# Introduction to Containers

NA Partner Channels – IBM Cloud
Technical Evangelism Team
January 23, 2018

© Copyright IBM Corporation 2017

# Agenda

1. Containers
2. Why Containers?
3. Docker Containers
4. Container Orchestration
5. Kubernetes
6. Conclusion

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Why Containers?

© Copyright IBM Corporation 2016

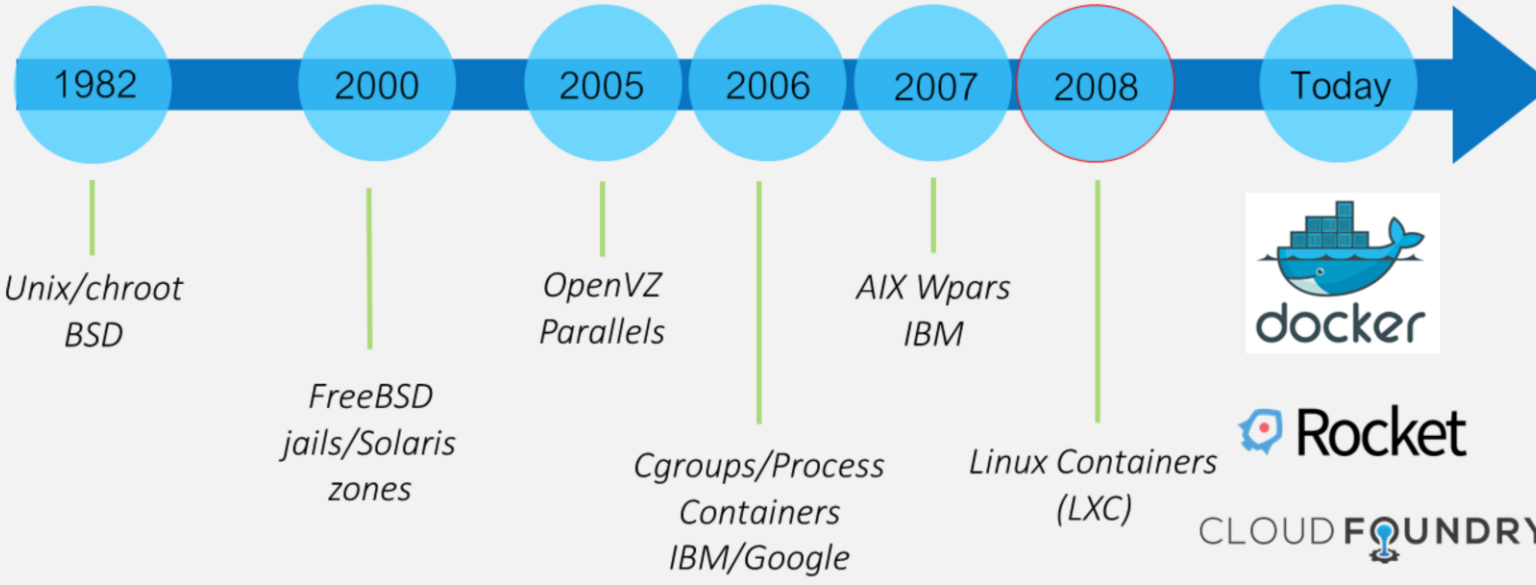
# Everyone Loves Containers


- A standard way to package an application and all its dependencies so that it can be moved between environments and run without changes
- Containers work by isolating the differences between applications inside the container so that everything outside the container can be standardized

