Overview of Container Management

Wyn Van Devanter | @wynv
Vic Kumar

excella.com | @excellaco

Agenda

- Why Container Management?
- What is Container Management?
  - Clusters, Cloud Architecture & Containers
  - Container Orchestration
  - Tool Overview
  - Container Management Systems and Beyond
- Kubernetes Basics
- Hands-On

Why container management?

Building and managing software using **containers** is becoming the **standard**

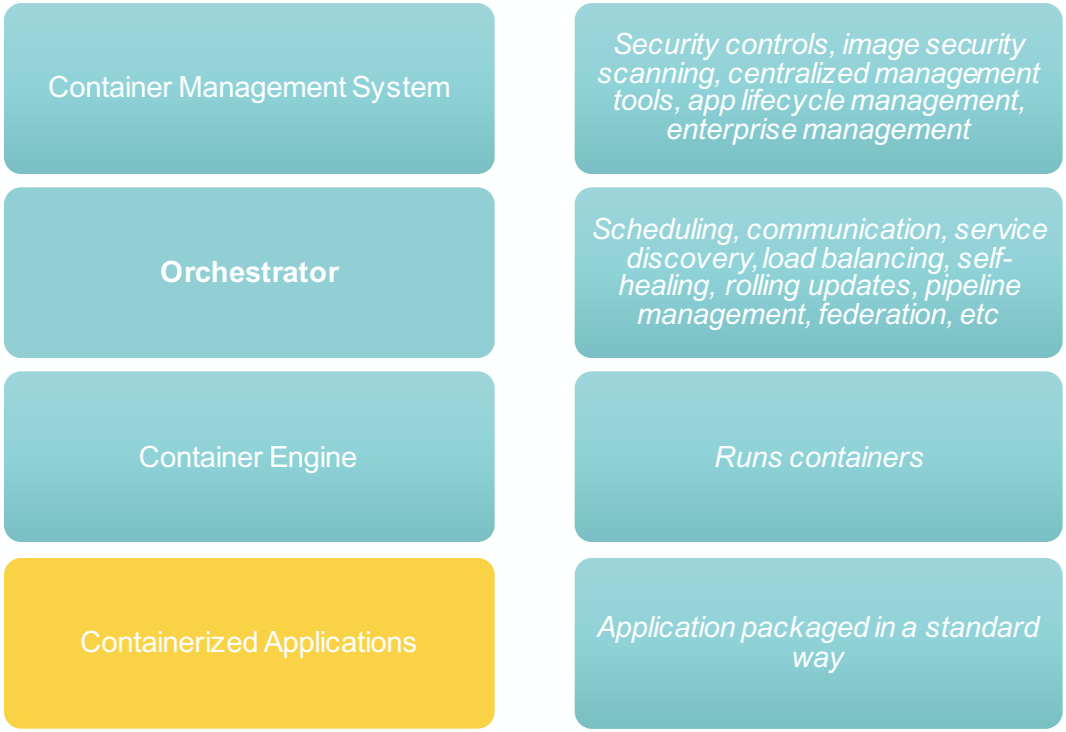
Creates new ways to package, deploy and manage software

Development to Production Parity

Centralizes software management

Fosters standardized deployment platform

Container-based Infrastructure



Enterprise: Docker EE, OpenShift

Cluster management with Kubernetes, Swarm, Mesos, Fleet, Managed: ECS, EKS

Server: Docker on Azure/AWS/VM

Local devleopment: App with React UI container, ASP.NET Core API container

What are containers?

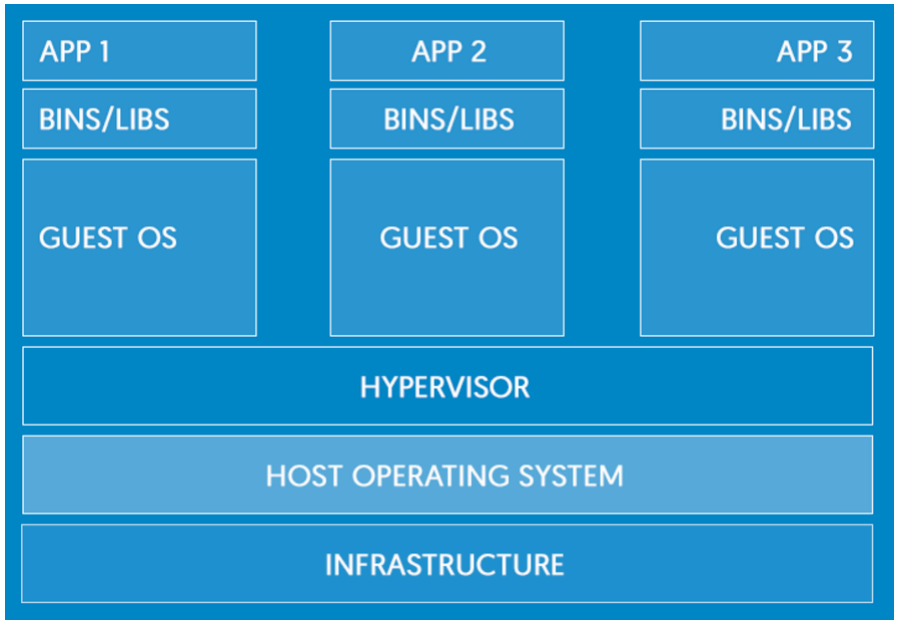
Isolate an **application**, its **dependencies** and **resource use** into a standard unit of execution & deployment

Your application and everything it needs to run, with isolation benefits *without the OS overhead*

