SYNERGETICS
GET IT RIGHT

Microsoft Partner
Gold Cloud Platform
Silver Learning

Docker & Containers

Envisioning Application Modernization with Containers

Agenda

MODULE 1: INTRODUCTION TO CONTAINERS

MODULE 2: CONTAINER ARCHITECTURE

MODULE 3: DOCKER ARCHITECTURE

MODULE 4: APPLICATION MODERNIZATION

MODULE 5: UNDERSTANDING ORCHESTRATION

Module 1

Introduction to Containers

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Application packaging strategies

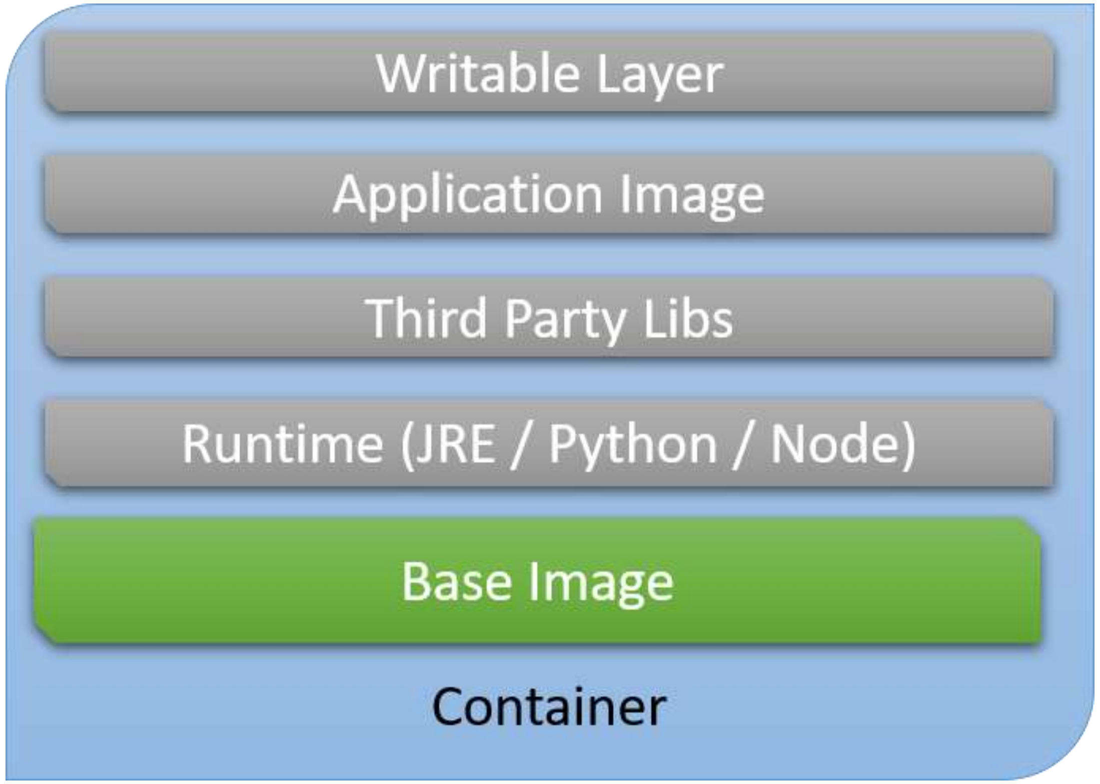
- Every Application has certain dependencies.
  - Libraries provided by Operating System
  - Libraries provided by runtime environment like Java, Python & Dot Net runtime
  - Server Runtime like Tomcat for Java, IIS for Asp.net, Apache httpd
  - Third Party Libraries
- Change in Dependencies affects Application.

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Application packaging strategies /challenges

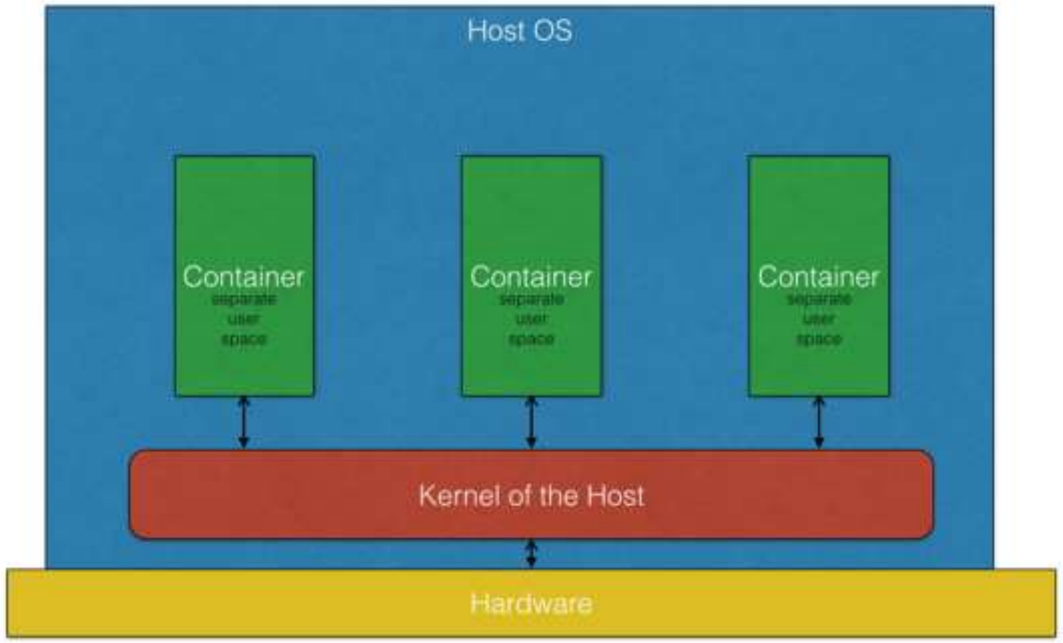
- Application goes through following phases:
  - Development
  - Testing
  - Staging
  - Production
- Managing dependencies across all these environments could be challenging.
- Creating compatible dev-test environment may take considerable time.
- Question : Challenges in moving application between environments
- Question: List Existing solutions based on Virtualization

Containers as alternative



- A Typical container would have application & all its dependencies packed together.
- This makes container "decoupled" from targeted environment.
- Only restriction being Windows or Linux platform.

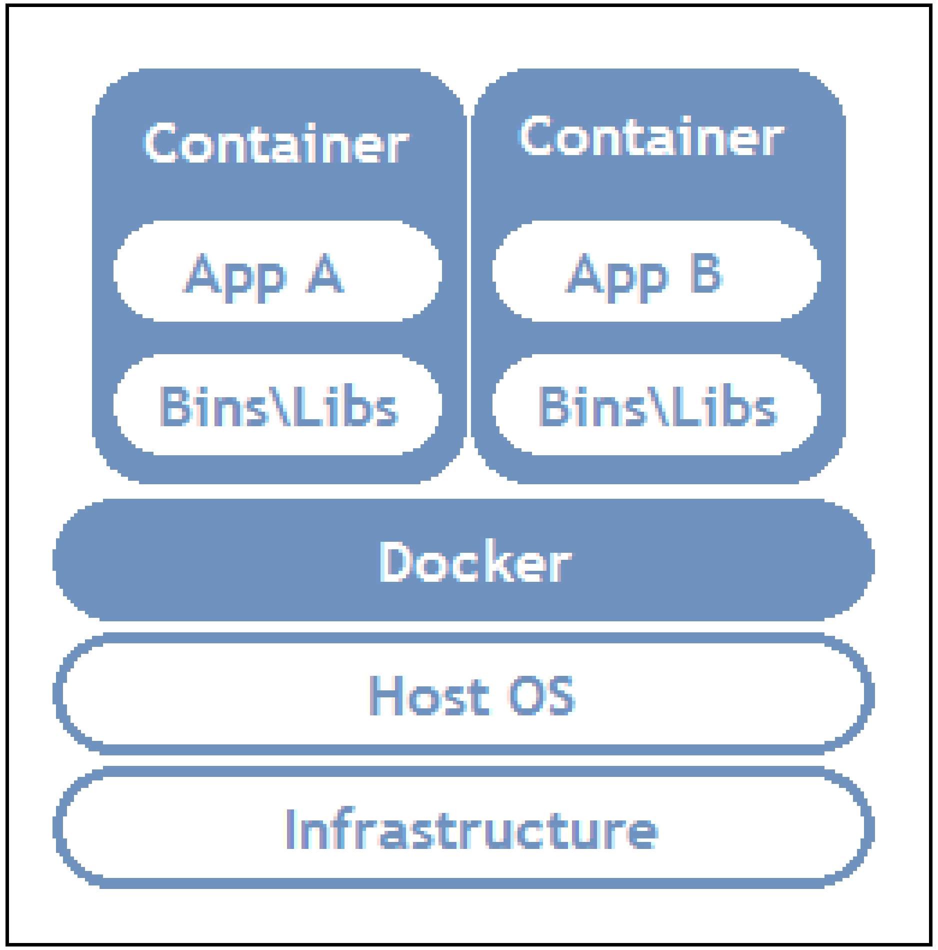
Benefits of Containers



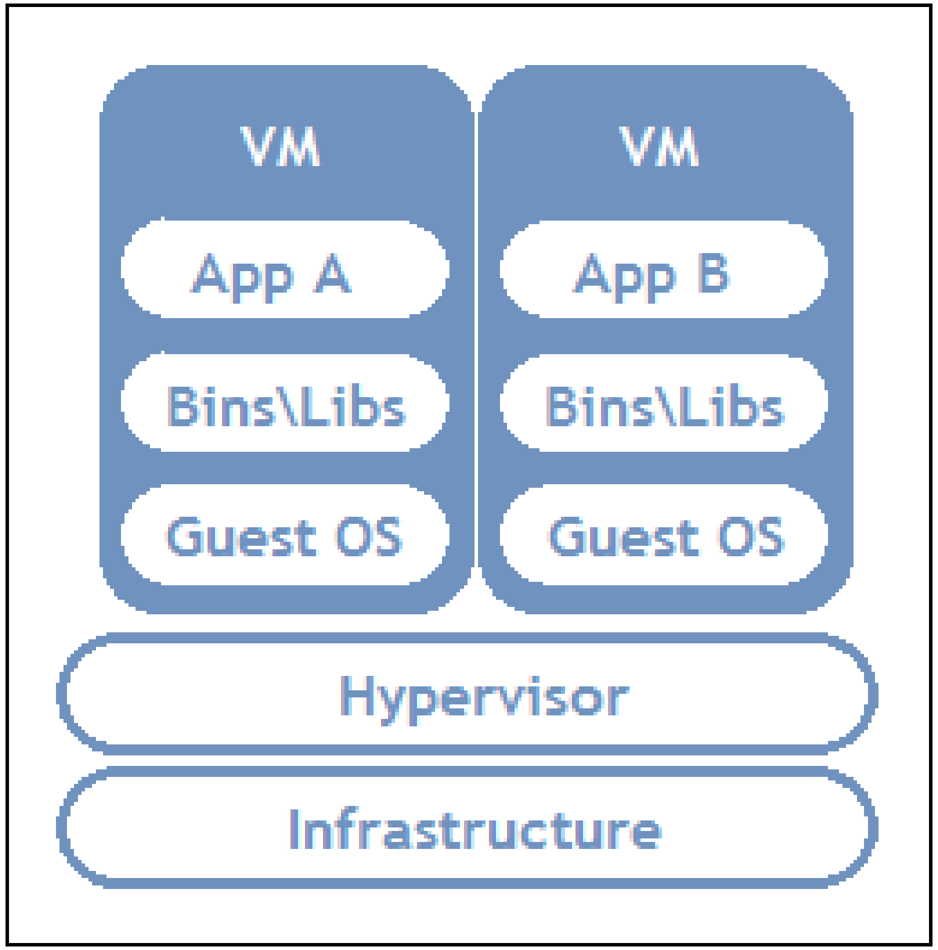
- Isolate application from underlying OS and Hardware.
- Lightweight, provides higher density than Virtual machines.
- Dev-Ops ready. Many CI/CD platforms support container deployments/build.
- Every container executes in a separate user space.