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

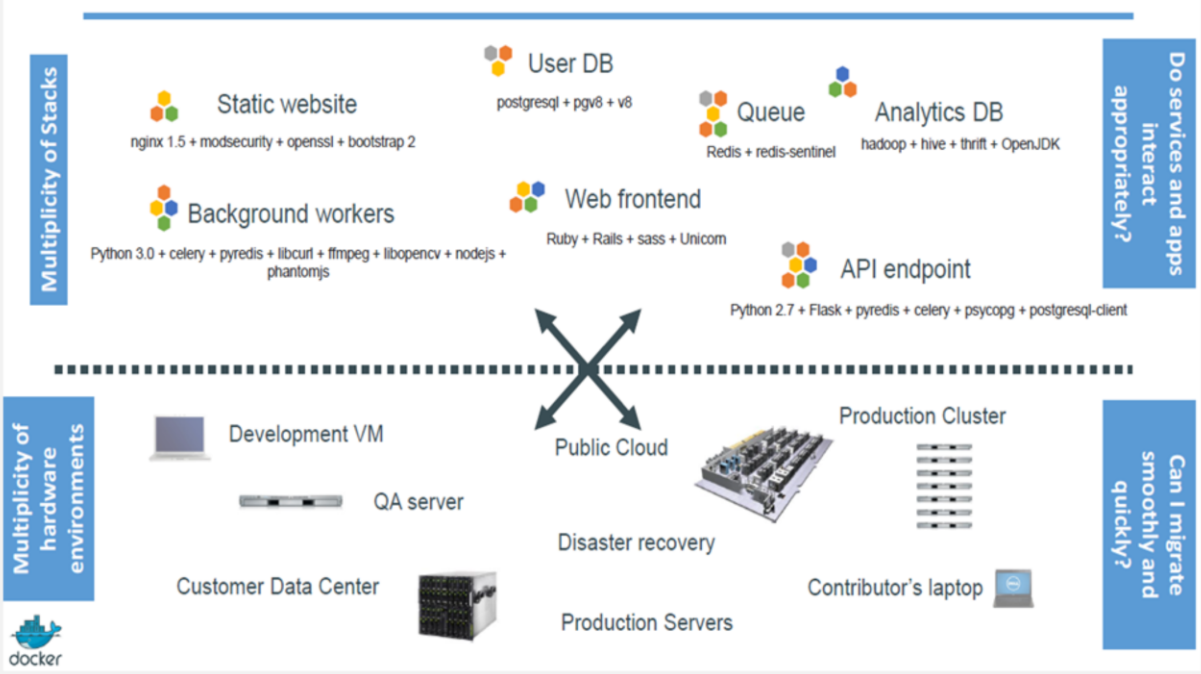
# Container History Lesson




BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# The Challenge




BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

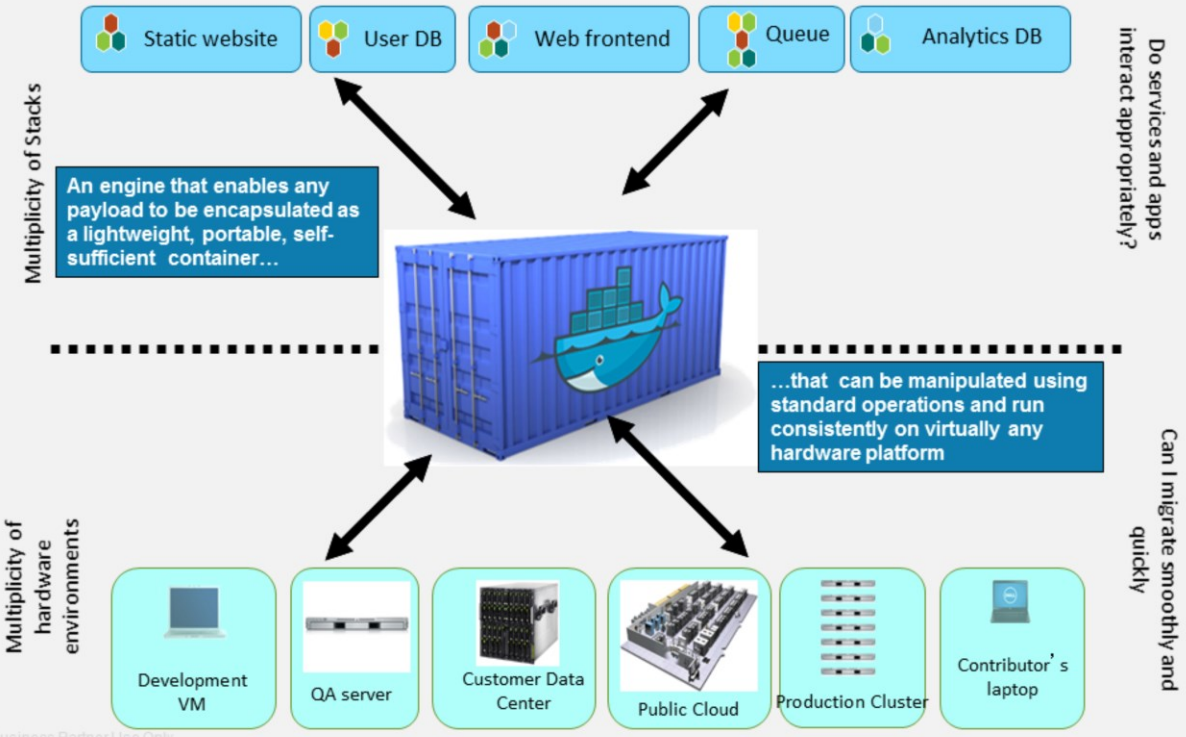
# The Matrix from Hell

| | Development VM | QA Server | Single Prod Server | Onsite Cluster | Public Cloud | Contributor's laptop | Customer Servers |
|---|---|---|---|---|---|---|---|
| Static website | ? | ? | ? | ? | ? | ? | ? |
| Web frontend | ? | ? | ? | ? | ? | ? | ? |
| Background workers | ? | ? | ? | ? | ? | ? | ? |
| User DB | ? | ? | ? | ? | ? | ? | ? |
| Analytics DB | ? | ? | ? | ? | ? | ? | ? |
| Queue | ? | ? | ? | ? | ? | ? | ? |