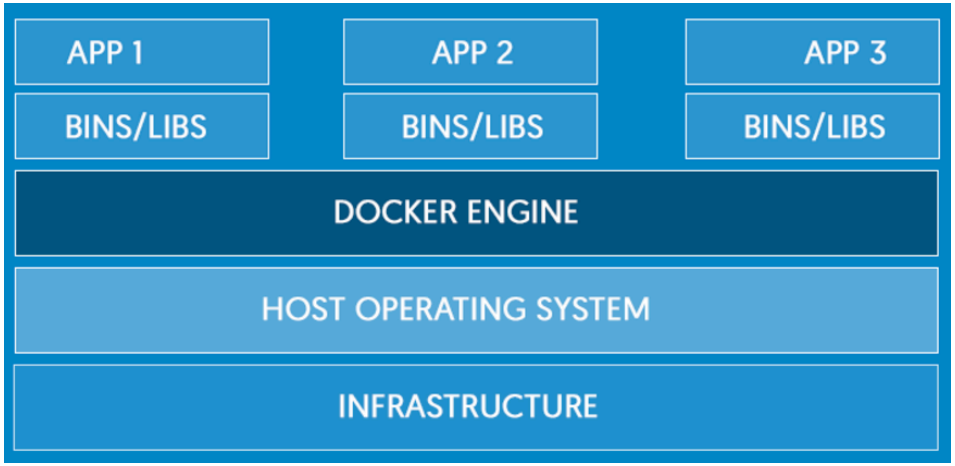
More portable, less resource use, more shippable…

What are containers?

Virtual Machines



Containers



Why Containers?

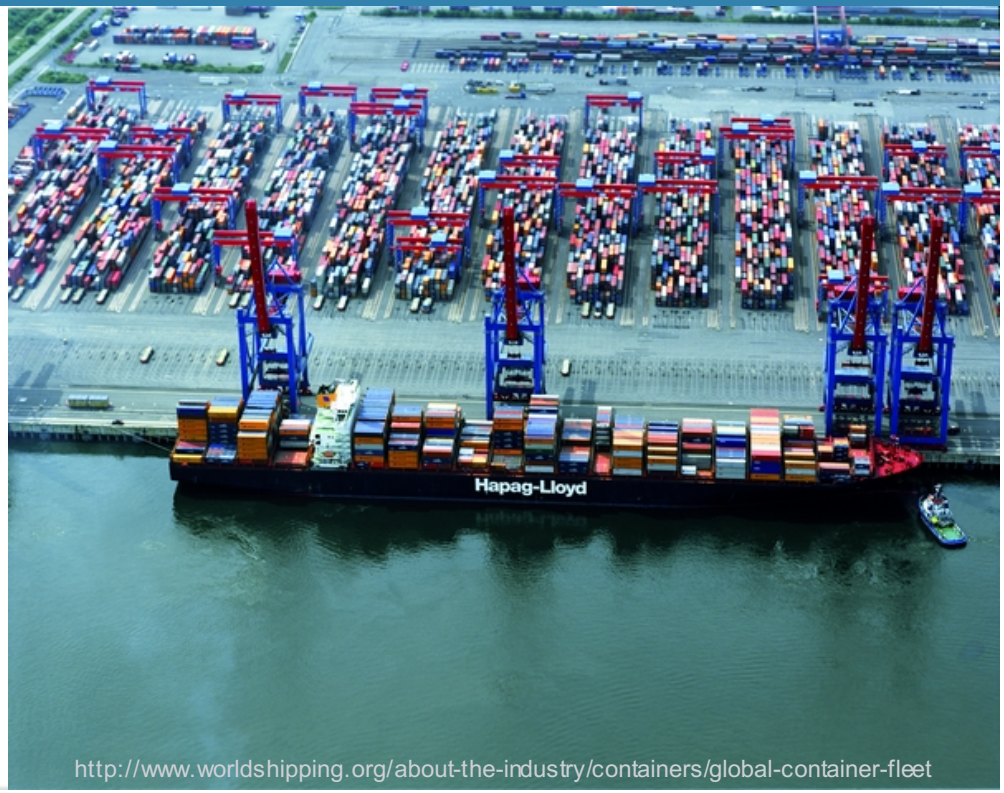
Simplifies packaging applications and its dependencies

Consistent experience

Host isolation

Deploying Containers

- Must manage many pieces (containers)
- Must be able to find each other
- Distributing workload
- Auto-scaling
- Rolling updates, rollbacks, canary, green/blue releases
- Self-service infrastructure
- Cloud agnosticism
- Multiple OS management



http://www.worldshipping.org/about-the-industry/containers/global-container-fleet

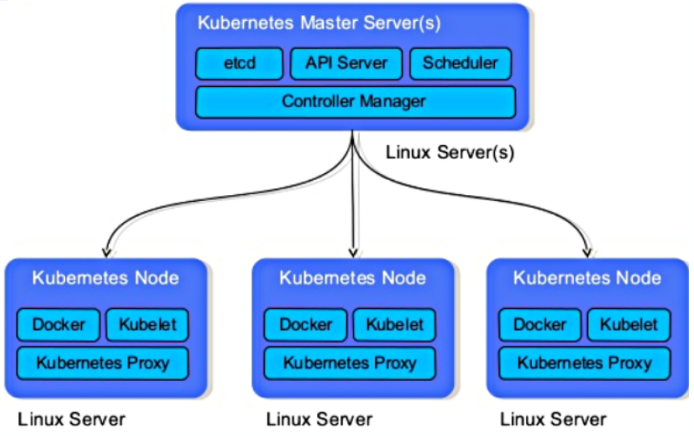
Clusters

Backbone of container infrastructure. Typically for large scale, now container-based software deployment too. A collection of servers called nodes, with masters.

Makes managing a **pool of servers** & their resources as simple as managing a **single system**

You **don't care where containers run**

With multiple container-based apps, a cluster is a way to **standardized deployment**



https://www.cbronline.com/

What is Container Orchestration?