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Container Based Implementation



Virtual Machine Implementation



Container vs Virtual machines

- Virtual machines provides hardware level virtualization.
- Virtual machines contains Operating system.
- Virtual machines use guest OS kernel.
- Host OS and Guest OS could be completely different.
- Booting up VM is equivalent of booting a physical machine.

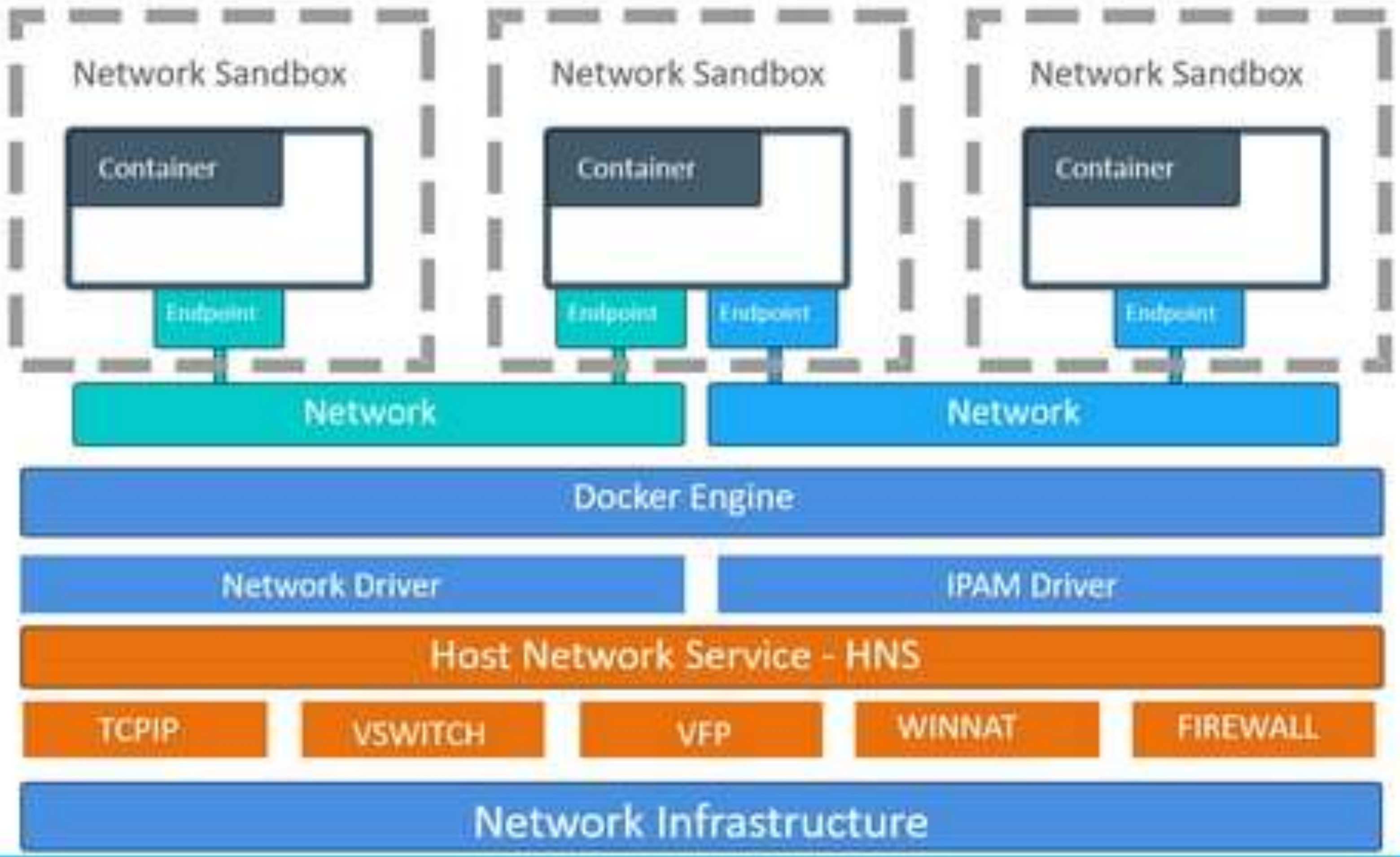
- Containers provide software level isolation, no hardware virtualization.
- Containers use “base image” which provide basic OS level libraries but not the OS.
- Containers use host OS kernel.
- Due to dependence on Host OS Kernel, containers must be deployed on compatible host OS.
- Launching container is equivalent of launching an application.

Container platforms

- Docker from Docker Inc
- LXC from Linux
- LXD from Ubuntu
- RKT from CoreOS

Microsoft Partner

Container Networking Model

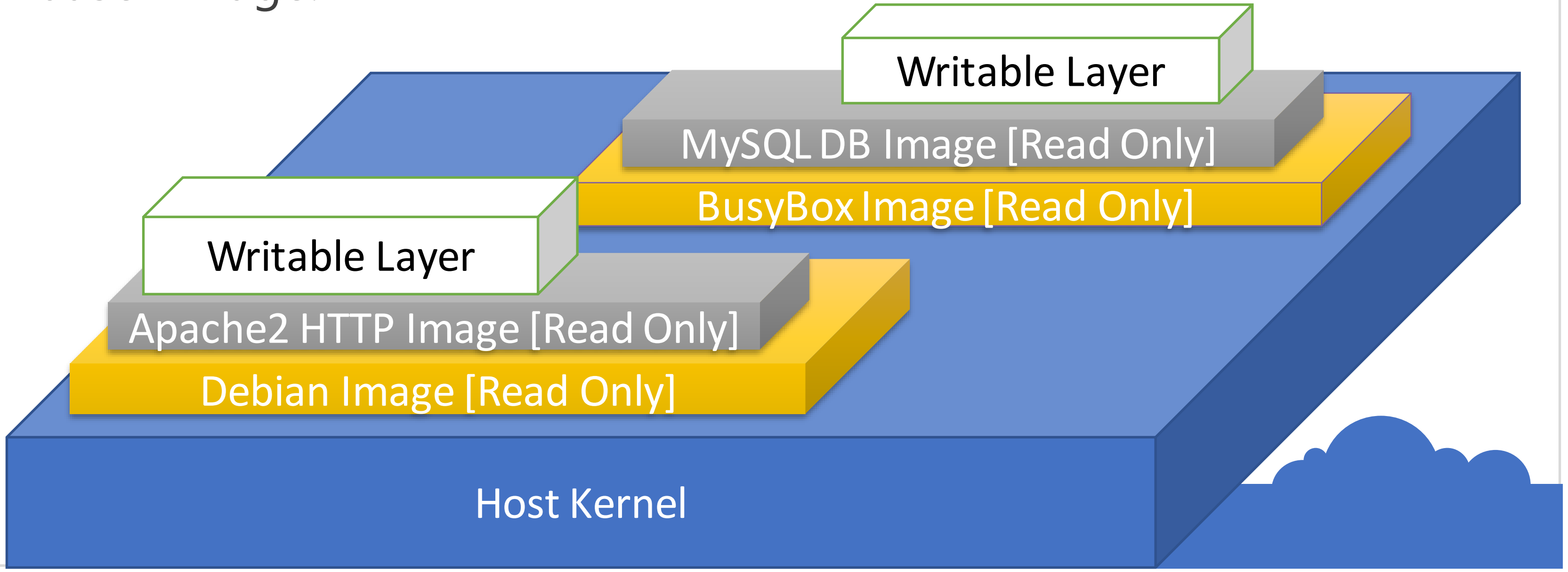


Module 2

Container Architecture

Container & Images

- Containers are running instances of image
- Image is a read-only filesystem made of “layers”
- Each “layer” is an image which may contain a specific application, library or runtime.
- Container has an extra “Writable” layer.
- The bottom-most image is “base” image.


Writable Layer
MySQL DB Image [Read Only]
BusyBox Image [Read Only]
Writable Layer
Apache2 HTTP Image [Read Only]
Debian Image [Read Only]
Host Kernel

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Read-Only vs Writable layers

- Read-only layers are shared by multiple containers / images.
- Writable layer allows read and write operations
- Every container has ONE writable layer.
- Every layer has unique ID and meta-data.