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

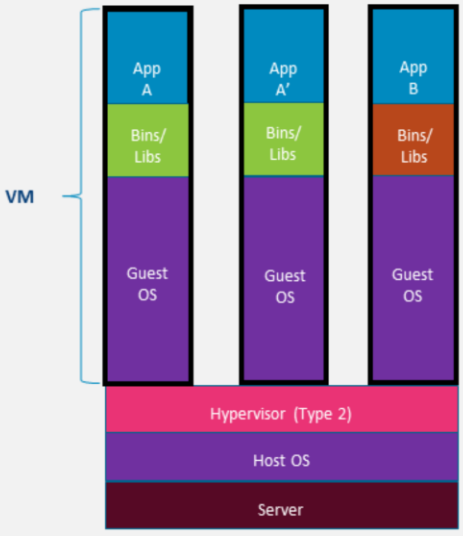
# A Shipping Container for Code




BM 2017 Internal and Business Partner Use Only

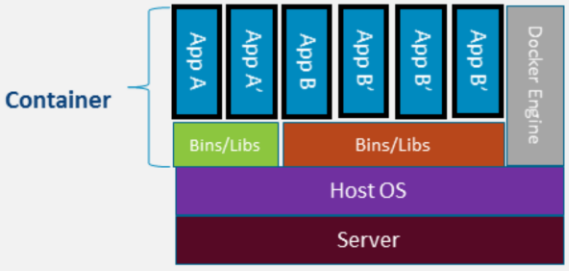
© Copyright IBM Corporation 2016

# VM's vs. Containers




Containers are isolated, but share OS and, where appropriate, bins/libraries

…faster, less overhead




BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Dev vs. Ops

| Dev | Ops |
| --- | --- |
| • Code<br>• Libraries<br>• Configuration<br>• Server runtime<br>• OS | • Logging<br>• Remote access<br>• Network configuration<br>• Monitoring |

## Separation of concerns

A container separates and bridges the Dev and Ops in DevOps

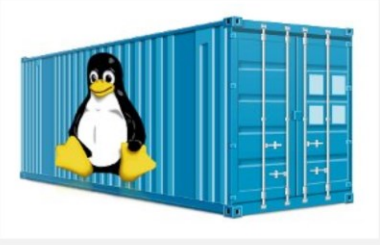
- Dev focuses on the application environment
- Ops focuses on the deployment environment

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Container Advantages

- Containers are portable
- Containers are easy to manage
- Containers provide “just enough” isolation
- Containers use hardware more efficiently
- Containers are immutable



BM 2017 Internal and Business Partner Use Only

- Containers are portable
  - Any platform with a container engine can run containers
- Containers are easy to manage
  - Container images are easy to share, download, and delete
    - Especially with Docker registries
  - Container instances are easy to create and delete
  - Each container instance is easy and fast to start and stop
- Containers provide “just enough” isolation
  - More lightweight than virtual machines
  - Processes share the operating system kernel but are segregated

© Copyright IBM Corporation 2016

- Containers use hardware more efficiently
  - Greater density than virtual machines
    - Especially Docker containers, which can share layers
- Containers are immutable
  - Container images are versions
  - Containers cannot (should not) be patched

© Copyright IBM Corporation 2016

# Docker Containers

© Copyright IBM Corporation 2016

# Docker Adoption

Docker enables application development **efficiency**, making deployment more **efficient**, eliminating vendor 'lock-in' with true **portability**

- Open software
- Open contribution
- Open design
- Open governance

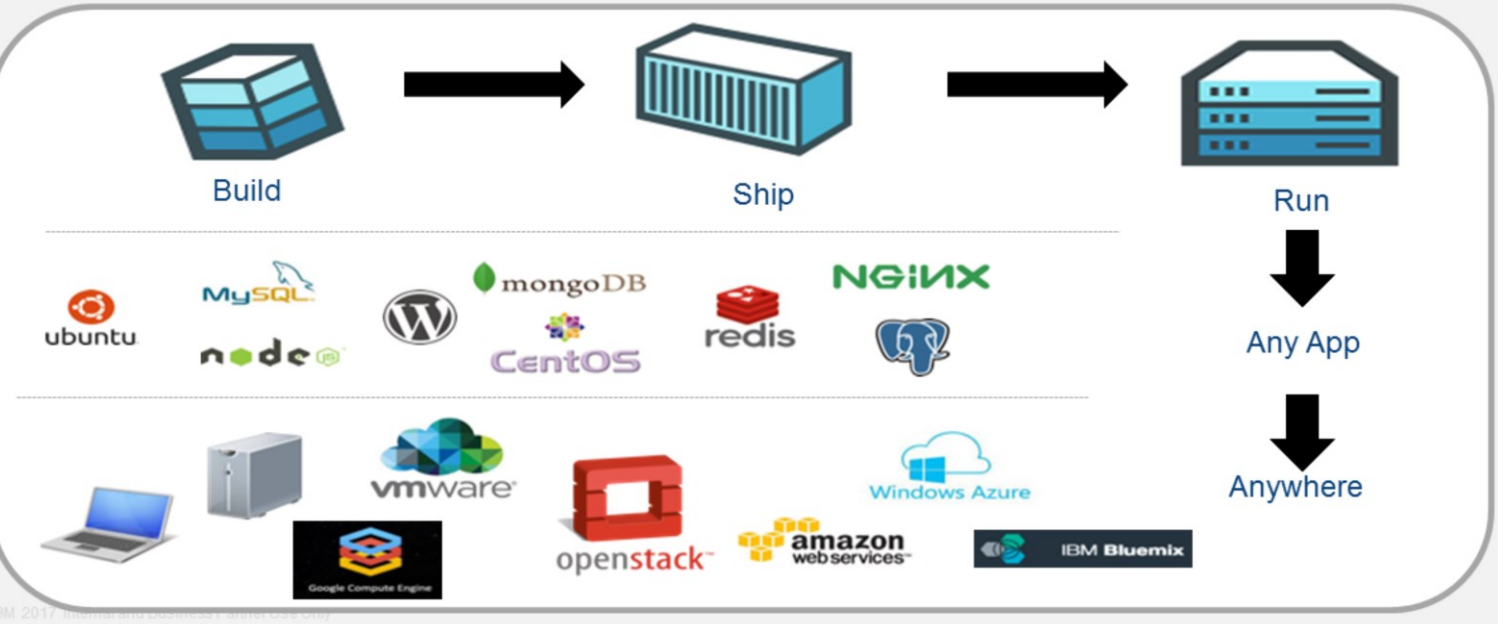
BM 2017 Internal and Business Partner Use Only

