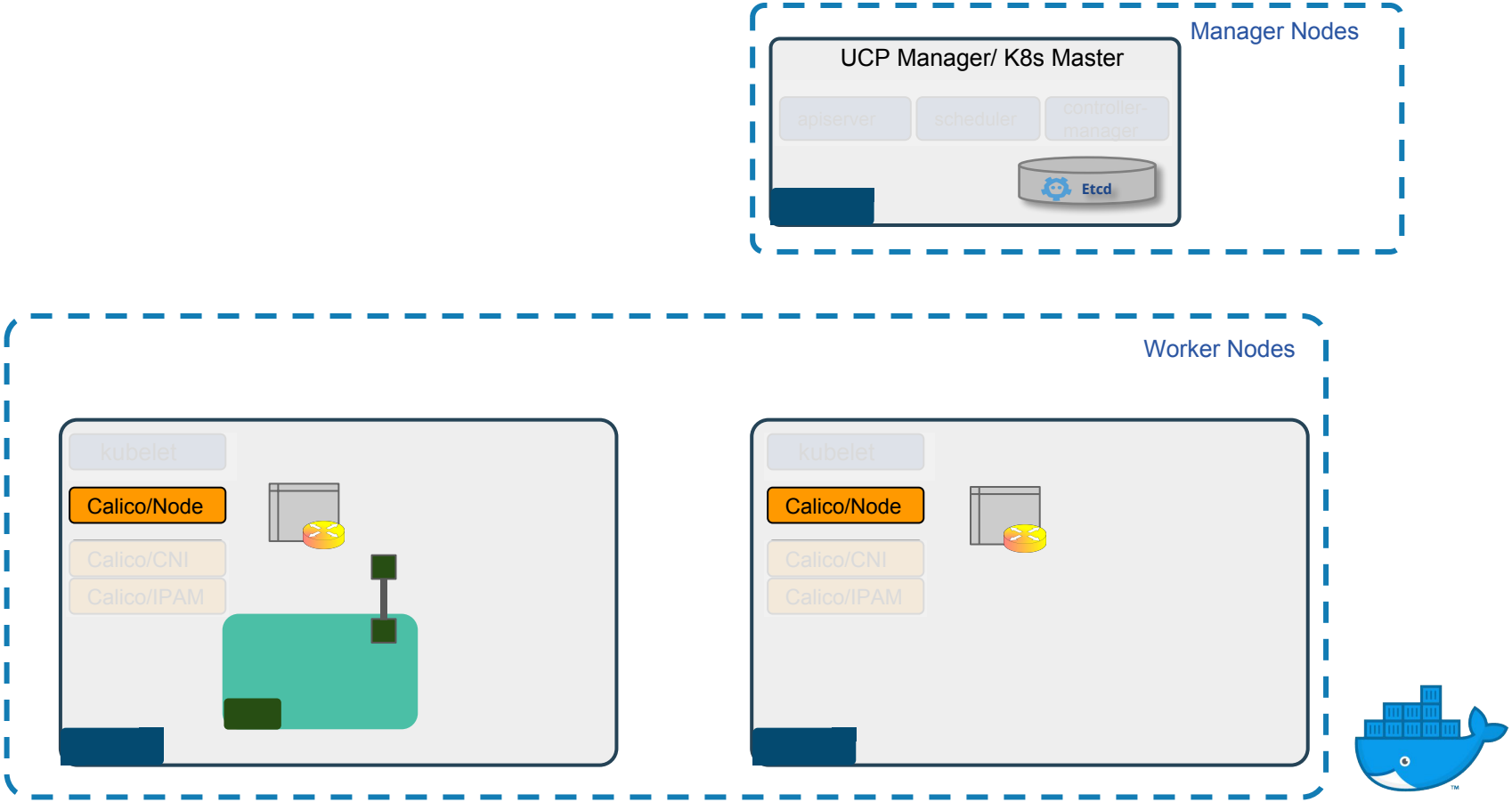
Pod Creation: Kubelet



Pod Network: Calico/CNI



Pod Network: Calico/Node

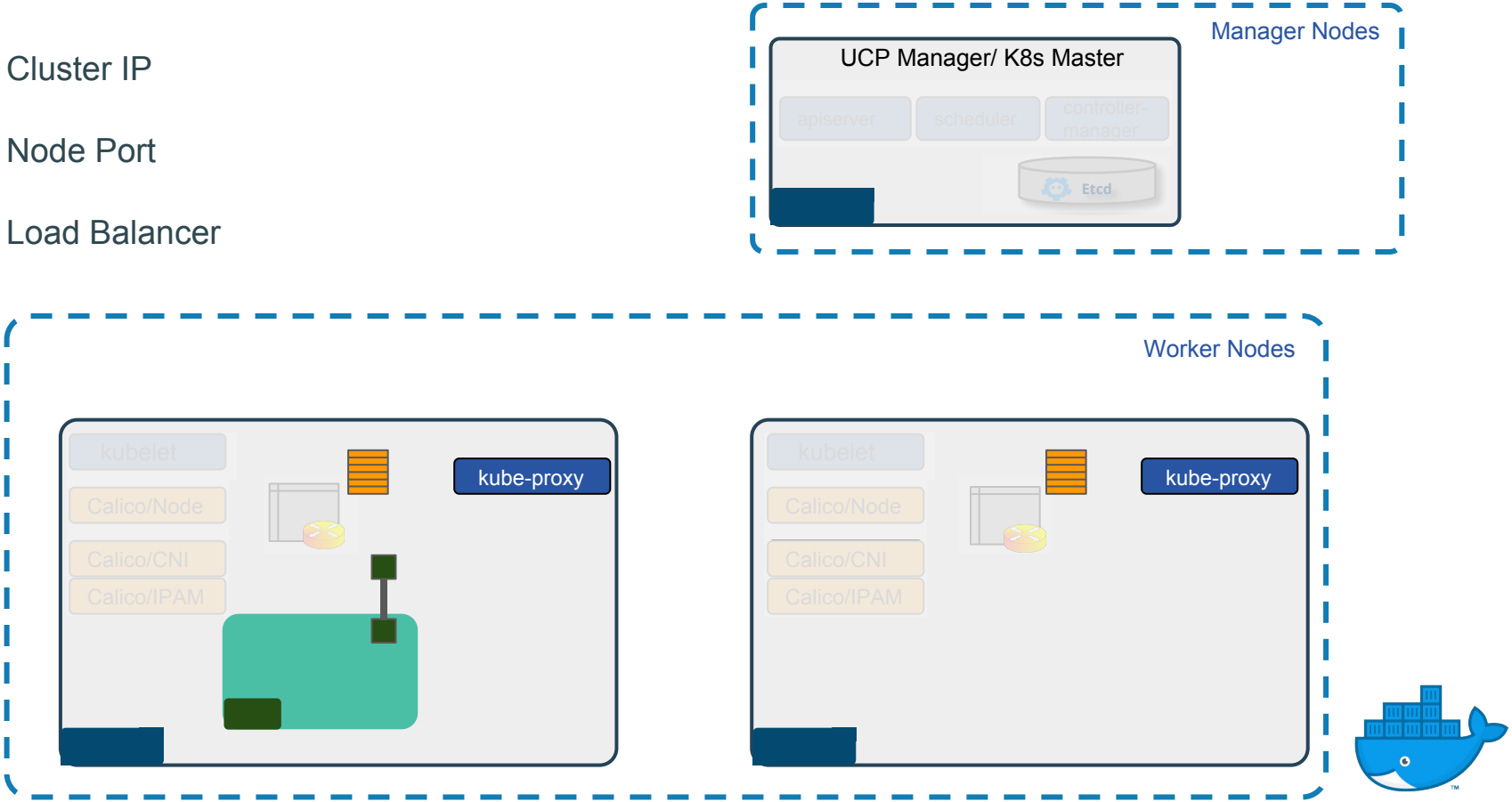


Kubernetes Services: Kube-proxy