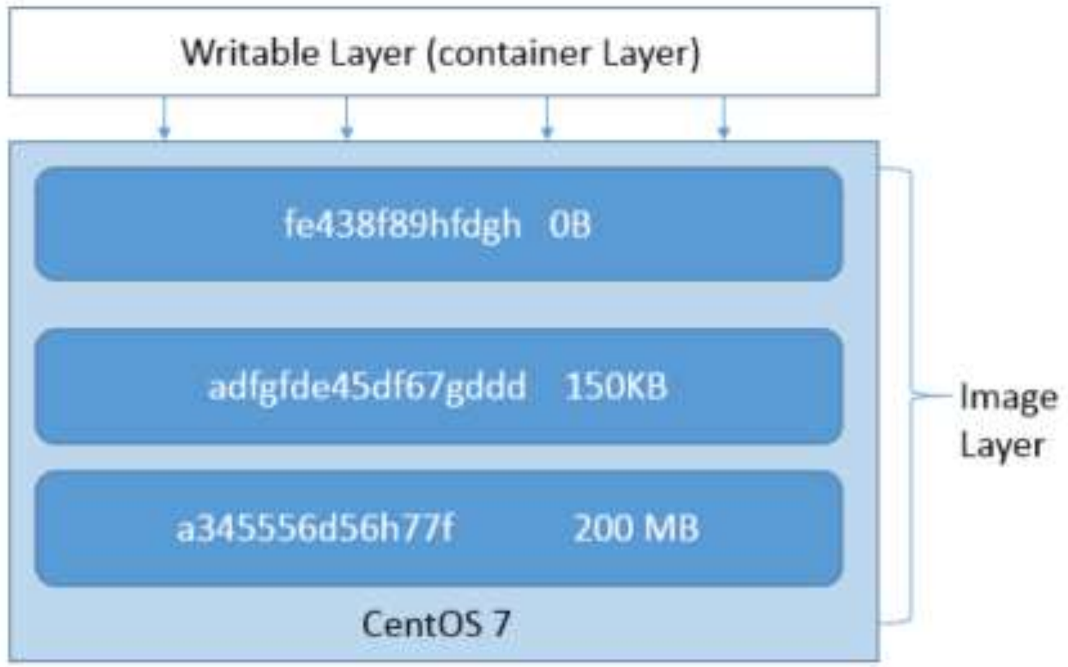
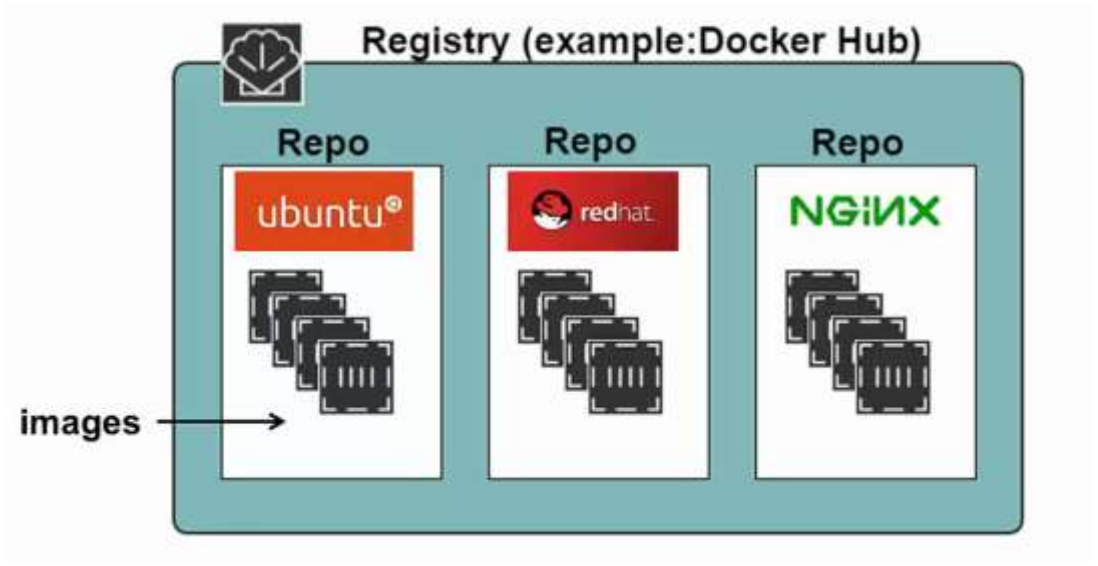


Image repositories

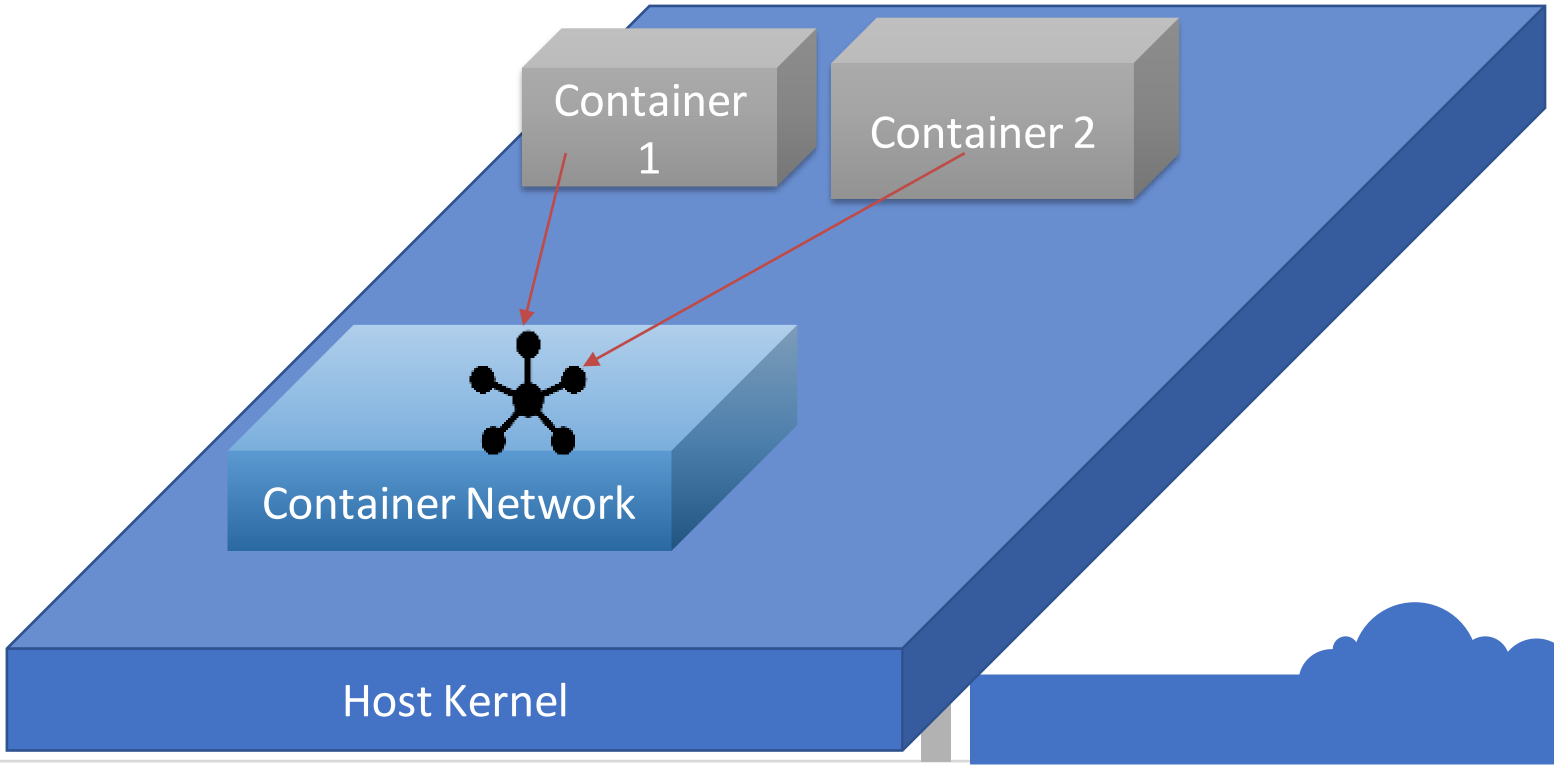
- An Image repository is persistent storage for container images
- A private repository can be accessed by selected users.
- A public repository can be accessed by any one.
- A Developer may "Push" application as container image.
- An operations team may "Pull" the same image for deployment.
- Many cloud platforms do provide "Hosted" container repositories.
- Host repositories are managed by vendor.



Container Networking

- Containers can communicate through a container network.
- Container platform like docker have

  Multiple network models.