- Open software
  - Launched **March 2013**
  - 2.0+ billion downloads of Docker images
- Open contribution
  - 2000+ contributors
  - #2 most popular project
  - 185 community meet-up groups in 58 countries
- Open design
  - Contributors include IBM, Red Hat, Google, Microsoft, VMware, AWS, Rackspace, and others
- Open governance
  - 12 member governance advisory board selected by the community

© Copyright IBM Corporation 2016

# Docker Mission

Docker is an **open platform** for building distributed applications for **developers** and **system administrators**

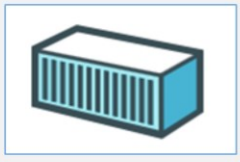



© Copyright IBM Corporation 2016

# Docker Components

**Image**

- A read-only snapshot of a container stored in Docker Hub to be used as a template for building containers



**Container**

- The standard unit in which the application service resides or transported


**Docker Hub/Registry**

- Available in SaaS or Enterprise to deploy anywhere you choose
- Stores, distributes, and shares container images

**Docker Engine**

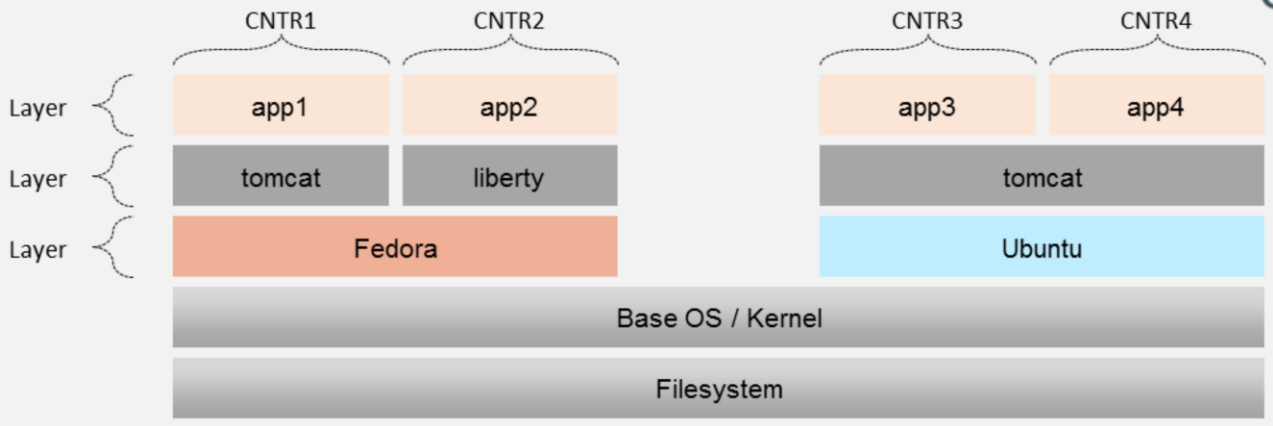
- A program that creates, ships, and runs application containers
- Runs on any physical and virtual machine or server locally, in private or public cloud
- Client communicates with Engine to execute commands

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Docker Containers

- Docker uses a copy-on-write (union) filesystem
- New files (& edits) are only visible to current/above layers




- Layers allow for reuse
  - More containers per host
  - Faster start-up/download time – base layers are "cached"
- Images
  - Tarball of layers (each layer is a tarball)

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Containers

Everyone's container journey starts with one container….

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Containers

At first the growth is easy to handle….

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Containers


BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Orchestration

© Copyright IBM Corporation 2016

# More to Containers than just Docker


Serverless

APACHE OpenWhisk™

PaaS

CLOUD FOUNDRY

Container Orchestration

Apache MESOS

kubernetes

Swarm/
Swarm Mod

Container Engine

OPEN CONTAINER INITIATIVE

rkt

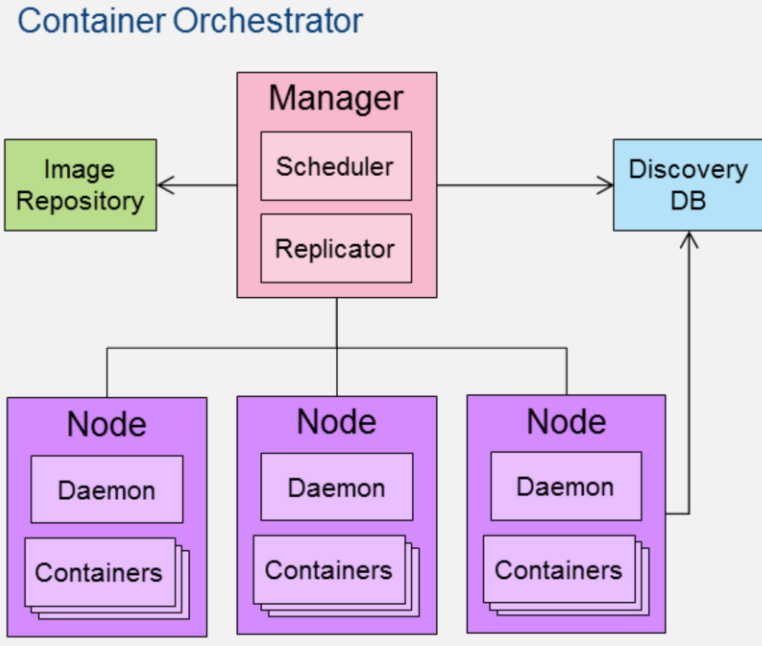
docker

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# What is Container Orchestration?