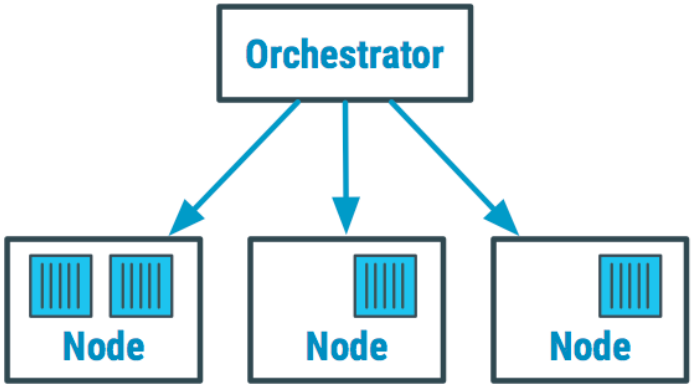
An abstraction that **simplifies** tasks of **building, deploying**, and **maintaining containers across servers**

Automates the distribution of applications across a cluster of machines, ensuring higher levels of utilization

Single platform for application deployment across servers & clouds

Decouples development teams **from machines** they're using

Operationalized efficiency across the organization

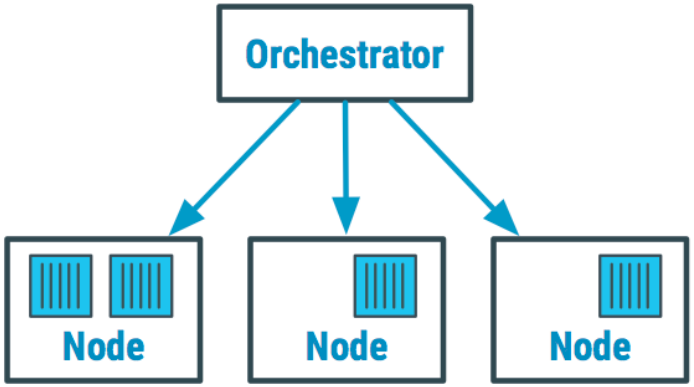


https://blog.docker.com/2017/10/least-privilege-container-orchestration/

What is Container Orchestration?

Orchestrator functionality includes:

- Provisioning and managing cluster
- Scheduling
- Cross-node container communication
- Service discovery
- Load balancing
- Scaling
- Self-healing (automatic restarts)
- Rolling updates & rollbacks
- Blue/green & canary deployments
- Storage management



https://blog.docker.com/2017/10/least-privilege-container-orchestration/

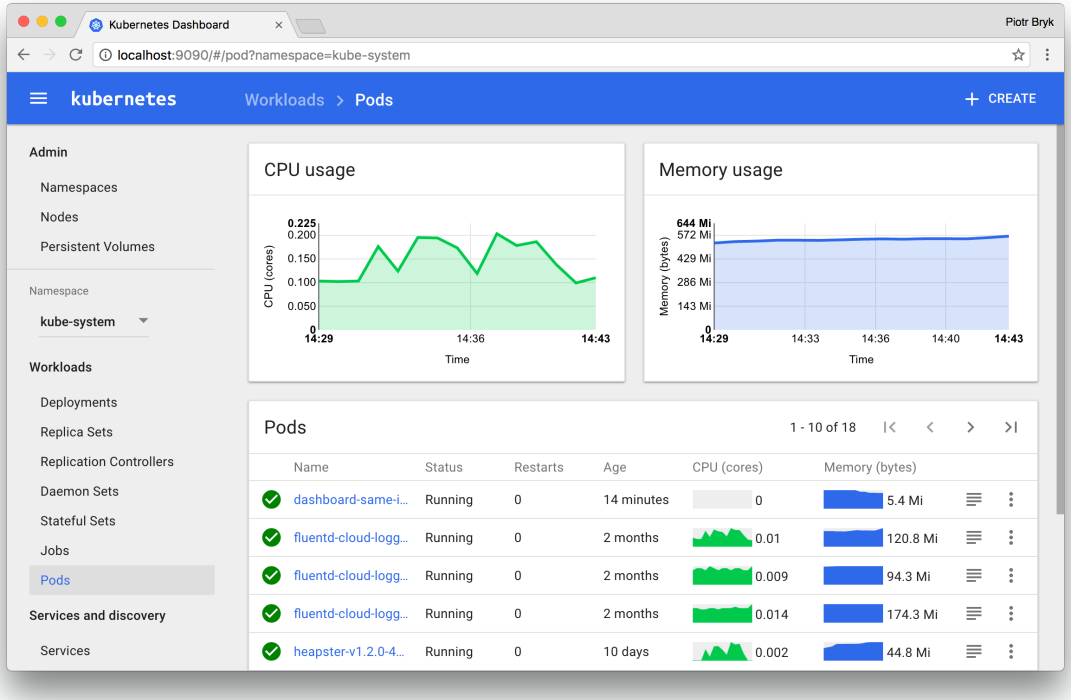
Container Orchestration

Options:

- Kubernetes
- Swarm
- Mesos/Marathon
- Nomad

Managed:

- ECS, ACS
- EKS, AKS, GKE
- Fargate



How the tools work

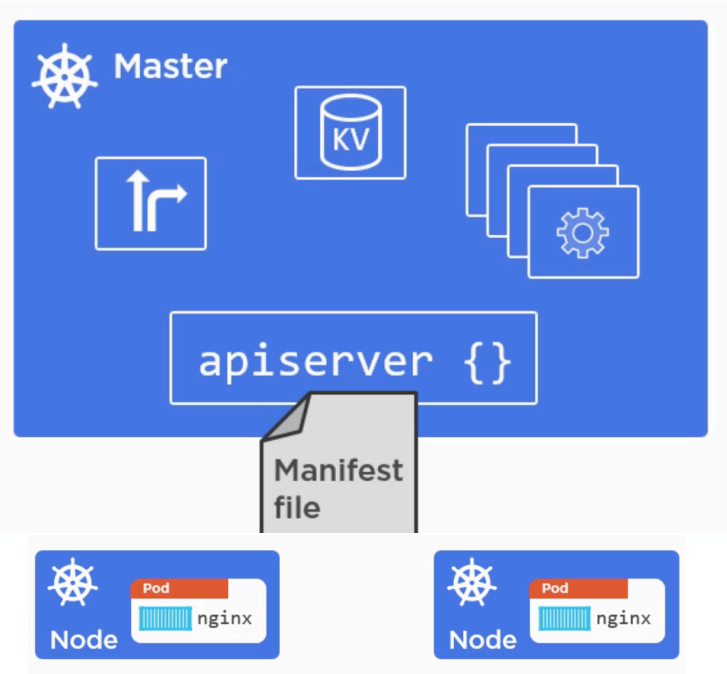
Create configuration file that specifies containers and other settings that make up a service