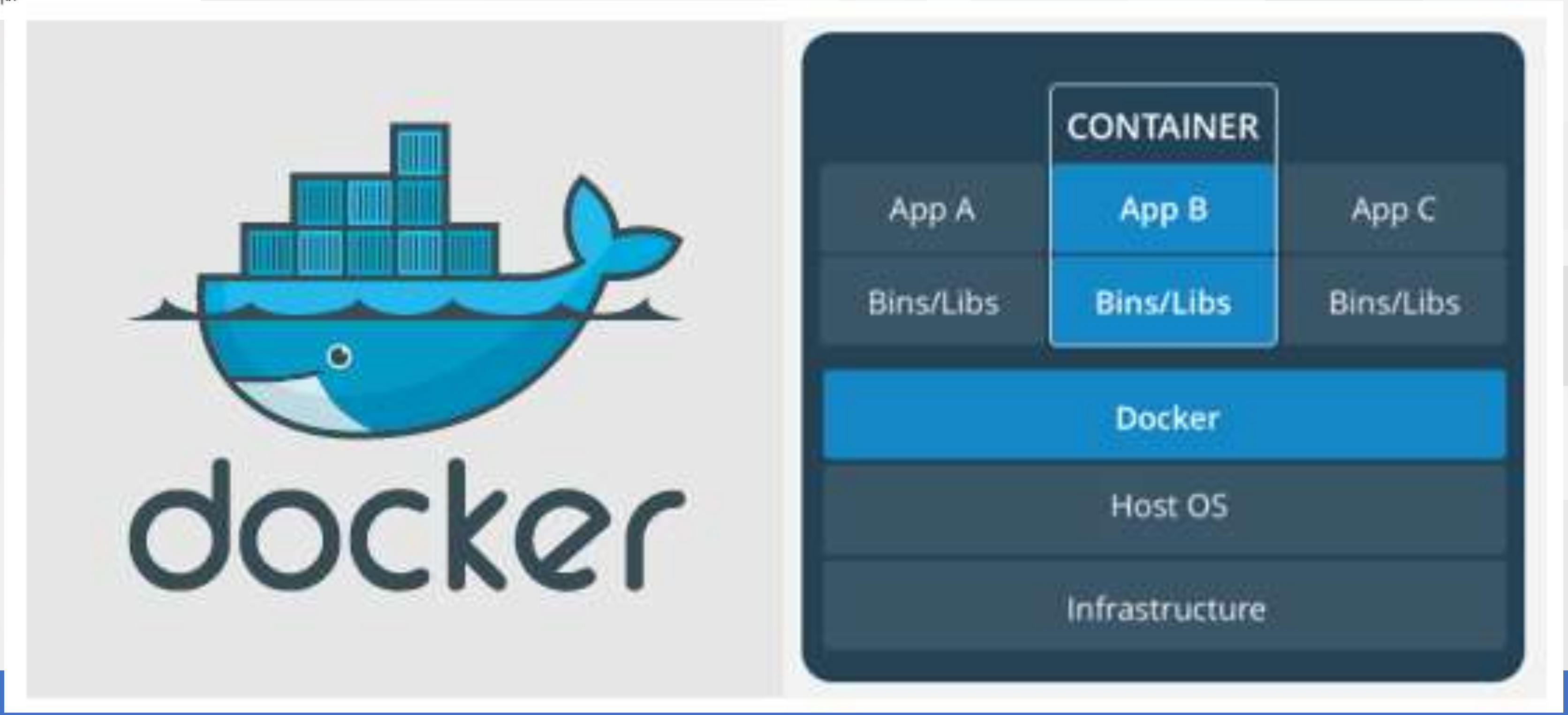


Container Networks

BRIDGE NETWORK

HOST NETWORK

OVERLAY NETWORK

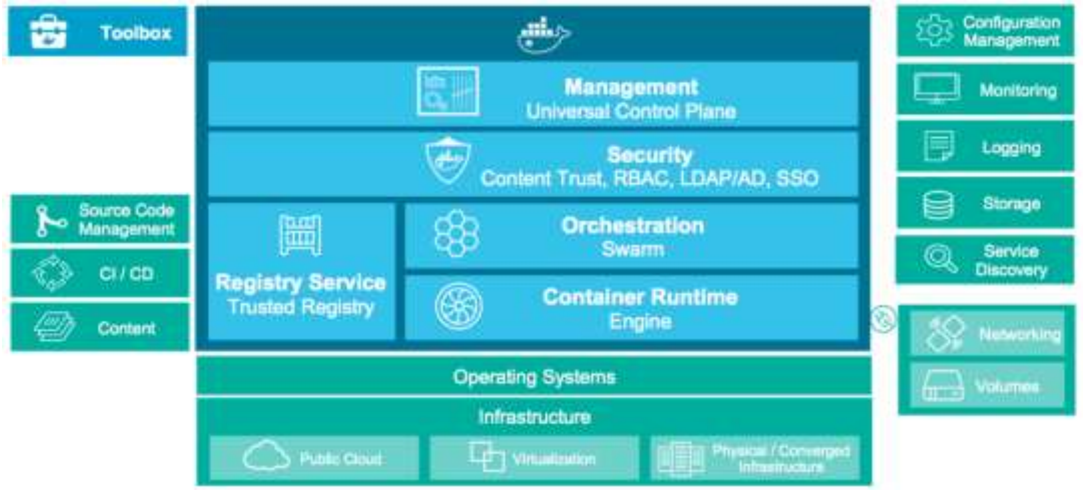


Module 03

Docker Architecture

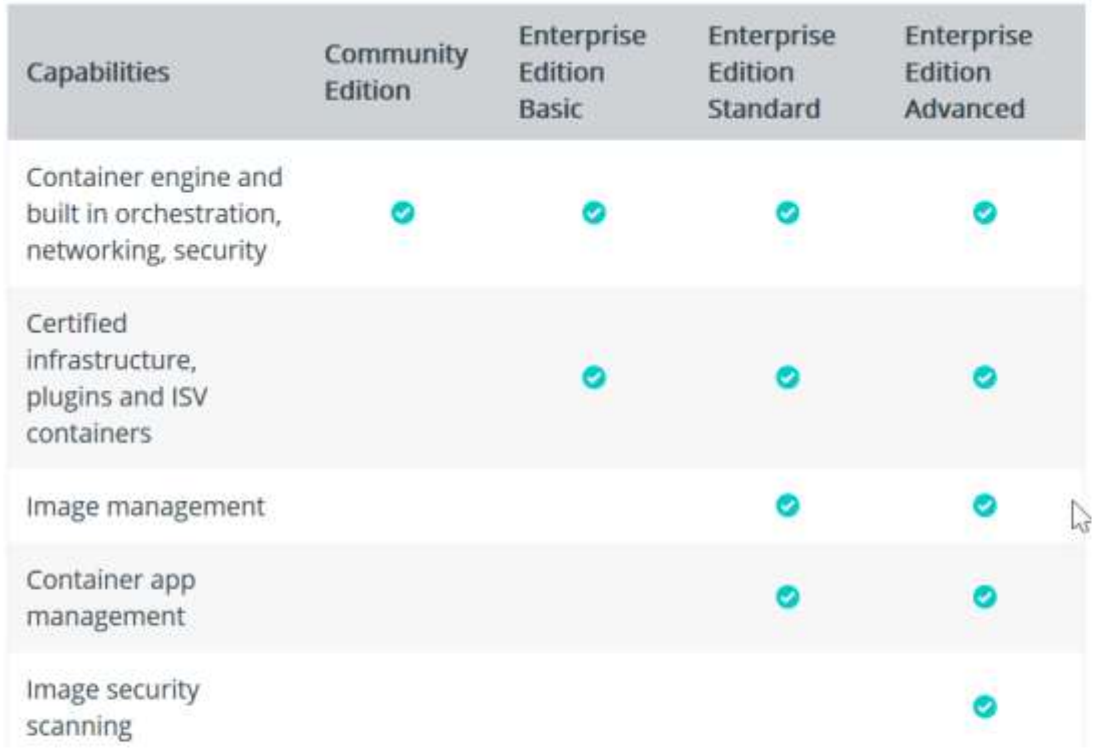
Docker As a Container platform

- An open platform for developing, shipping, and running applications.
- Started it's life as a PaaS provider by name dotCloud.
- Since its inception, working with linux container.
- The container platform was named "Docker"
- In 2013, got "rebranded" as Docker Inc.
- A container platform based on "runc" and "libcontainer"



SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Docker Editions



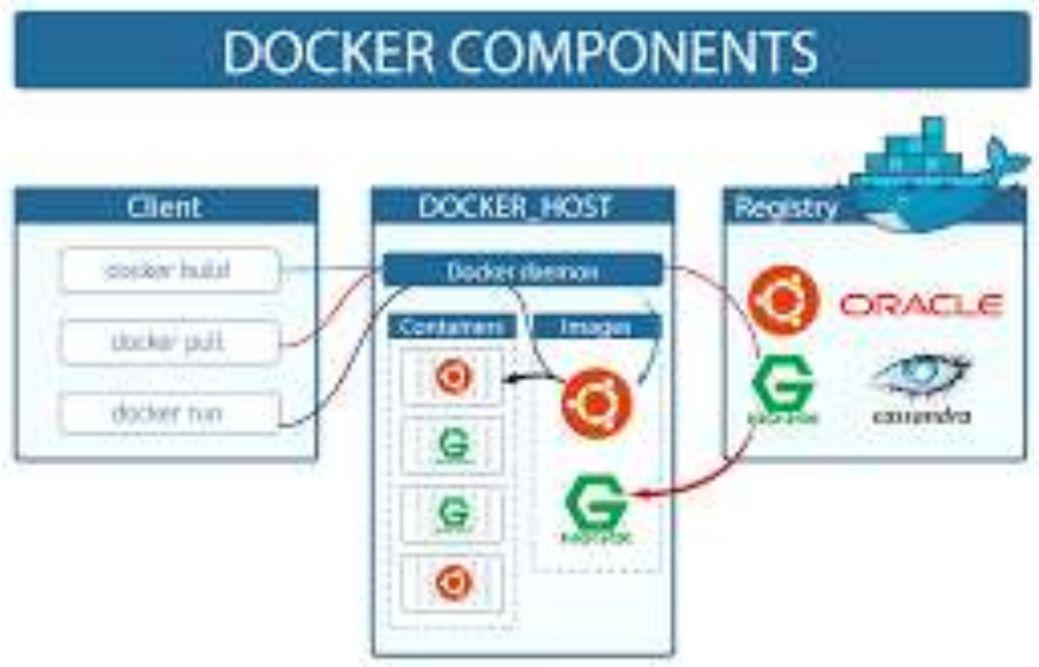
Capabilities	Community Edition	Enterprise Edition Basic	Enterprise Edition Standard	Enterprise Edition Advanced
Container engine and built in orchestration, networking, security	✓	✓	✓	✓
Certified infrastructure, plugins and ISV containers		✓	✓	✓
Image management			✓	✓
Container app management			✓	✓
Image security scanning				✓

- Community edition (Docker CE)
  - Ideal for developers and small teams experimenting with containers.
- Enterprise Edition (Docker EE)
  - Enterprise development & IT team for business critical applications at production scale.

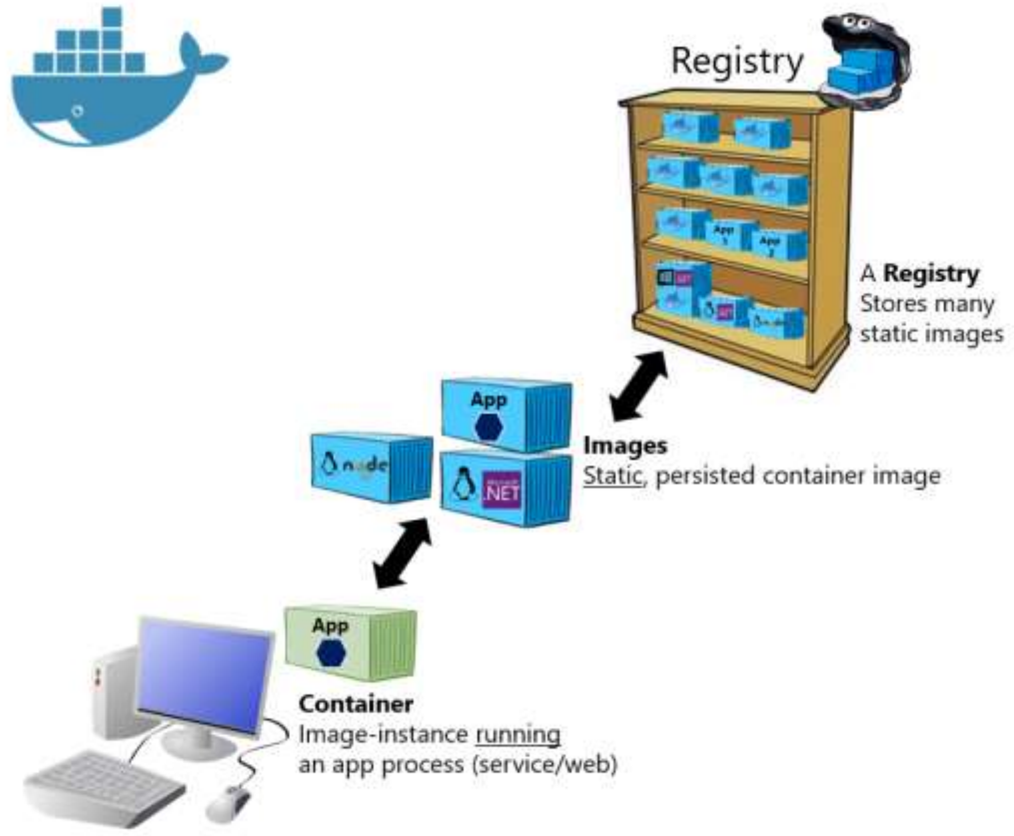
SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Docker components

- Docker engine (docker daemon)
- Docker container images
- Docker containers
- Docker CLI (client)
- Docker (REST) API
- Docker Registry



Basic taxonomy in Docker


Registry
A **Registry** Stores many static images
App
node
.NET
Images Static, persisted container image
App
Container Image-instance running an app process (service/web)

Hosted Docker Registry

Docker Trusted Registry on-prem.