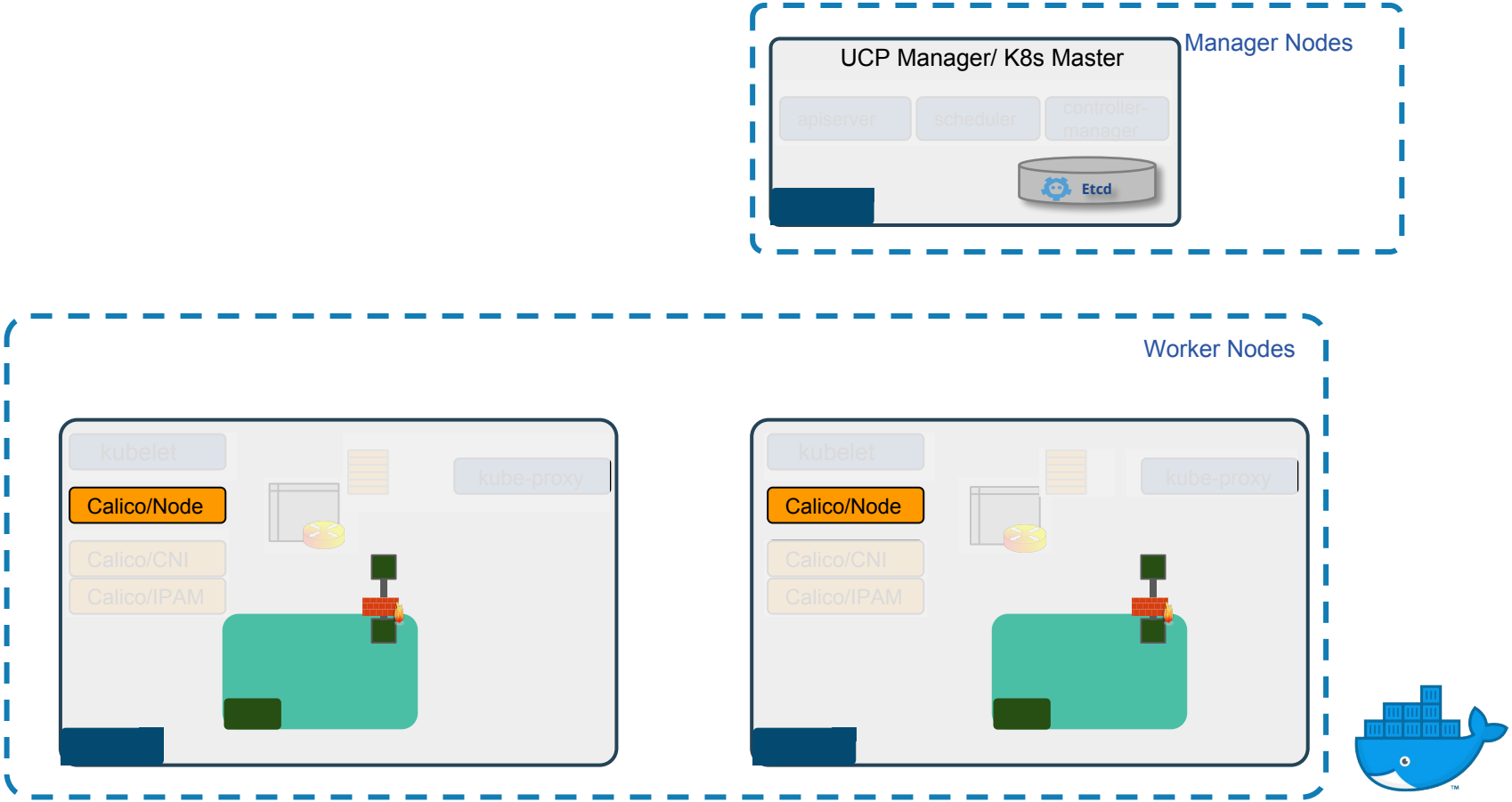
> Cluster IP

> Node Port

> Load Balancer

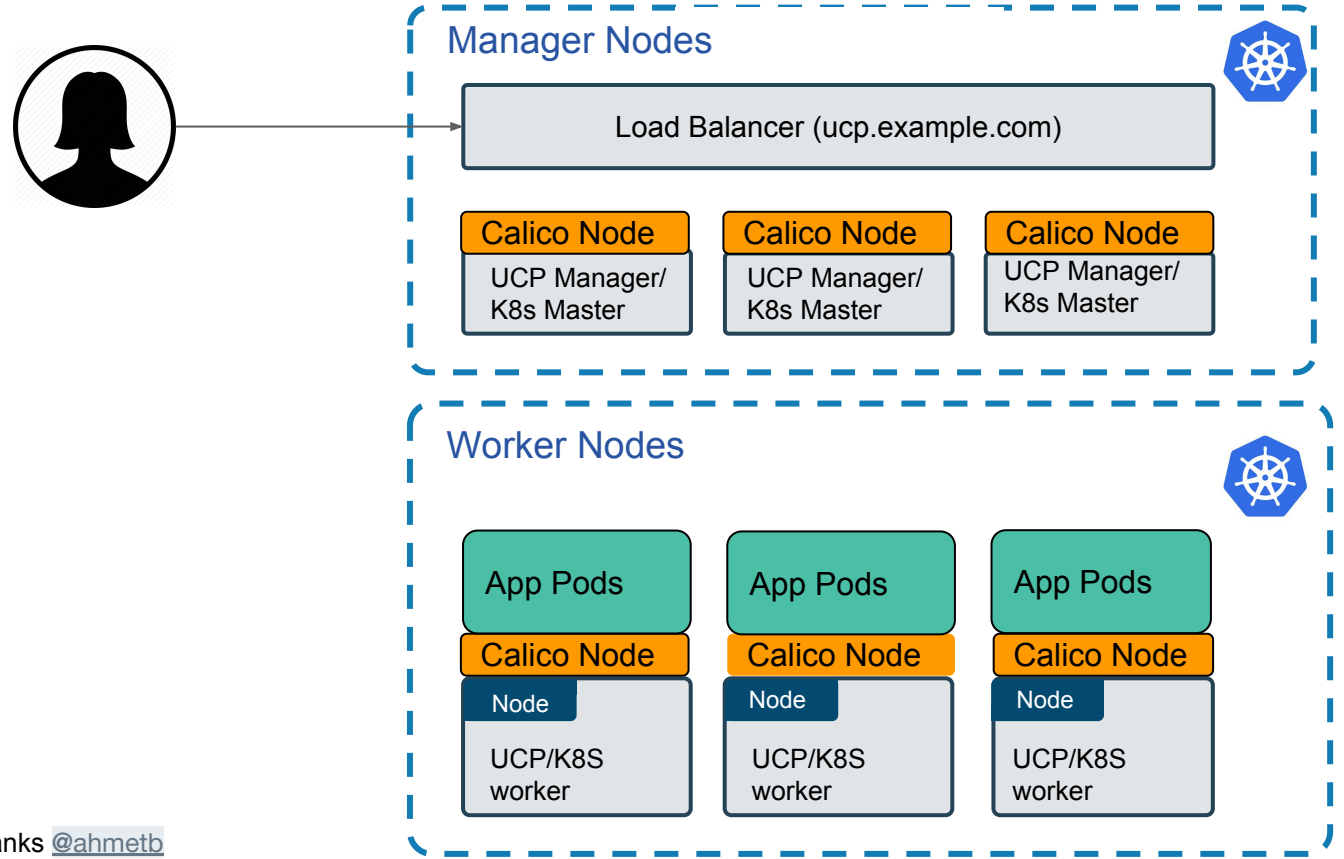


Network Policy



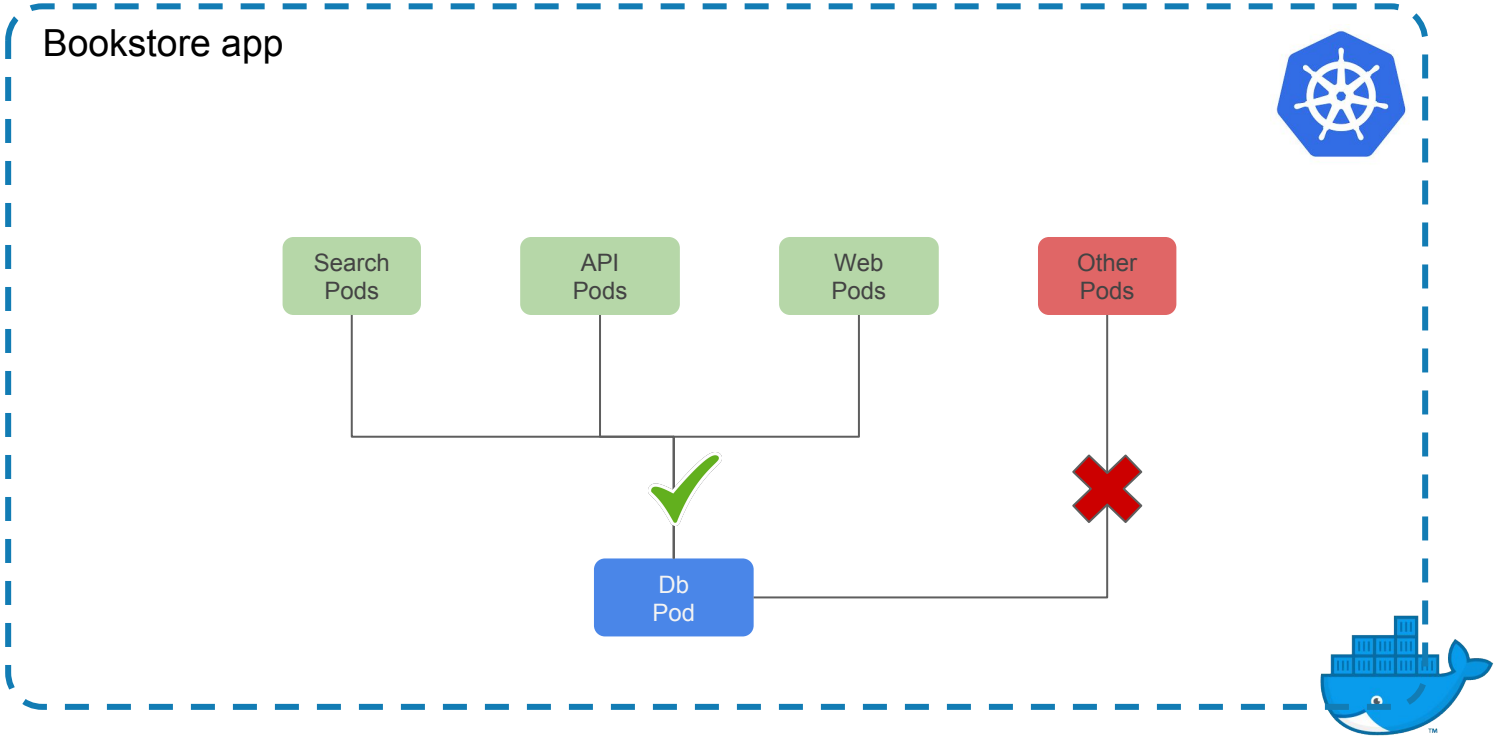
Let's demo