Declarative State

Tool places containers on nodes in cluster

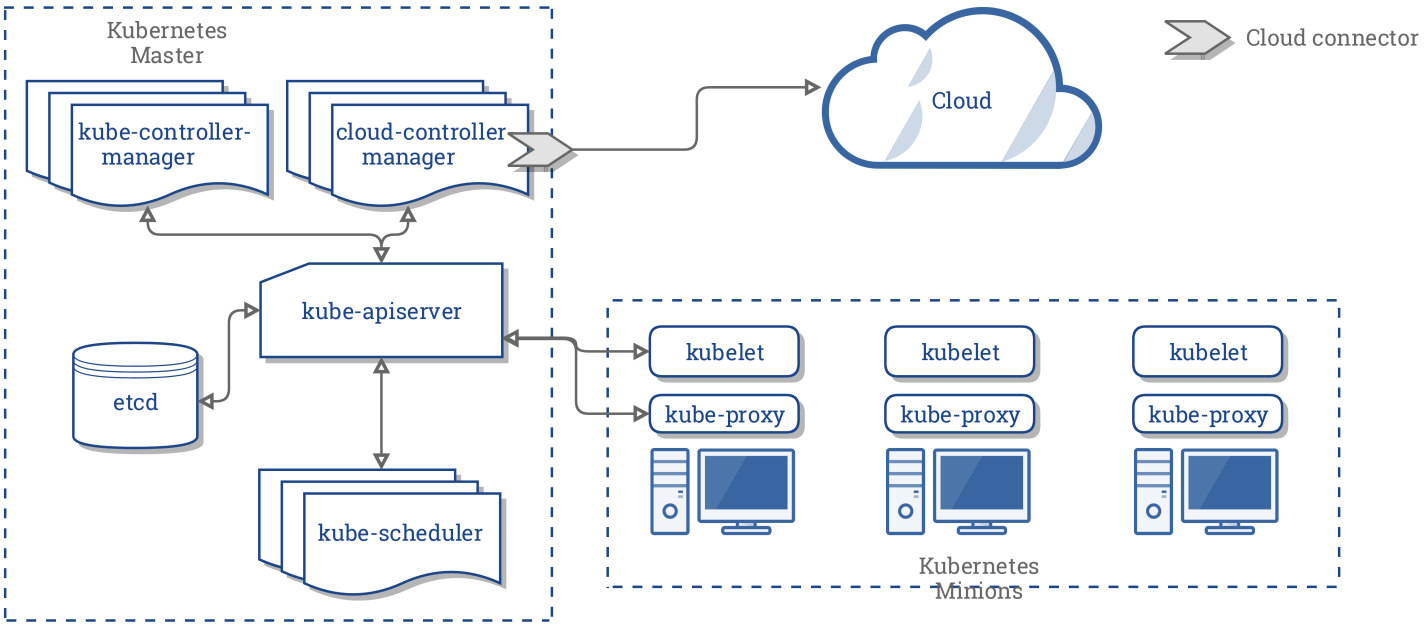
Makes sure they stay healthy

Can manage deployments from the tool



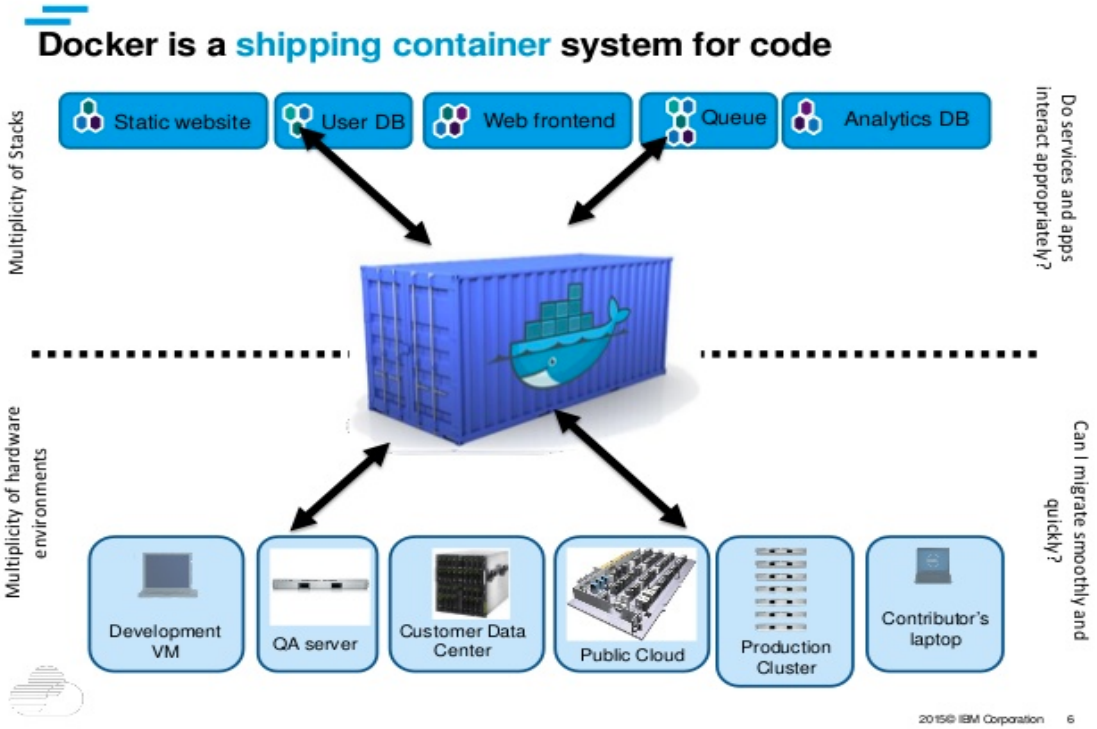
Getting Started with Kubernetes, Pluralsight, Nigel Poulton

Kubernetes Cluster



https://kubernetes.io/docs

Container Orchestration



Choosing a Container Management System

- What kinds components are going into containers?
- How do the components of the application talk to one another?
- How is availability handled?
- How is access control handled?
- What sort of scalability is needed
- How many applications am I planning to support?
- Do I need to manage my own cluster(s)?