→ **On-premises** ('n' private organizations)

Docker Hub Registry

Docker Trusted Registry on-cloud

Azure Container Registry

AWS Container Registry

Google Container Registry

Quay Registry

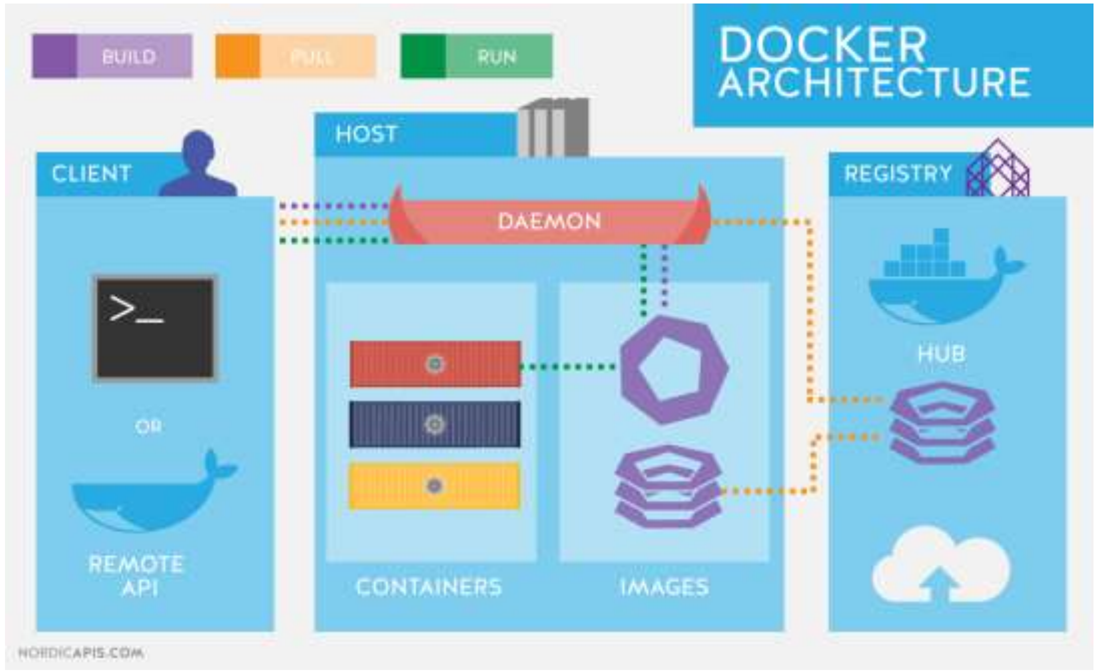
Other Cloud

→ **Public Cloud** (specific vendors)

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Docker components

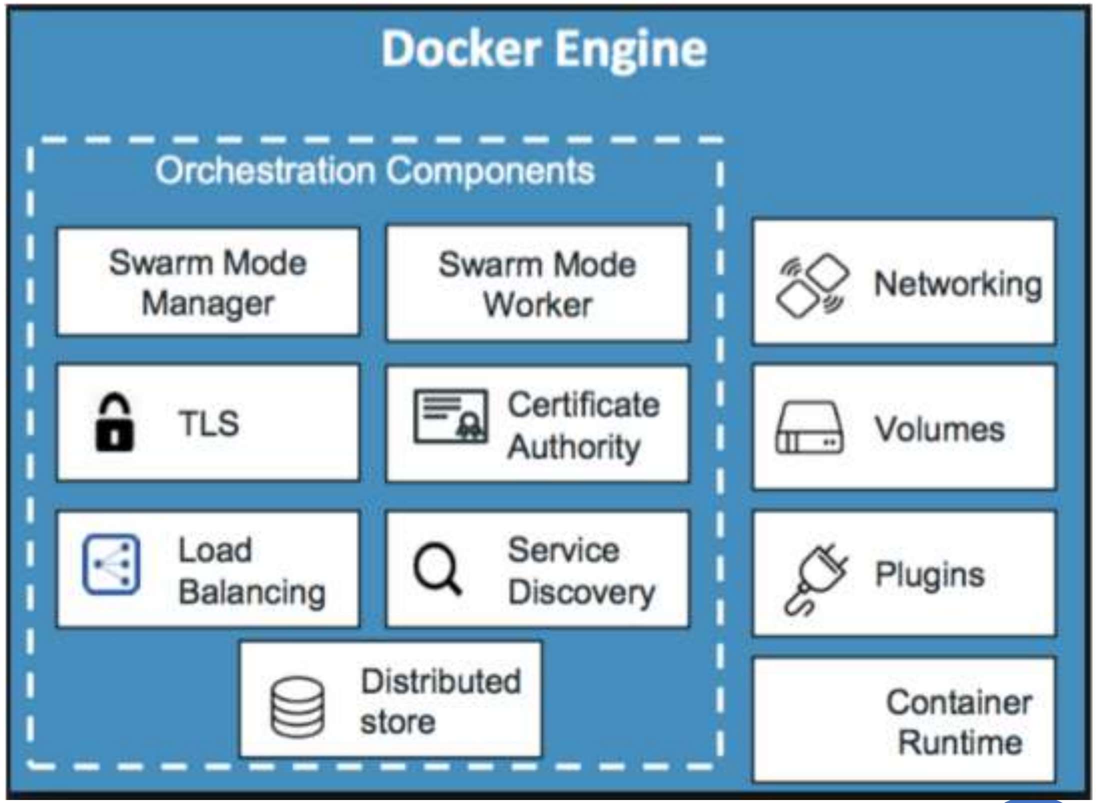
- Batteries included but "removable" ("replaceable")
- Built in security
  - Linux hosts can use AppArmor or SELinux
  - Namespaces to restrict access to host system resources.



SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Docker on Windows & Linux hosts

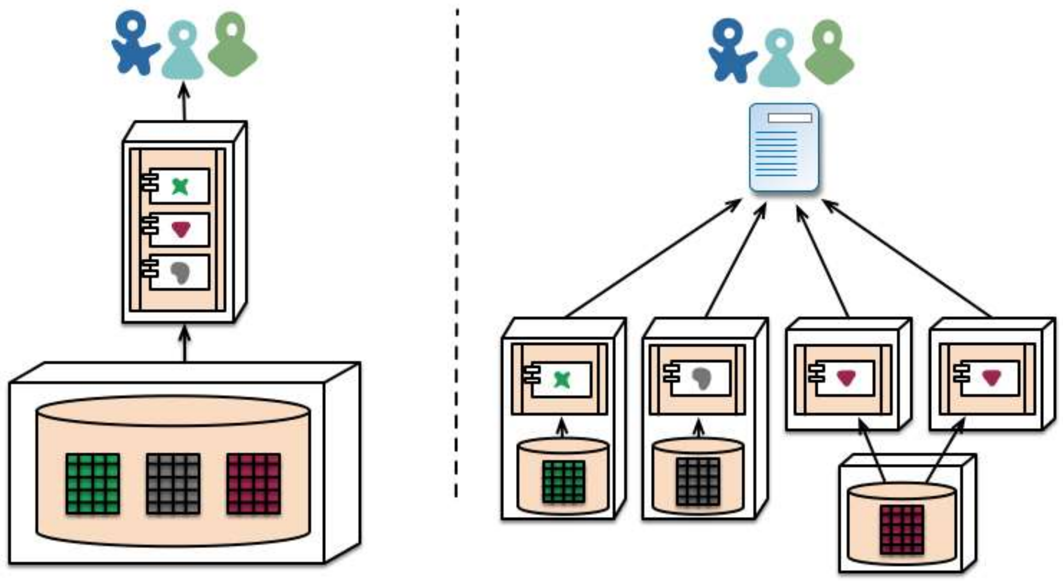
- Docker being only container platform supporting both Windows and Linux.
- Windows based container supported only on Windows 10 PRO and Windows Server 2016.
- Docker installation on windows uses different components than linux host.
- Images built for linux cannot be deployed on windows hosts.
- Windows do have a trick: docker-machine and moby project to run linux containers through a lightweight VM.