Demo 1: Docker Enterprise + Calico Overview



Thanks @ahmetb

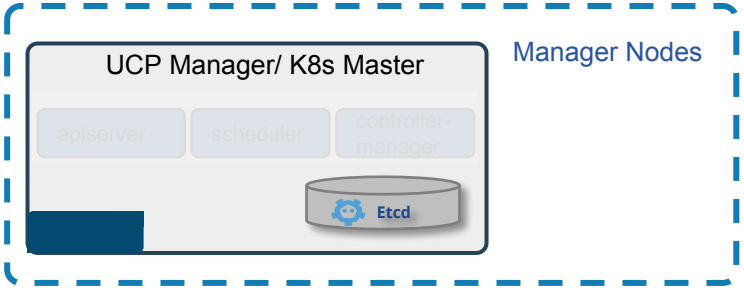
Demo 1: Network policy

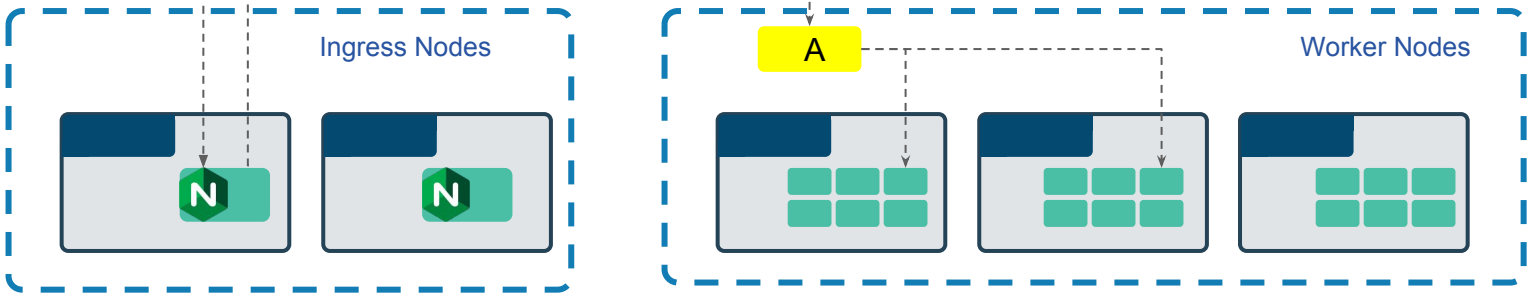


Thanks @ahmetb

Kubernetes Services: Ingress

service-a.example.com





svc

pod

Kubernetes Services: Ingress

Components

- Ingress Controller
- Ingress Resource
- Default Backend

Kubernetes Services: Ingress

Smart Loadbalancers

- Kubernetes Nginx *
- Nginx/Nginx Plus
- Traëfik

Framework Specific

- Docker Enterprise Interlock *
- Openshift Router

Loadbalancers Controllers

- **Cluster External Ingress Elements Interaction:**
  - Big IP F5 Ingress
  - Octavia Ingress Controller
  - GLBC (GCE L7 lbctl)
  - Netscaler
  - AVI Networks Vantage
- **Cluster Internal API Gateways:**
  - Kong