- Container orchestration
  - Cluster management
  - Scheduling
  - Service discovery
  - Replication
  - Health management




BM 2017 Internal and Business Partner Use Only

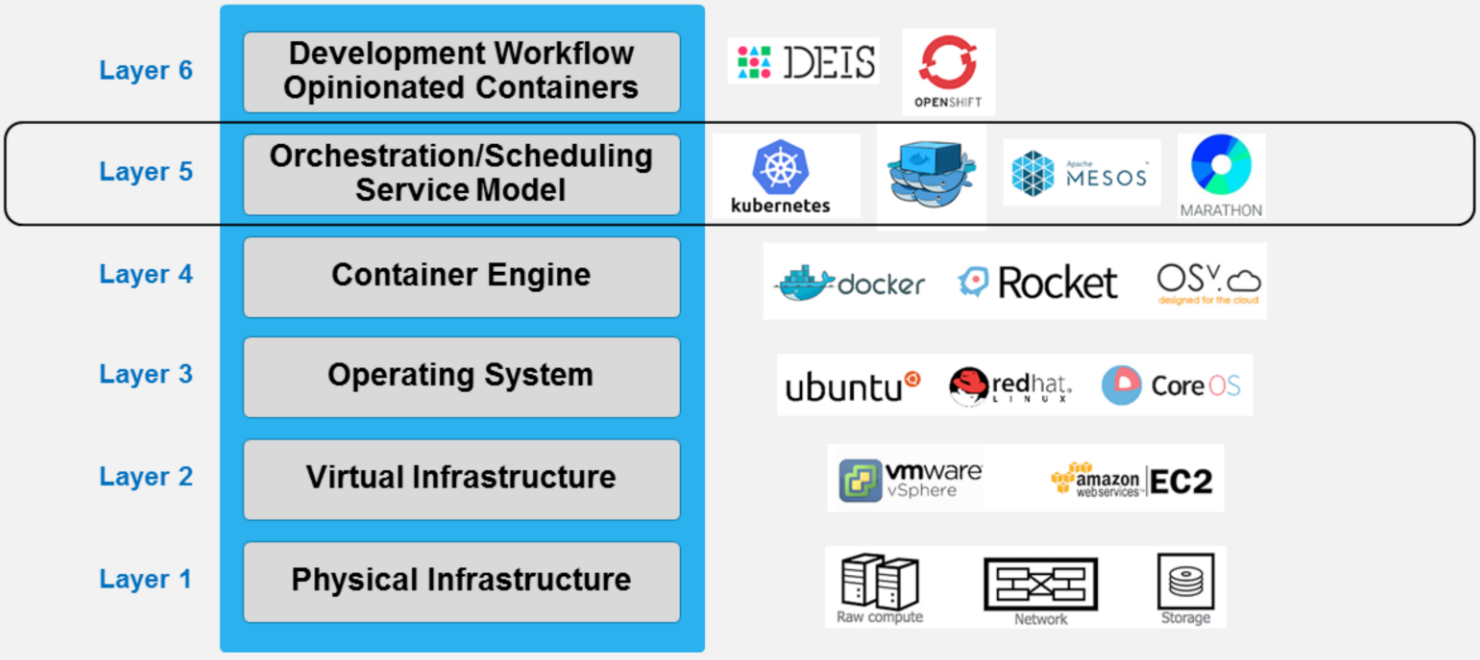
- Container orchestration
  - Manages the deployment, placement, and lifecycle of workload containers
- Cluster management
  - Federates multiple hosts into one target
- Scheduling
  - Distributes containers across nodes
- Service discovery
  - Knows where the containers are located
  - Distributes client requests across the containers
- Replication
  - Ensures the right number of nodes and containers

© Copyright IBM Corporation 2016

- Health management
  - Replaces unhealthy containers and nodes

© Copyright IBM Corporation 2016

# Container Ecosystem Layers




BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Kubernetes

© Copyright IBM Corporation 2016

# What is Kubernetes?

- Container orchestrator
- Manage applications, not machines
- Designed for extensibility
- Open source project managed by the Linux Foundation