Module 04

Application modernization

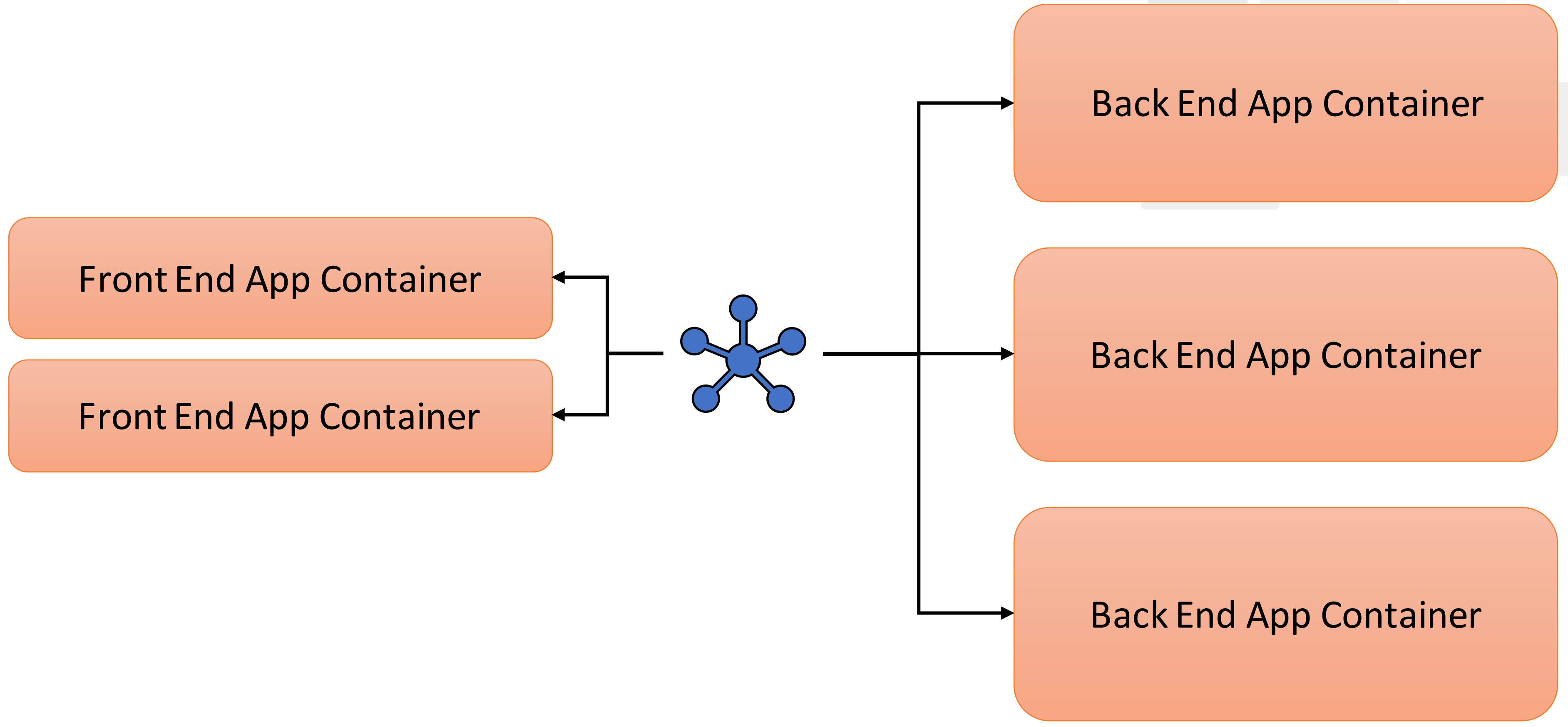
Application types



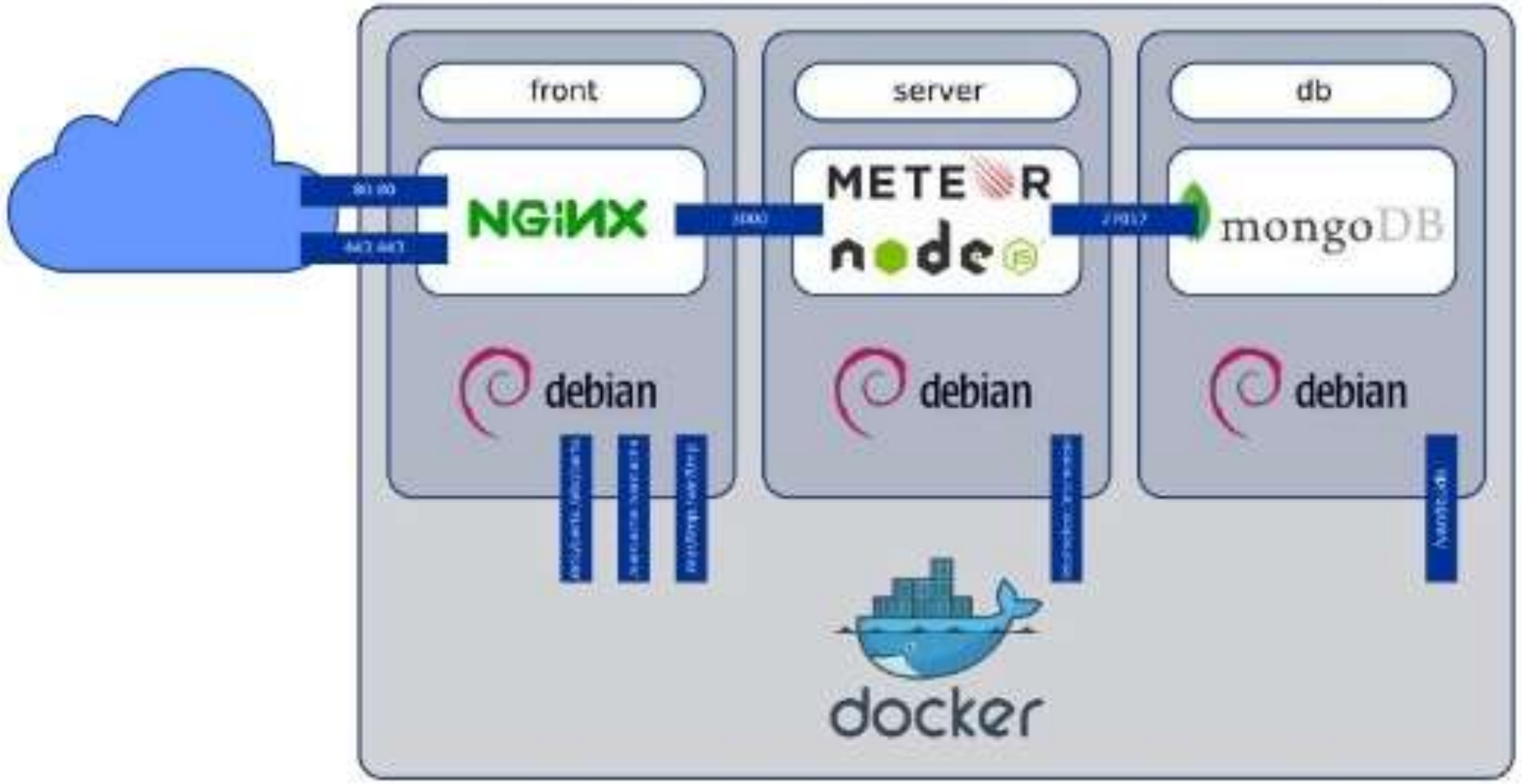
- A multi-layered monolith application
- A distributed application / SOA
- Micro-services
- Containers do support all of them!
- Monolith = Single large Container for App

  + DB Container
- Distributed = Container for each Service / Module

A typical TWO TIER Application

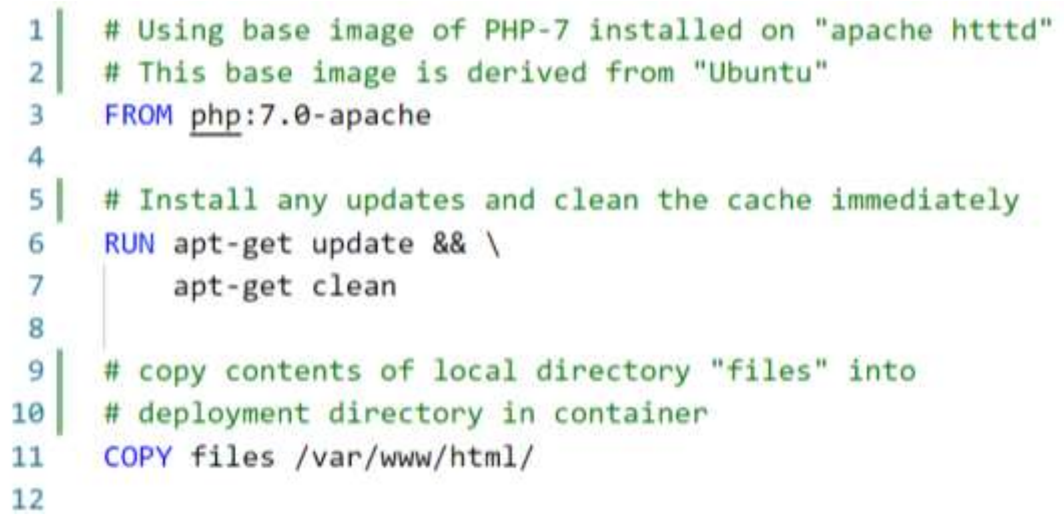


Life after Docker



Dockerfile

- Automates the “Build” process of a container.
- Integrates with all popular CI/CD tools.
- Simple language to define entire build workflow
- Basic system administration commands needed.
  - Windows : prefer powershell
  - Linux: prefer bash
- Can be placed in SCM



```dockerfile
# Using base image of PHP-7 installed on "apache htttd"
# This base image is derived from "Ubuntu"
FROM php:7.0-apache

# Install any updates and clean the cache immediately
RUN apt-get update && \
    apt-get clean

# copy contents of local directory "files" into
# deployment directory in container
COPY files /var/www/html/

```

Dockerfile reference

Command	Description
FROM	Initializes a new build and define "base" image to be used for building.
RUN	A Step to be executed in build process. Can be used for installing necessary package.
CMD	Provide DEFAULT "startup" action for container. Must be used only ONCE.
LABEL	Provides additional META-DATA like Author name
EXPOSE	A Hint for port used by container, can be overridden at run-time.
ENV	Define environment variables.
COPY	Copy files from host system to container. NOTE : optional feature --chown for linux containers

Module 05

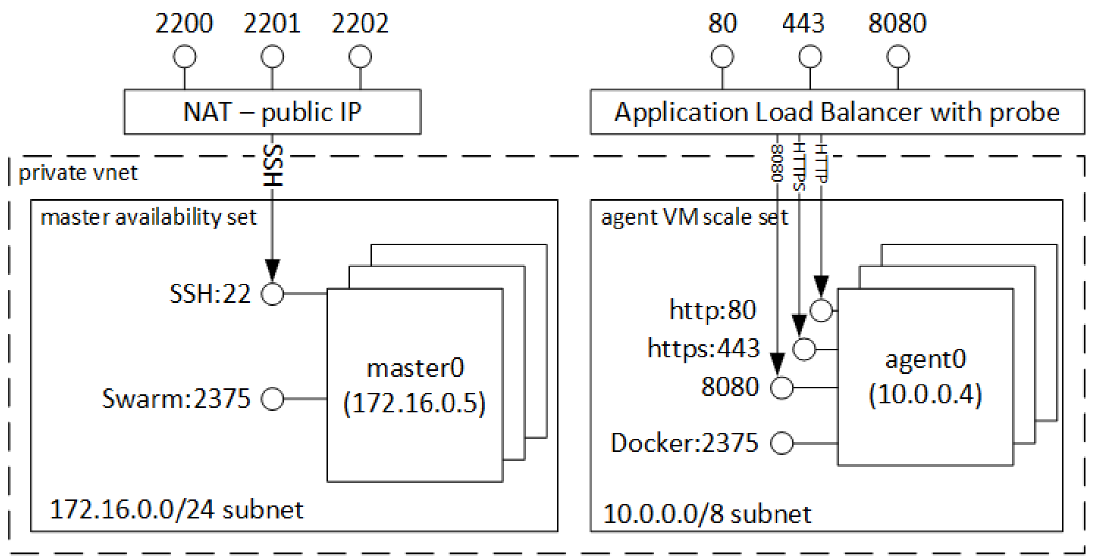
Understanding Orchestration

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Orchestration

- Need for Orchestration
  - Ephemeral computing
  - Desired State
  - Cluster management
- Container Orchestration platforms
  - Docker Swarm (from Docker Inc)
  - Kubernetes (from Google)
  - DC/OS (based on Apache Mesos)
  - Nomad (from HashiCorp)

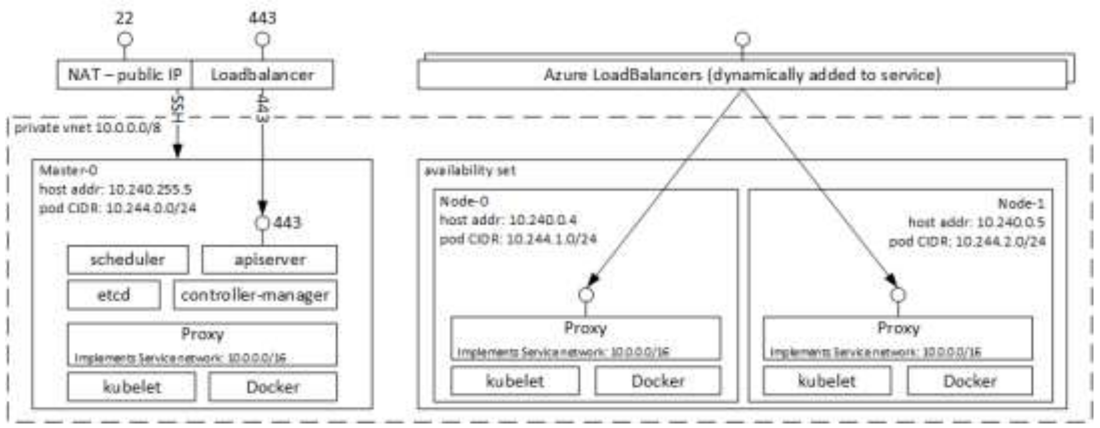
Docker Swarm



- Cluster management *integrated with Docker* engine
- Decentralized design (for cluster management)
- Declarative service model (ref: docker-compose)
- Scaling
- Desired state reconciliation
- Multi-host networking
- Service discovery
- Load balancing