Kubernetes

Began in 2014 by Google, inspired by their Borg and other internal container scaling projects

It has quickly risen to dominate the container orchestration space

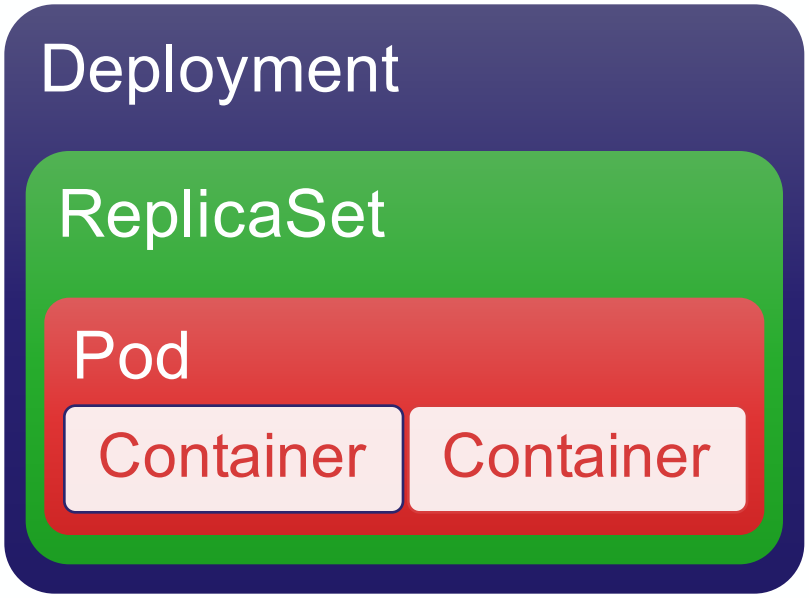
Heavyweights behind it including Red Hat, IBM

kubernetes

Kubernetes Concepts

Kubernetes uses **persistent objects** to manage its state, including

- What containerized applications are running, on which nodes
- Resources available to those applications
- Policies around how those applications behave (i.e. restart, upgrades, fault-tolerance)



Kubernetes Concepts

- Control Plane
- Master & Nodes - normally doesn't run containers, but the key Kubernetes services; nodes
- Namespaces - logical grouping of cluster for use with multiple users or projects
- Labels - K/V pair for categorizing objects such as pods
- Pods - wraps container(s)
- Deployments, ReplicaSets - manages the desired state, i.e. specify number of pods
- Services - allows external pod communication
- Volumes - share data between containers, persistent storage; beefier than Docker volumes

Manifest

Written in YAML or JSON, these files describe the **desired state** of your application in terms of Kubernetes API objects

A file can include one or more API object descriptions (i.e. service, deployment)

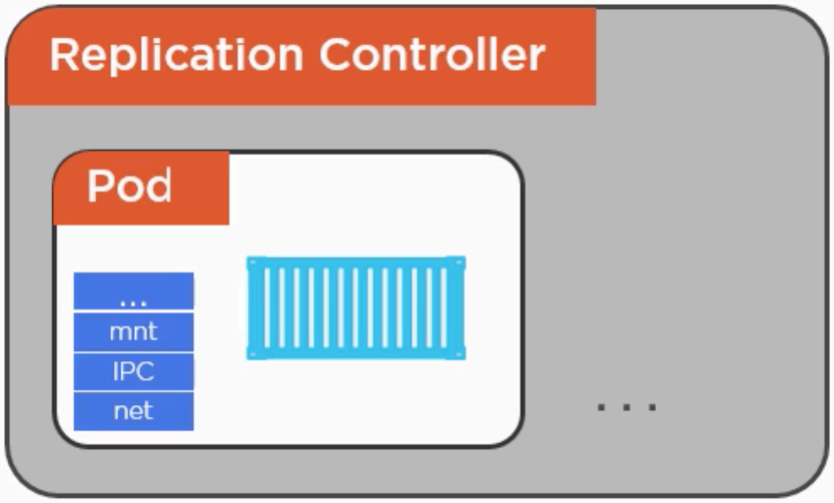
Typical way to deploy container-based services to the cluster

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

Pod

It is a **wrapper** around a group of one or more **containers** with *shared storage, network and a specification* for how to run.

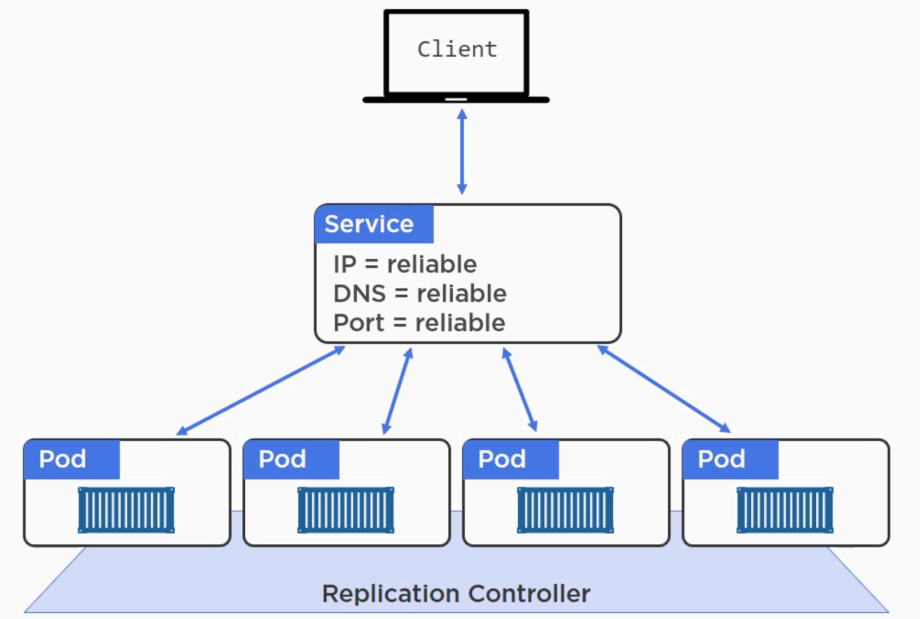
It represents **processes** that would run on the **same server** in the pre-container world. Pods do act like a single server.