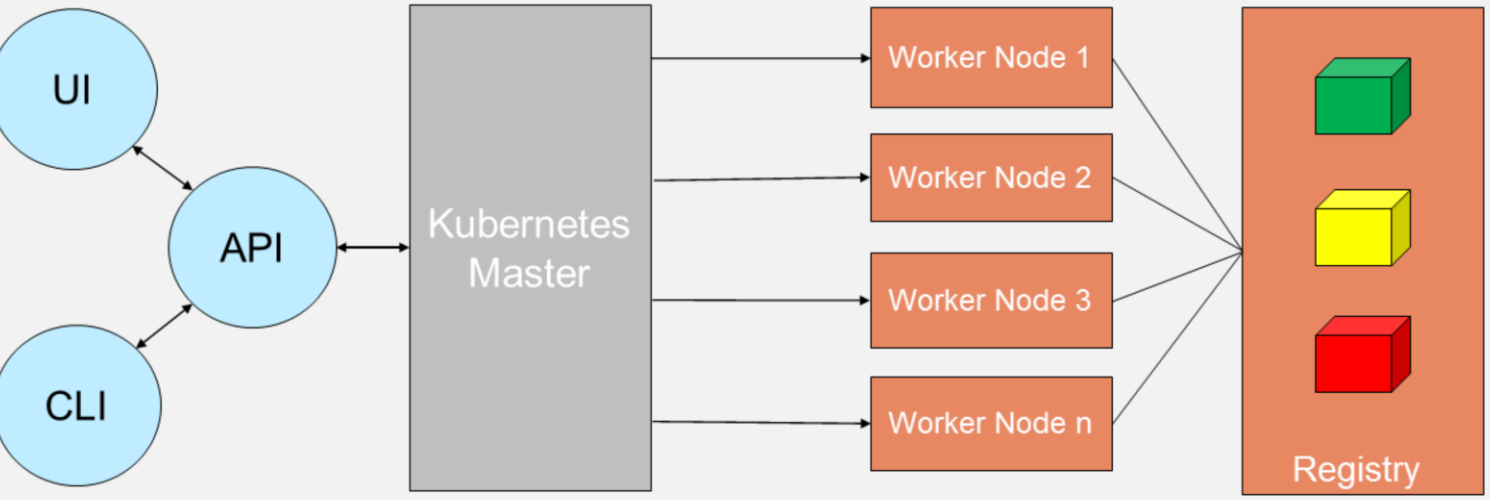
- Container orchestrator
  - Runs and manages containers
  - Unified API for deploying web applications, batch jobs, and databases
  - Maintains and tracks the global view of the cluster
  - Supports multiple cloud and bare-metal environments
- Manage applications, not machines
  - Rolling updates, canary deploys, and blue-green deployments
- Designed for extensibility
  - Rich ecosystem of plug-ins for scheduling, storage, networking

© Copyright IBM Corporation 2016

- Open source project managed by the Linux Foundation
  - Inspired and informed by Google's experiences and internal systems
  - 100% open source, written in Go

© Copyright IBM Corporation 2016

# Kubernetes Architecture



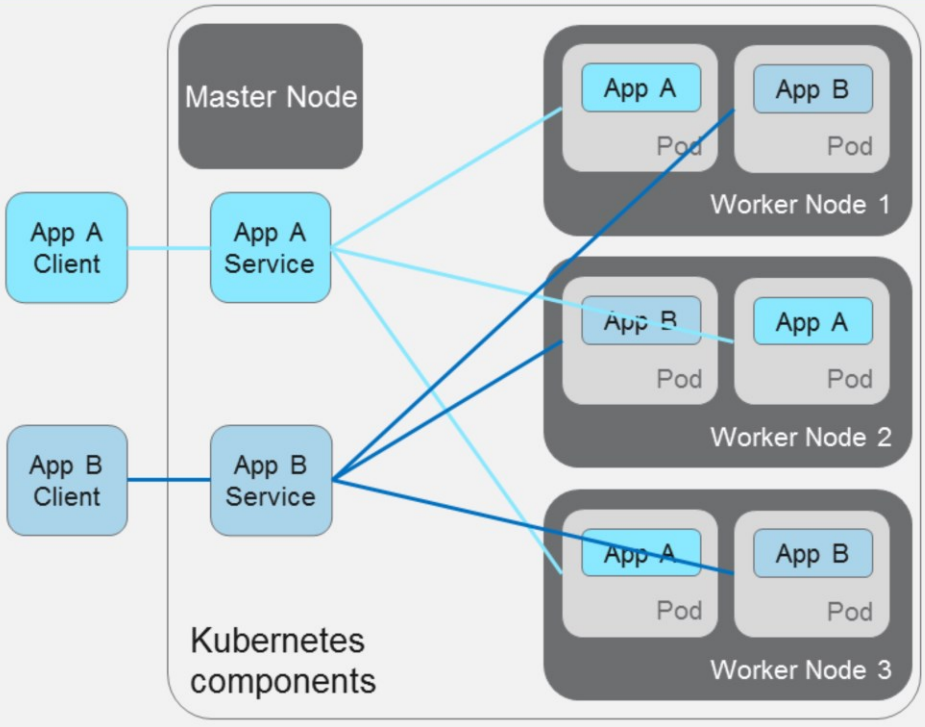

- Etcd
- API Server
- Controller Manager Server
- Scheduler

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Kubernetes Architecture: Workloads

- Container
  - Packaging of an app
- Pod
  - Unit of deployment
- Service
  - Fixed endpoint for 1+ pods

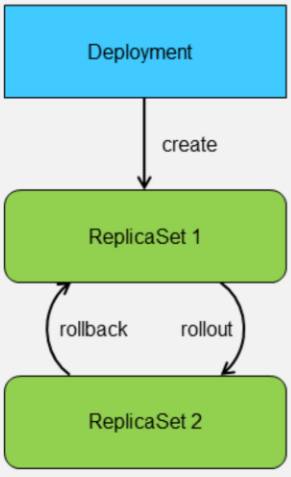


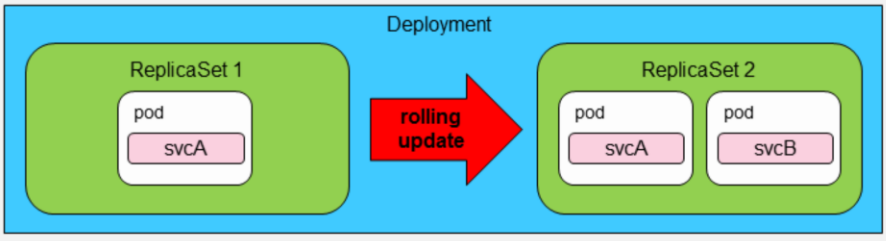

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Kubernetes Terminology: Deployment