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

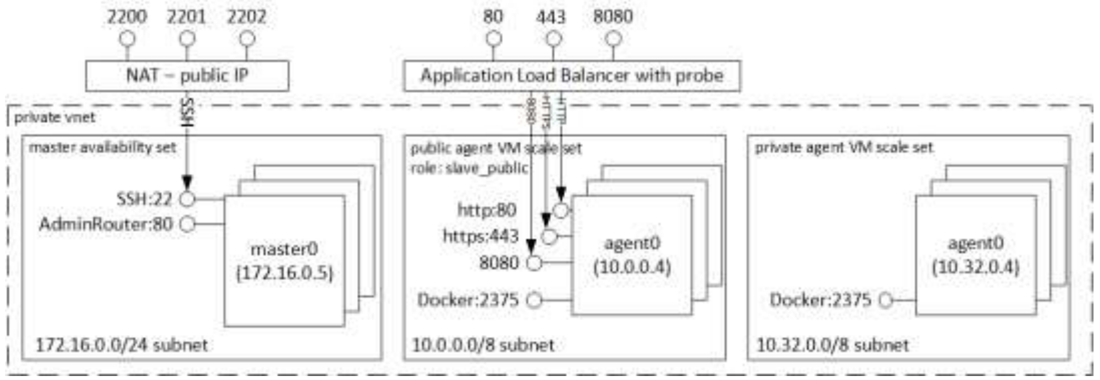
Kubernetes



- Cluster managed from google with 10+ years of internal use by google.
- Open source, run anywhere
- Service discovery
- Load balancing
- Self healing
- Storage orchestration
- Scaling

DC/OS

- Distributed OS based on Apache Mesos
- Treat cluster as a single machine
- High resource utilization
- Container orchestration using “Marathon”
- Cloud agnostic installation



SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Q/A

SYNERGETICS GET IT RIGHT | Microsoft Partner Gold Cloud Platform Silver Learning

Kubernetes

Kubernetes : Introduction

- ***"Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications."***

- One of the most popular cluster management tool.
- Having Largest developer community.
- Architecture designed by Google.
- Backed by 10 years of container deployment experience

Why Kubernetes

Manage your Application, ignore the machines.

Manage Applications

- Where to run
- When to Get, when to run, and when to discard

Application Image usage

- Image lifecycle management
- Run image as application (containers)

What Kubernetes provide?

High Level workload abstractions

If container resembles **atom**, kubernetes provide **molecules**

Storage

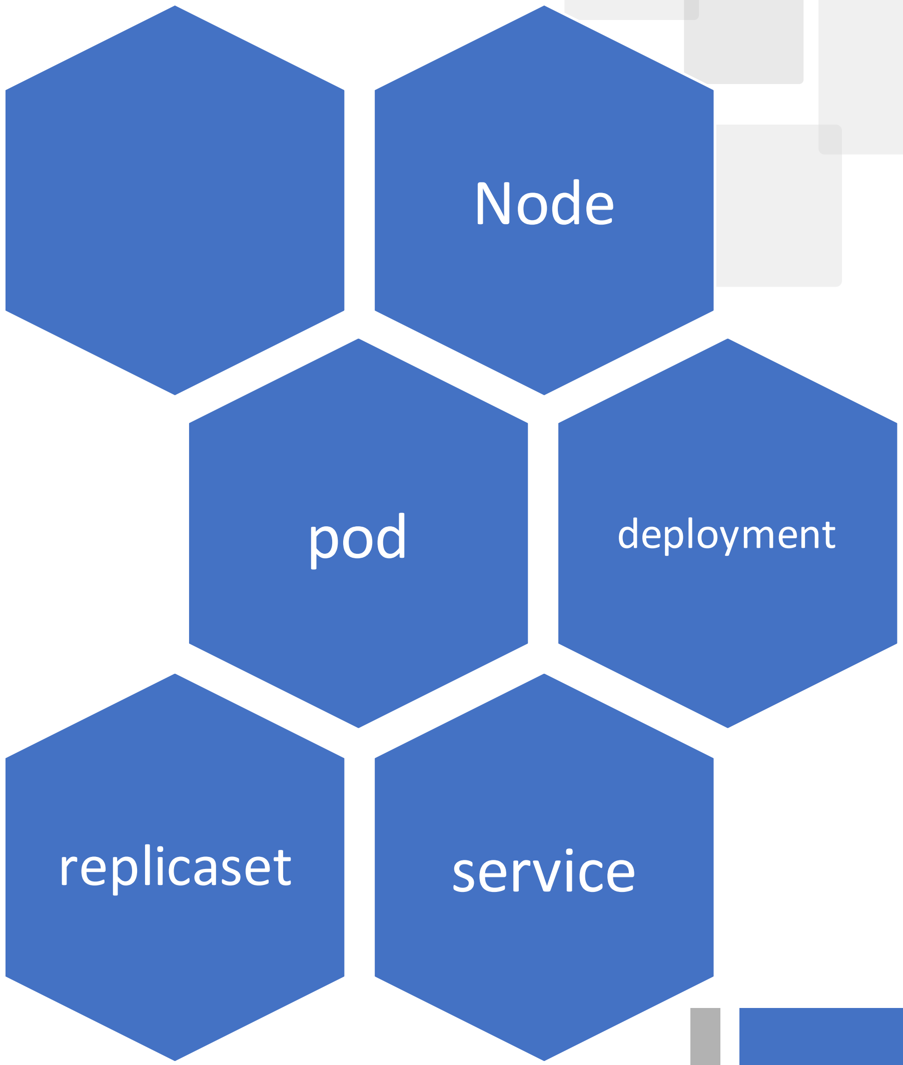
Network

Monitoring

Scaling

Communication

Basic Terminology (Clusters)



Node

A Node is a host machine (physical or virtual) where containerized processes run.

Node activity is managed via one or more master instances.

Node

Designed to exists on multiple machines (Distributed System)

High availability

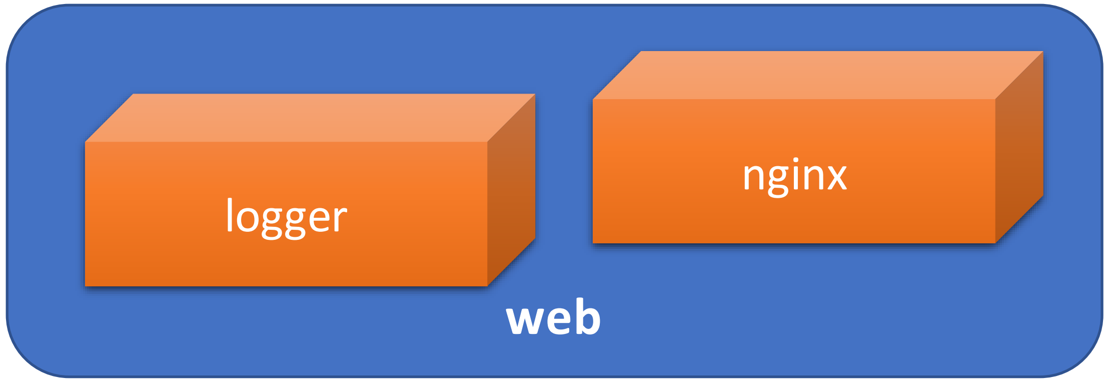
Built in Scalability

Kubernetes API supports both YAML and JSON syntaxes

Pod

A group of one or more co-located containers. Pods represent your minimum increment of scale.

Pods Scale together and they fail together.



Pod

Pods are scheduled to be run on nodes

Asynchronous request processing

Declarative specifications

Automatic health checks, lifecycle management for containers,

Service

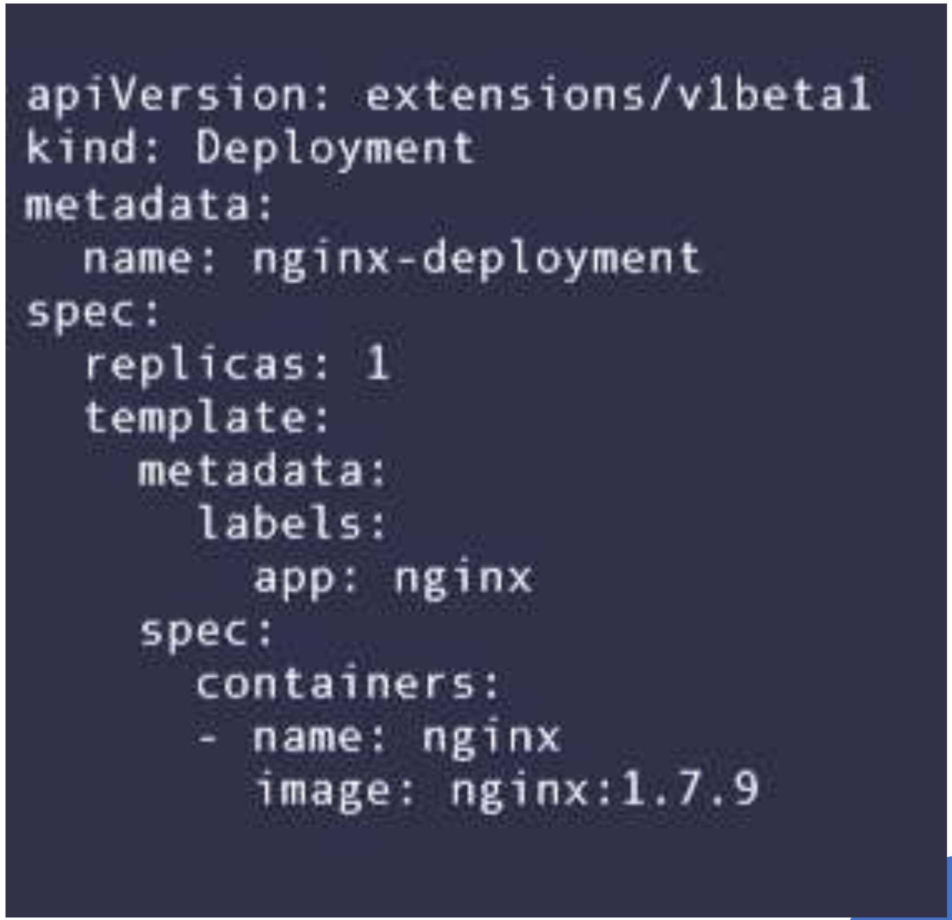
Services establish a single endpoint for collection of replicated pods, distributing inbound traffic based on label selectors.