Getting Started with Kubernetes, Pluralsight, Nigel Poulton

Service

- Receives and load balances requests to pods
- Depending on the type of Service used, these requests can come from external client apps or be limited to apps within the same cluster.
- A Service can be tied to a specific Deployment using label selection.
- Services have a cluster-wide IP, DNS name and port



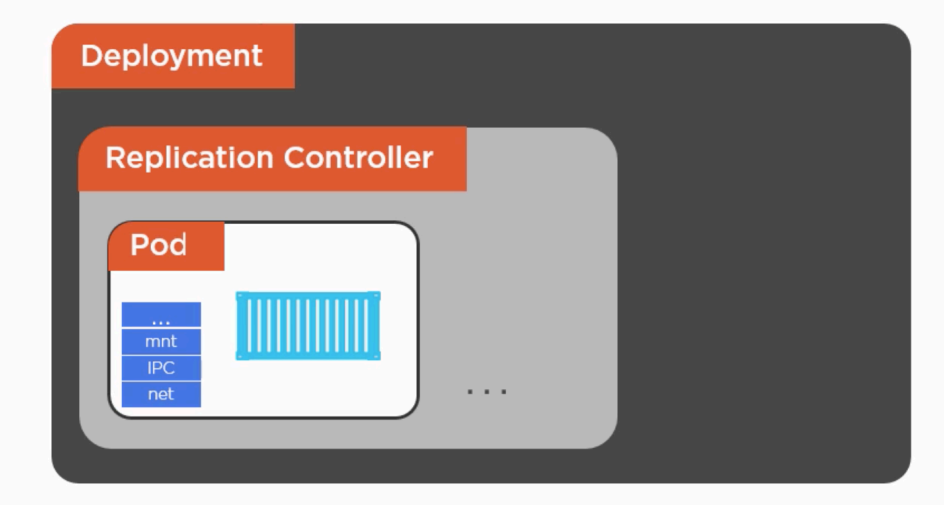
Getting Started with Kubernetes, Pluralsight, Nigel Poulton

Deployment

The most common way of running X copies (Pods) of your application thru ReplicaSets

Supports versioned rolling updates and rollbacks

Easy to have multiple concurrent versions thus blue/green and canary deployments

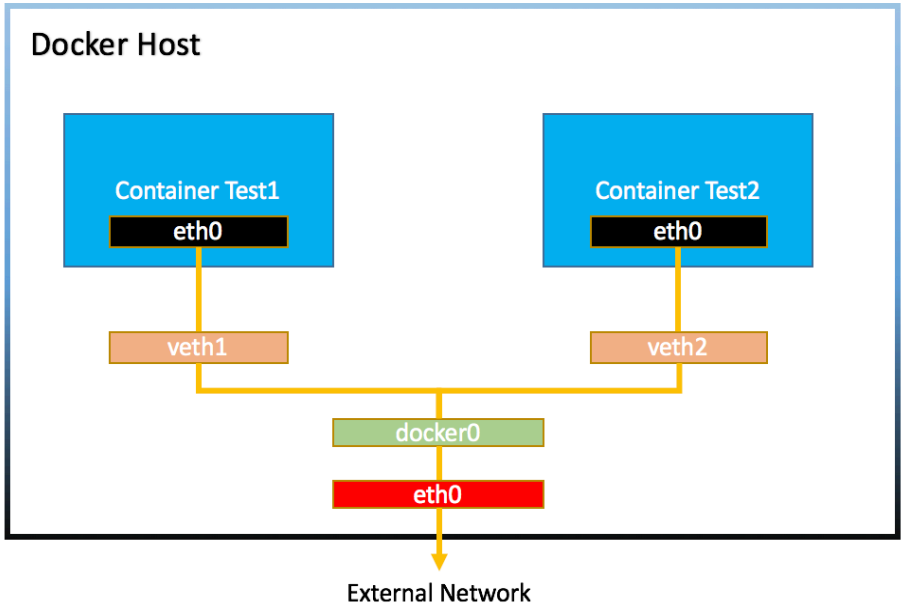


Getting Started with Kubernetes, Pluralsight, Nigel Poulton

Container Networking

There are 4 distinct networking problems to solve:

- Highly-coupled *container-to-container* communications: this is solved by pods and localhost communications.
- *Pod-to-Pod* communications
- *Pod-to-Service* communications: done using services.
- *External-to-Service* communications: done using services



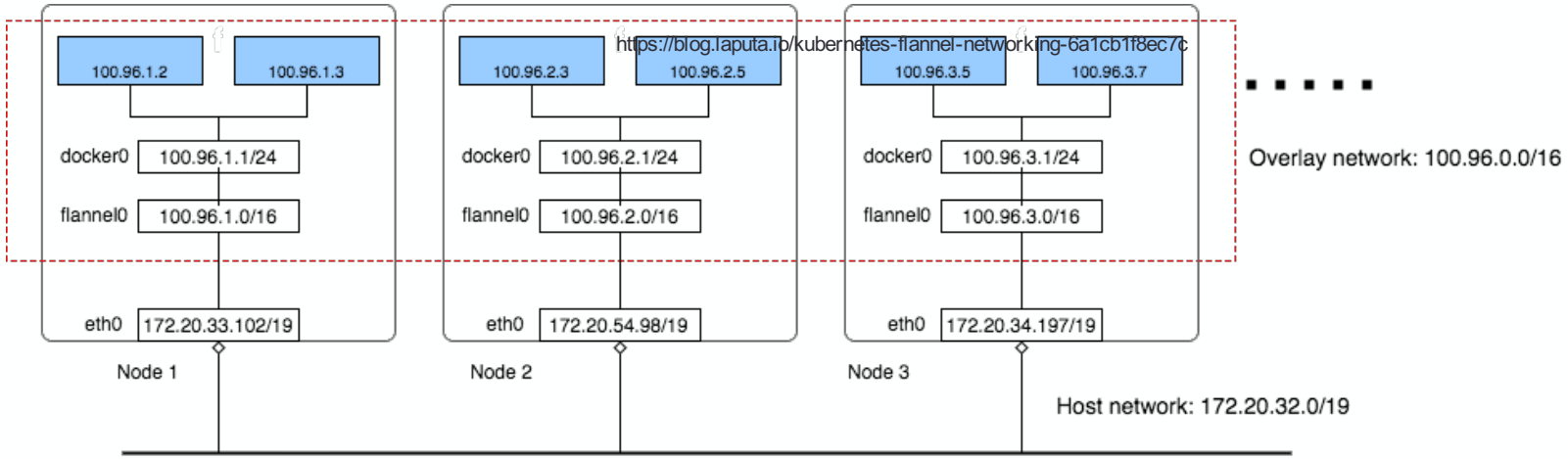
https://docker-k8s-lab.readthedocs.io/en/latest/docker/bridged-network.html