- Deployment
  - A set of pods to be deployed together, such as an application
  - Declarative: Revising a Deployment creates a ReplicaSet describing the desired state
  - Rollout: Deployment controller changes the actual state to the desired state at a controlled rate
  - Rollback: Each Deployment revision can be rolled back
  - Scale and autoscale: A Deployment can be scaled
- ReplicaSet
  - The next-generation ReplicationController
  - A set of pod templates that describe a set of pod replicas
  - Uses a template that describes specifically what each pod should contain
  - Ensures that a specified number of pod replicas are running at any given time



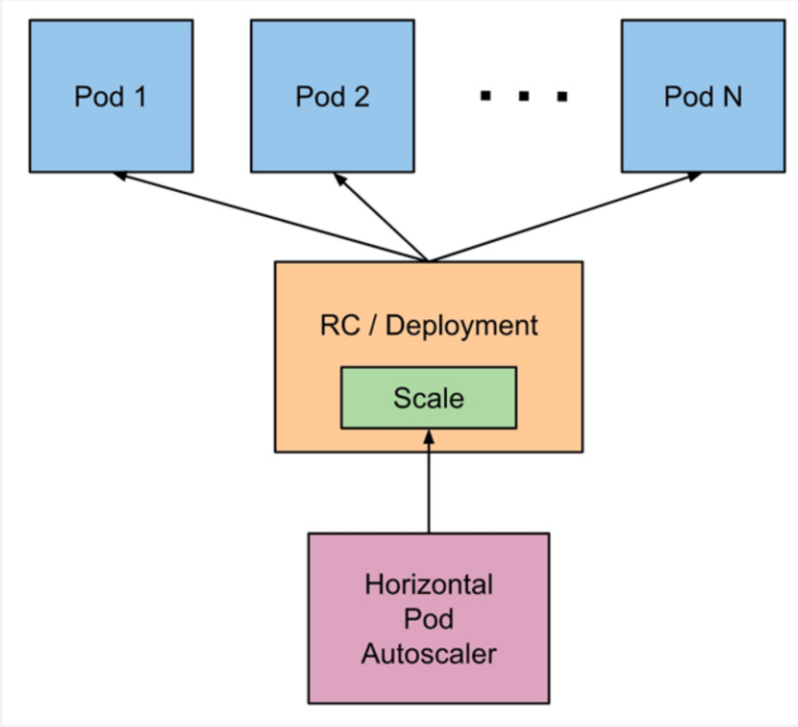

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Kubernetes Terminology: Autoscaling

- Horizontal Pod Autoscaling (HPA)

```
$ kubectl autoscale deployment
<deployment-name> --cpu-percent=50
  --min=1 --max=10 deployment
"<hpa-name>" autoscaled
```




BM 2017 Internal and Business Partner Use Only

- Horizontal Pod Autoscaling (HPA)
  - Automatically scales the number of pods in a replication controller, deployment, or replica set
  - Matches the observed average CPU utilization to the specified target
  - Fetches metrics in two different ways: direct Heapster access and REST client access
  - Kubernetes Heapster enables container cluster monitoring and performance analysis
  - Default config: query every 30 sec, maintain 10% tolerance, wait 3 min after scale-up, wait 5 min after scale-down

```
$ kubectl autoscale deployment
<deployment-name> --cpu-percent=50
```

© Copyright IBM Corporation 2016

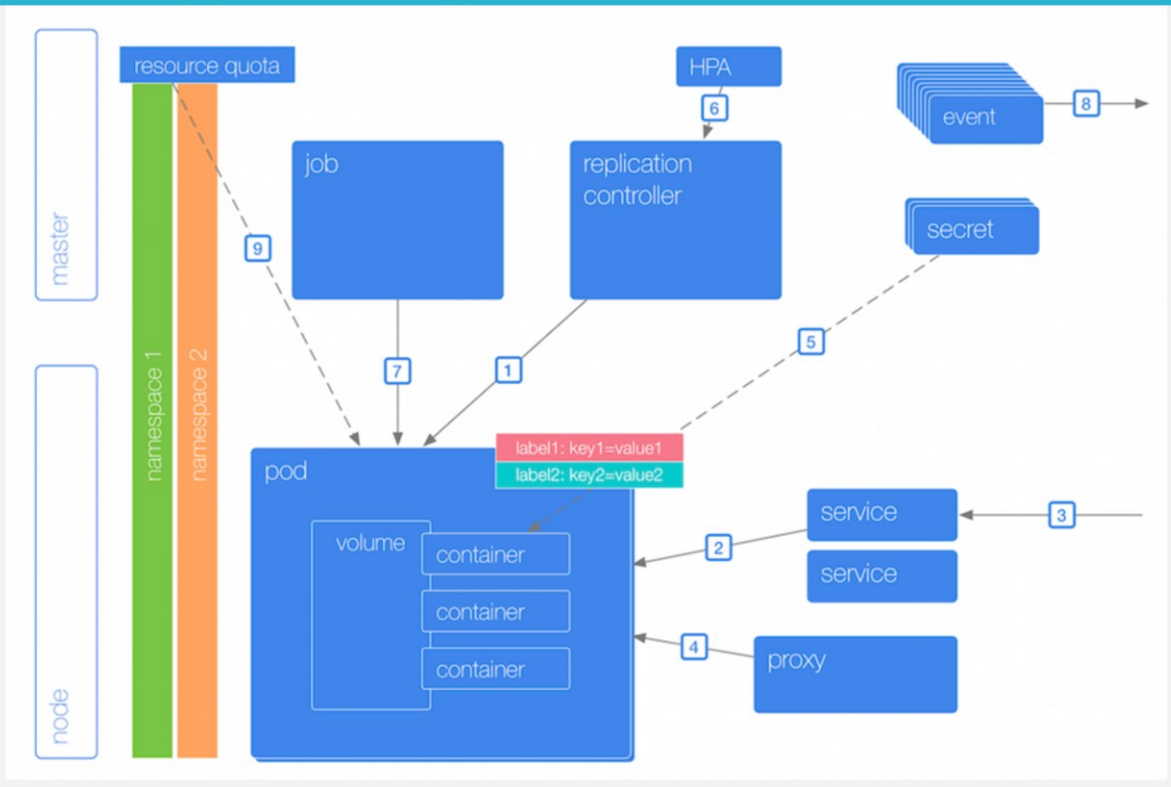
`--min=1 --max=10 deployment "<hpa-name>" autoscaled`