* Tested on Docker Enterprise

Kubernetes Services: Ingress

Ingress Resource

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: colors-ingress
  annotations:
    nginx.com/sticky-cookie-services: "serviceName=red-svc red_svc_id expires=60s path=/;"
    nginx.com/health-checks: "true"
    nginx.org/rewrites: "serviceName=red-svc rewrite=/;"
spec:
  rules:
  - host: blue.example.com
    http:
      paths:
      - path: /red/
        backend:
          serviceName: red-svc
          servicePort: 80
      - backend:
          serviceName: blue-svc
          servicePort: 80
  - host: red.example.com
    http:
      paths:
      - backend:
          serviceName: red-svc
          servicePort: 80
```

Swarm Services: Interlock

a.example.com

BYOLB

Manager Nodes

UCP Manager/ K8s Master

apiserver

scheduler

controller-manager

interlock

Etcd

Worker Nodes with Interlock Proxy

interlock-proxy

interlock-proxy

A

Worker Nodes

interlock-extension

service

tasks

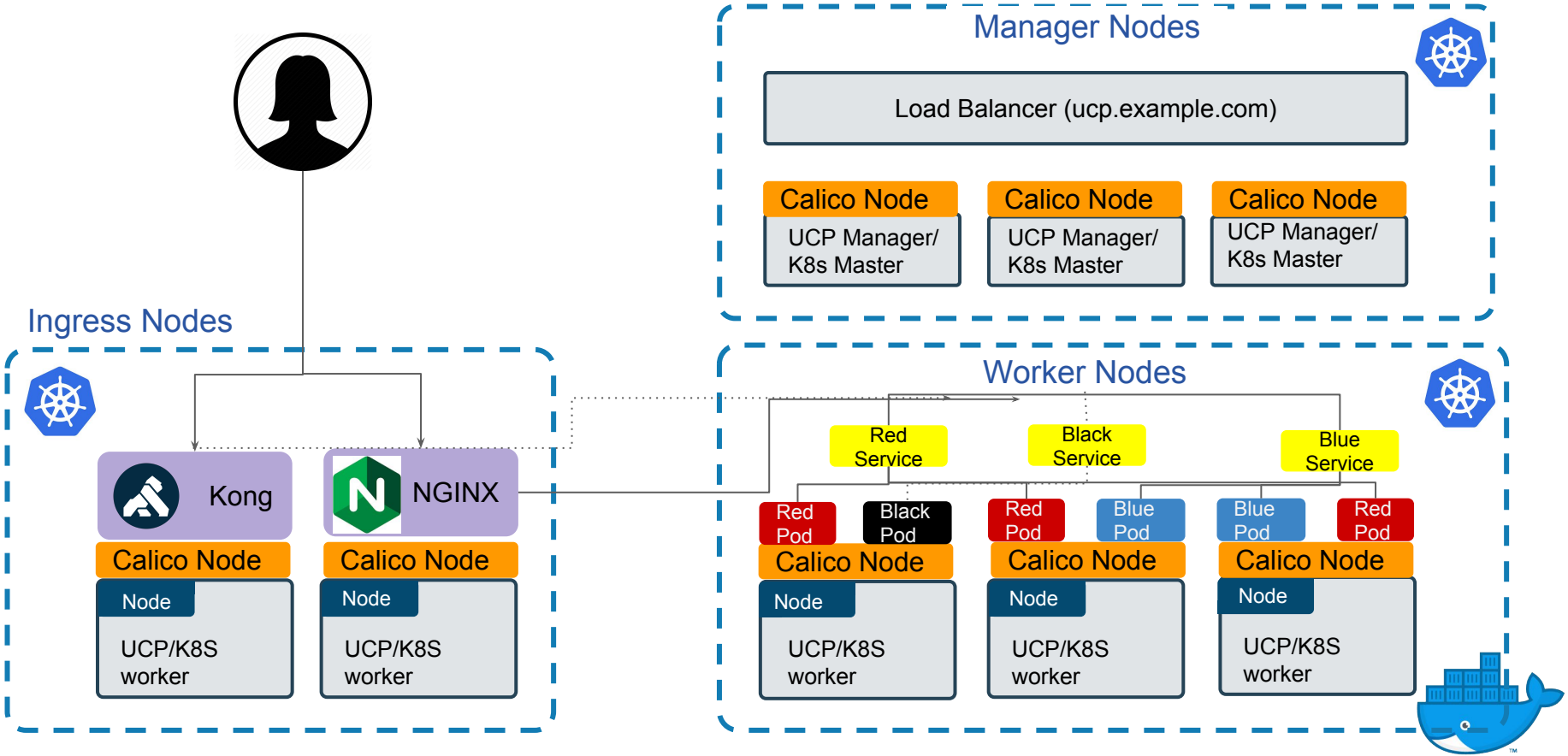
Swarm Services: Interlock

Interlock Service Publishing