Kubernetes Networking

All containers can communicate with all other containers without NAT

All nodes can communicate with all containers (and vice-versa) without NAT

The IP that a container sees itself as is the same IP that others see it as



CNI Plug-ins

Kubernetes uses CNI as an interface between network providers and Kubernetes networking.

bridge: Creates a bridge, adds the host and the container to it
ipvlan: Adds an ipvlan interface in the container
loopback: Set the state of loopback interface to up
macvlan: Creates a new MAC address, forwards all traffic to that to the container
ptp: Creates a veth pair
vlan: Allocates a vlan device

Third-Party Plug-ins

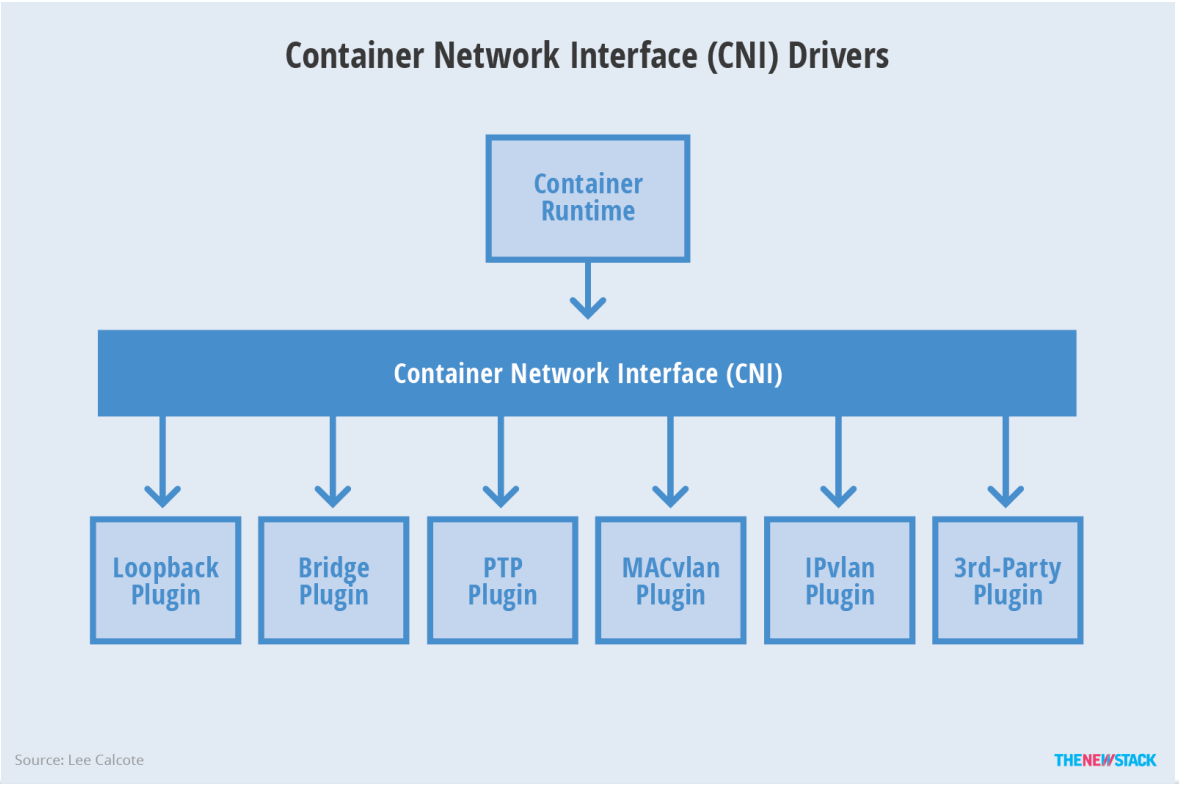
Weave

Flannel

Calico

Romana

CNI-Genie from Huawei



Namespaces

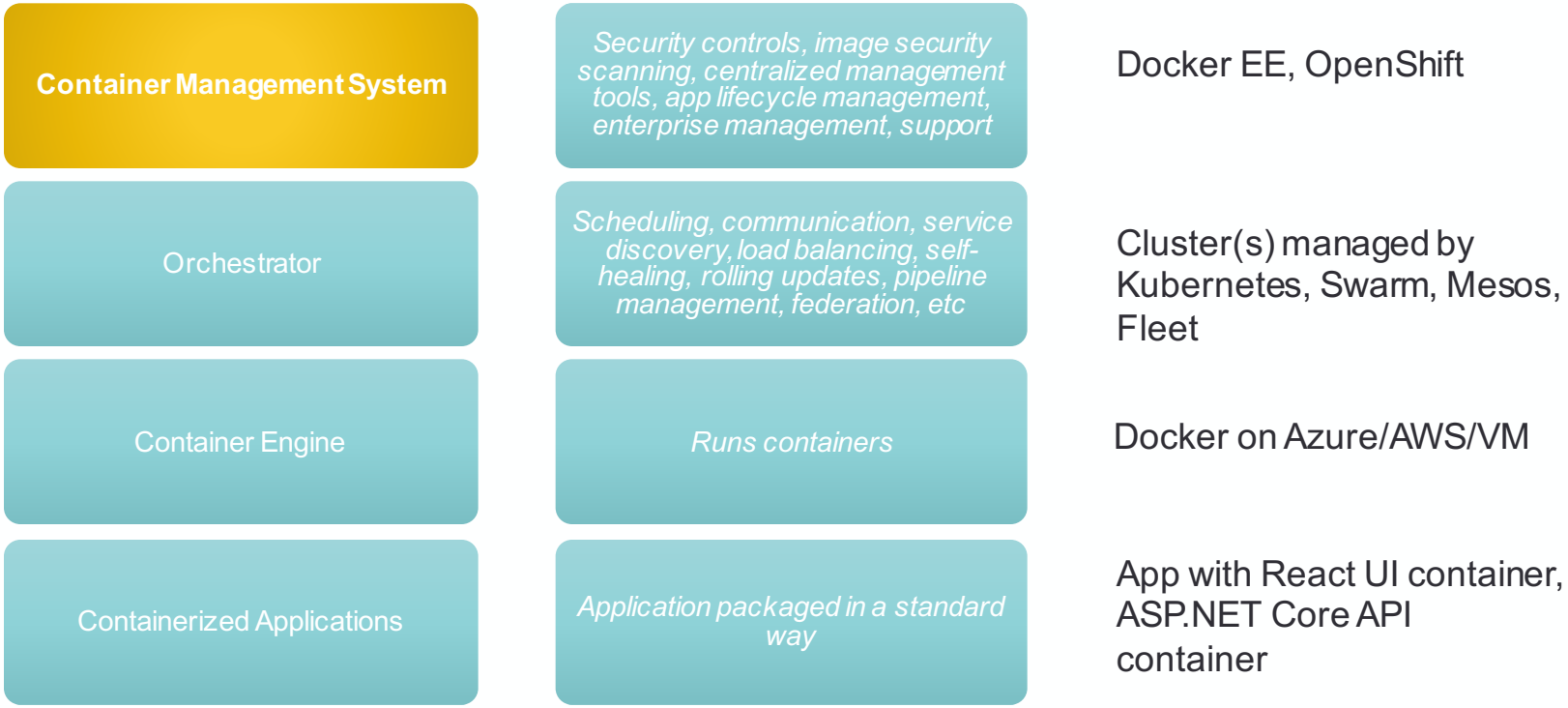
Groups pods

Can be used to provide quotas and limits around resource usage

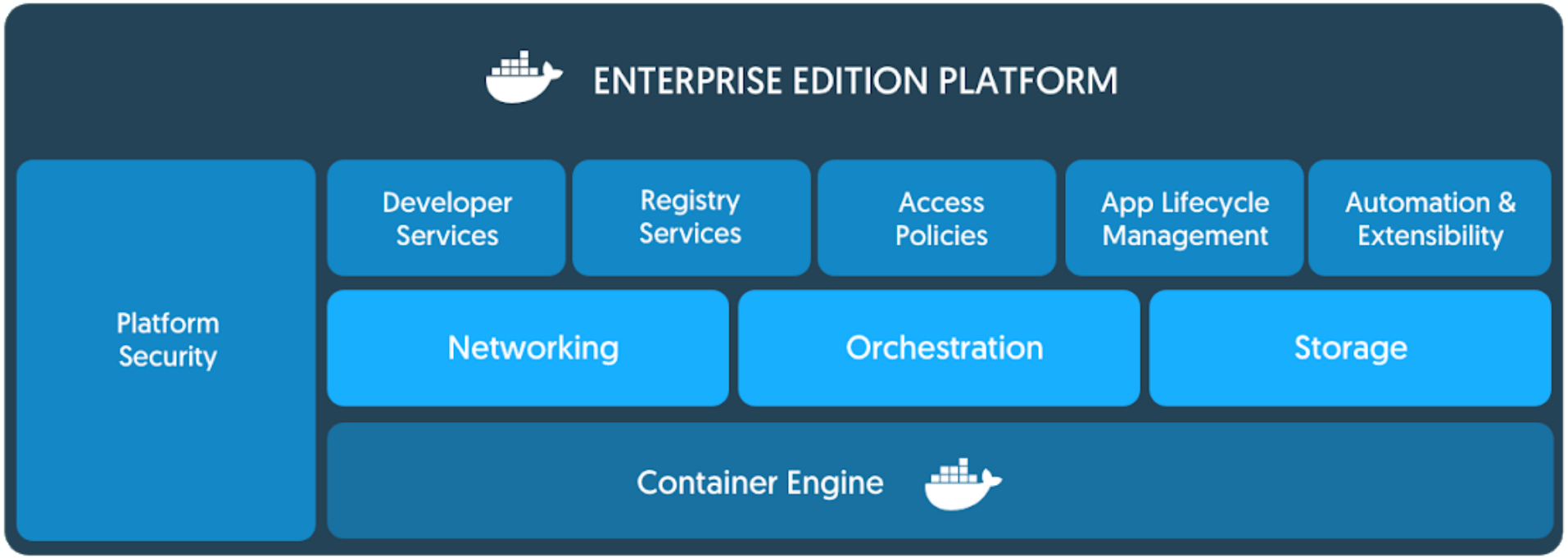
Can have impact on DNS names Kubernetes creates internal to the cluster

If no namespace is specific, “default” is used

Container Management Systems



Docker EE


ENTERPRISE EDITION PLATFORM
Platform Security
Developer Services
Registry Services
Access Policies
App Lifecycle Management
Automation & Extensibility
Networking
Orchestration
Storage
Container Engine

Hands-on

https://github.com/excellalabs/docker-workshop-2

https://goo.gl/CuUmXF

kubernetes

Thank you!

@wynv | wyn.vandevanter@excella.com

- **Getting Started with Docker, Self-Guided Workshop,** https://github.com/excellalabs/docker-workshop-1
- **Getting Started Deploying with Docker via Kubernetes, Self-Guided Workshop,** https://github.com/excellalabs/docker-workshop-2
- **Docker Swarm workshop**, https://github.com/jpetazzo/container.training
- **Deploy ASP.NET Core app to Kubernetes on Google Kubernetes Engine**, https://codelabs.developers.google.com/codelabs/cloud-kubernetes-aspnetcore

https://www.slideshare.net/wynvandevanter/container-orchestration-overview-107192685