- Creates a horizontal pod autoscaler
  - An HPA instance
  - Maintains between 1 and 10 replicas of the pods controlled by the deployment
  - Maintains an average CPU utilization across all pods of 50%

© Copyright IBM Corporation 2016

# Kubernetes Terminology: Naming

- Name
- Namespace
- Resource Quota
- Label
- Selector
- ConfigMap
- Secrets




BM 2017 Internal and Business Partner Use Only

- Name
  - Each resource object by type has a unique name
- Namespace
  - Resource isolation: Each namespace is a virtual cluster within the physical cluster
    - Resource objects are scoped within namespaces
    - Low-level resources are not in namespaces: nodes, persistent volumes, and namespaces themselves
    - Names of resources need to be unique within a namespace, but not across namespaces
  - Resource quotas: Namespaces can divide cluster resources
  - Initial namespaces
    - `default` – The default namespace for objects with no other namespace

© Copyright IBM Corporation 2016

  - `kube-system` – The namespace for objects created by the Kubernetes system

- Resource Quota
  - Limits resource consumption per namespace
  - Limit can be number of resource objects by type (pods, services, etc.)
  - Limit can be total amount of compute resources (CPU, memory, etc.)
  - Overcommit is allowed; contention is handled on a first-come, first-served basis
- Label
  - Metadata assigned to Kubernetes resources (pods, services, etc.)
  - Key-value pairs for identification
  - Critical to Kubernetes as it relies on querying the cluster for resources that have certain labels
- Selector
  - An expression that matches labels to identify related resources

- ConfigMap
  - Configuration values to be used by containers in a pod
  - Stores configuration outside of the container image, making containers more reusable

© Copyright IBM Corporation 2016

- Secrets
  - Sensitive info that containers need to read or consume
  - Encrypted in special volumes mounted automatically

© Copyright IBM Corporation 2016

# Common Kubernetes Commands

- Get the state of your cluster
  ```
  $ kubectl cluster-info
  ```
- Get all the nodes of your cluster
  ```
  $ kubectl get nodes -o wide
  ```
- Get info about the pods of your cluster
  ```
  $ kubectl get pods -o wide
  ```
- Get info about the replication controllers of your cluster
  ```
  $ kubectl get rc -o wide
  ```
- Get info about the services of your cluster
  ```
  $ kubectl get services
  ```
- Get full config info about a Service
  ```
  $ kubectl get service
  NAME_OF_SERVICE -o json
  ```
- Get the IP of a Pod
  ```
  $ kubectl get pod NAME_OF_POD -
  template={{.status.podIP}}
  ```
- Delete a Pod
  ```
  $ kubectl delete pod NAME
  ```
- Delete a Service
  ```
  $ kubectl delete service
  NAME_OF_SERVICE
  ```

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# lelm & Helm Chart

- Helm
  - Package manager for Kubernetes
  - Used to manage Kubernetes applications
- Helm Chart
  - Used to define, install, and upgrade complex Kubernetes applications
  - Easy to create, version, share and publish
  - Expressed in “Yet Another Markup Language” (YAML) files


BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Conclusion

© Copyright IBM Corporation 2016

# Conclusion

- Why containers?
- Docker containers
- Container orchestration
- Kubernetes

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016

# Resources

- Docker tutorial
  - https://docs.docker.com/get-started/
- Kubernetes tutorial
  - https://kubernetes.io/docs/tutorials/kubernetes-basics/
- The Evolution of Linux Containers and Their Future
  - https://dzone.com/articles/evolution-of-linux-containers-future
- Introduction to container orchestration
  - https://www.exoscale.ch/syslog/2016/07/26/container-orch/
- TNS Research: The Present State of Container Orchestration
  - https://thenewstack.io/tns-research-present-state-container-orchestration/
- Large-scale cluster management at Google with Borg
  - https://research.google.com/pubs/pub43438.html

BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016


BM 2017 Internal and Business Partner Use Only

© Copyright IBM Corporation 2016