```
version: "3.3"
services:
  red:
    image: codegazers/colors:1.12
    environment:
      - COLOR=red
    deploy:
     replicas: 3
     labels:
        com.docker.lb.hosts: red.example.com
        com.docker.lb.network: colors
        com.docker.lb.port: 3000
    networks:
      - colors
  blue:
    image: codegazers/colors:1.12
    environment:
      - COLOR=blue
    deploy:
     replicas: 2
     labels:
        com.docker.lb.hosts: blue.example.com
        com.docker.lb.network: colors
        com.docker.lb.port: 3000
    networks:
      - colors
networks:
  colors:
    driver: overlay
```

Let's demo

Demo 2: Let's Deploy a Sample Application



Takeaway

Docker Enterprise allows you to:

- **Easily publish your applications using Docker Swarm or Kubernetes**
- **Integrate with various solutions to support your choices of using any infrastructure**
- **Enforce a dynamic policy-based microsegmentation**

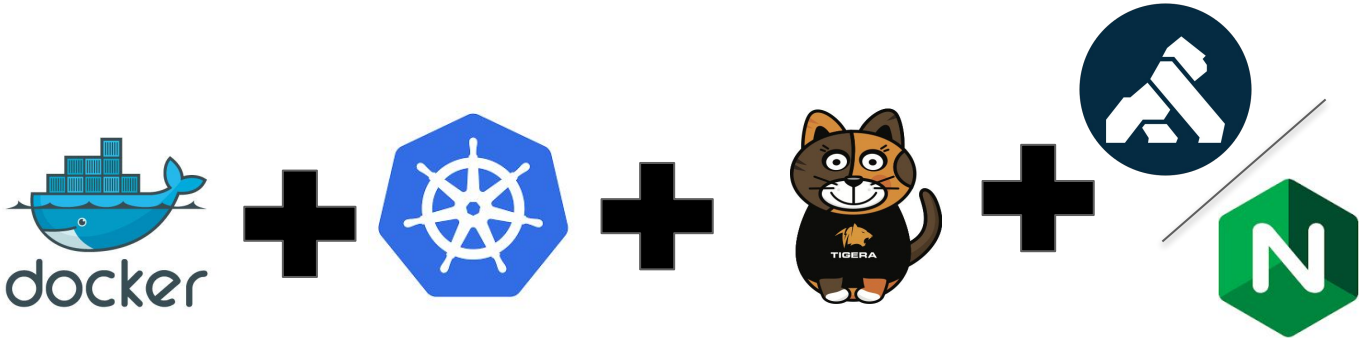
Thank you! Questions?

Try Docker Enterprise + Calico: trial.docker.com

community.docker.com

slack.projectcalico.org

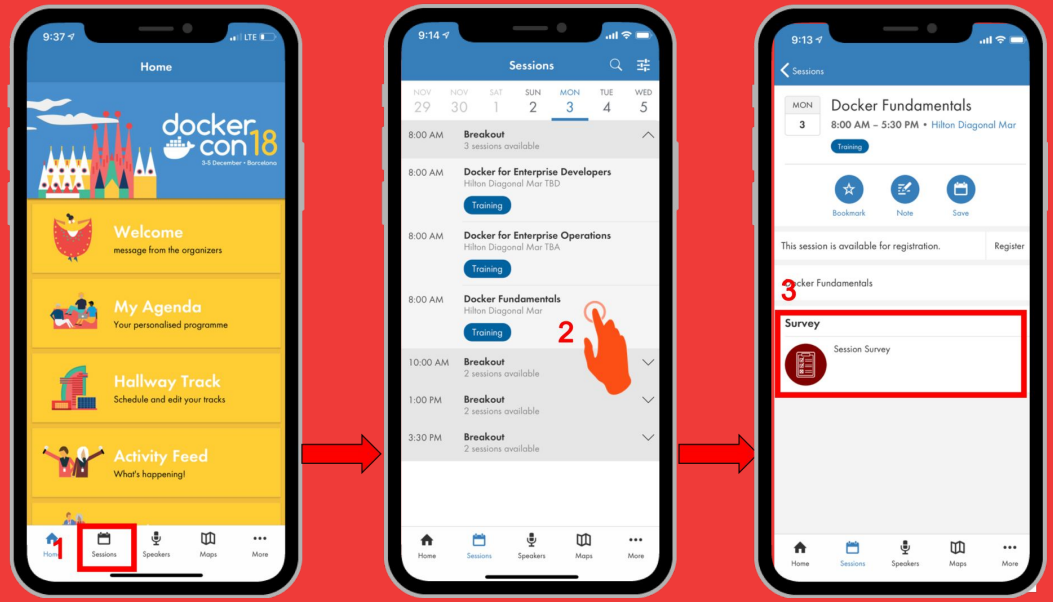
@docker @projectcalico @tigeraio @thekonginc @nginx



Take A Breakout Survey

Access your session and/or workshop surveys for the conference at any time by tapping the Sessions link on the navigation menu or block on the home screen.

Find the session/workshop you attended and tap on it to view the session details. On this page, you will find a link to the survey.



Come Join Us In San Francisco

dockercon19

April 29-May 2

2019