In kubernetes modelling language they represent a “Load Balancer”.

Pods and Services exists independently, have disjoint lifecycle.

Deployment

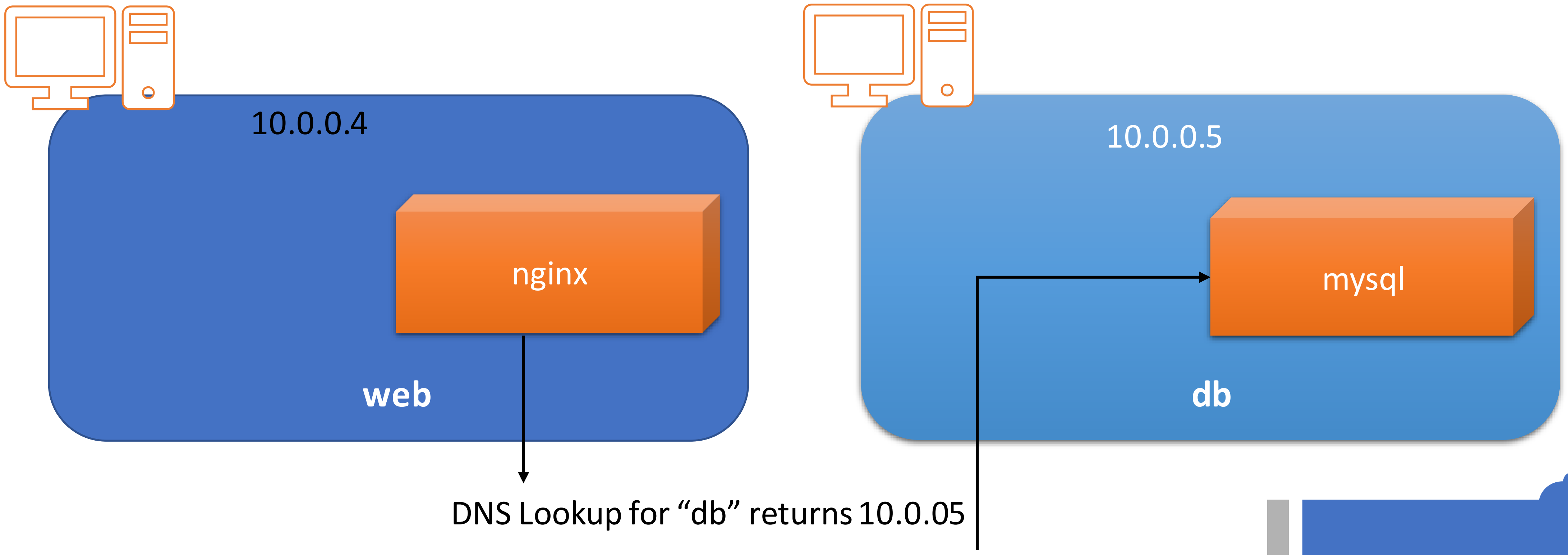
- A Deployment helps you specify container runtime requirements (pods)



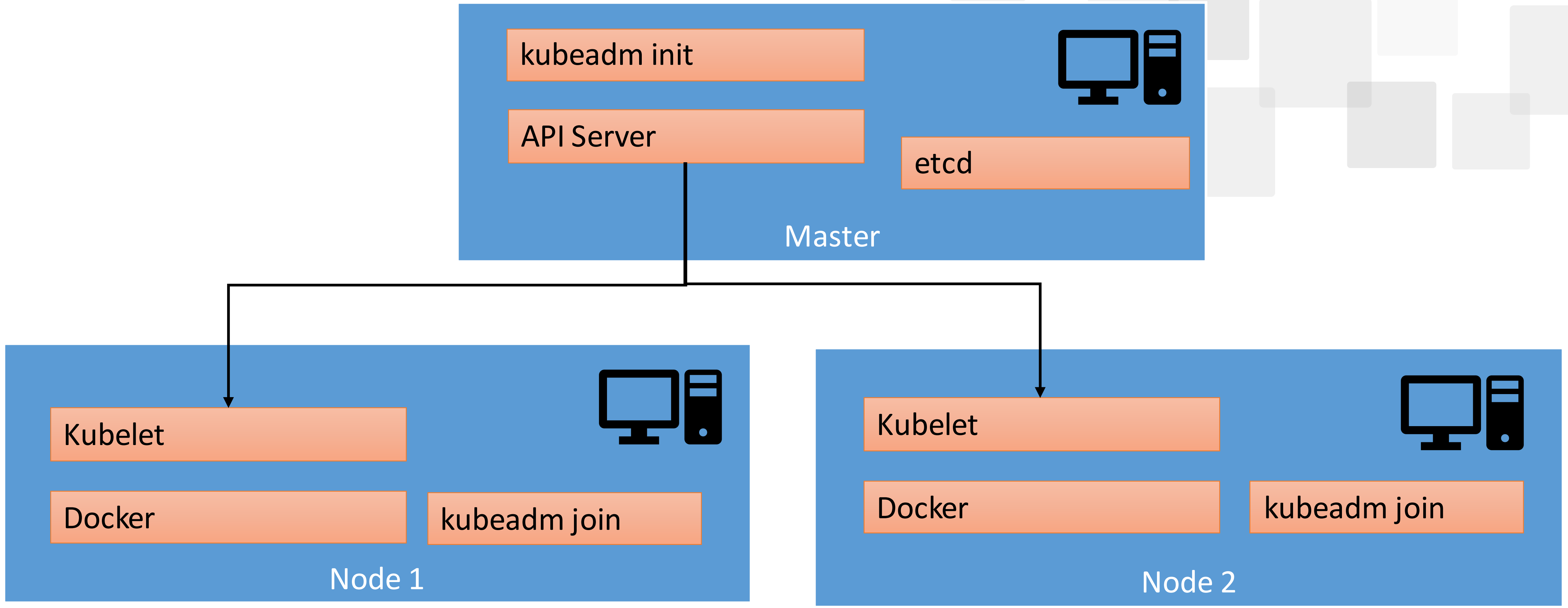
```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
```

Service Discovery

- Kubernetes provide DNS for service discovery



Kubernetes Architecture



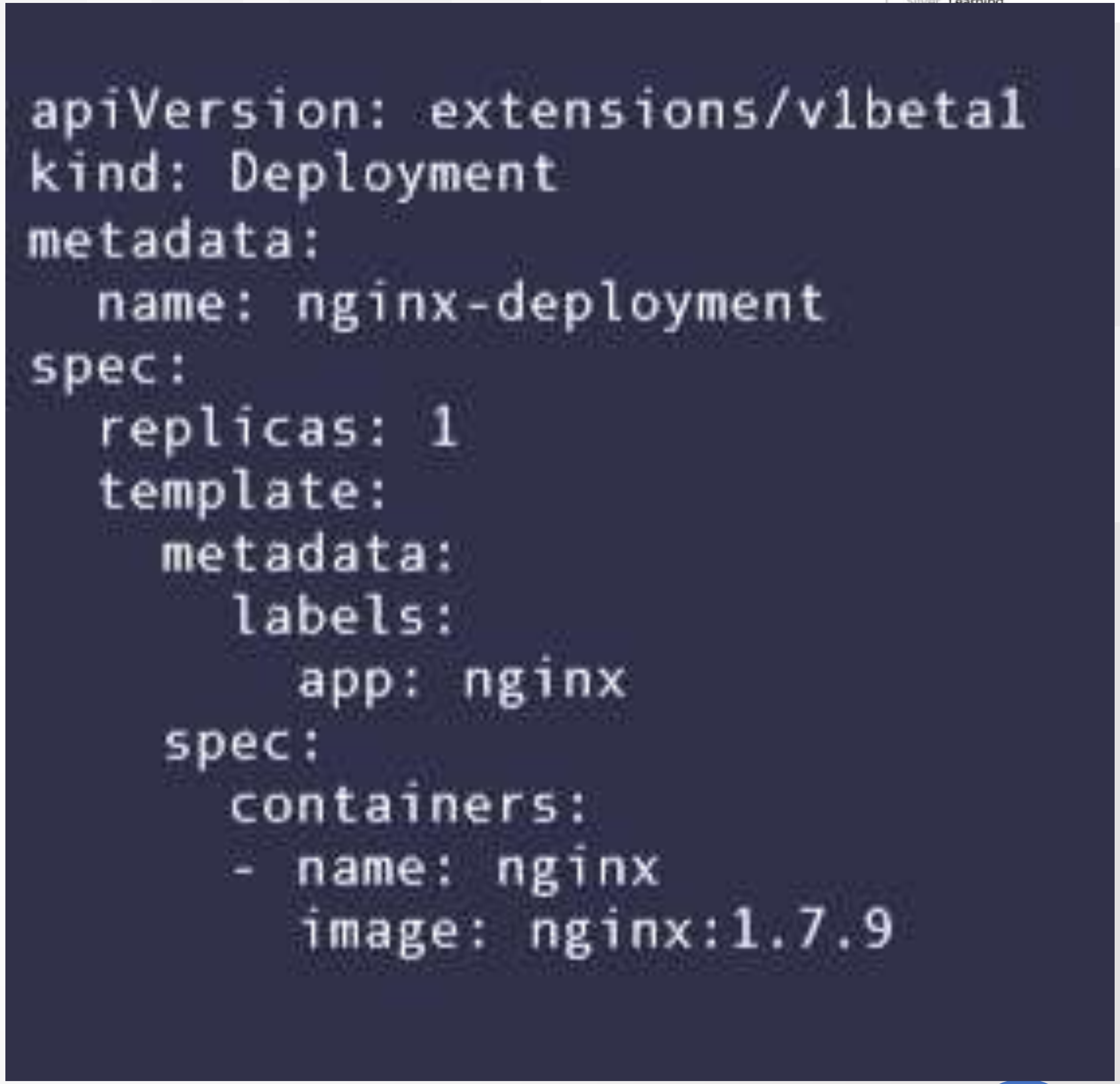
Kubernetes Architecture: API Server

- Every piece of information in Kubernetes is an API object (a.k.a Cluster metadata)
  - Node, Pod, Job, Deployment, Endpoint, Service, Event, PersistentVolume, ResourceQuota, Secrets, ConfigMap, ……
  - Schema and data model explicitly defined and enforced
  - Every API object has a UID and version number

- Desired state in cluster is achieved by collaborations of separate autonomous entities reacting on changes of one or more API object(s) they are interested in

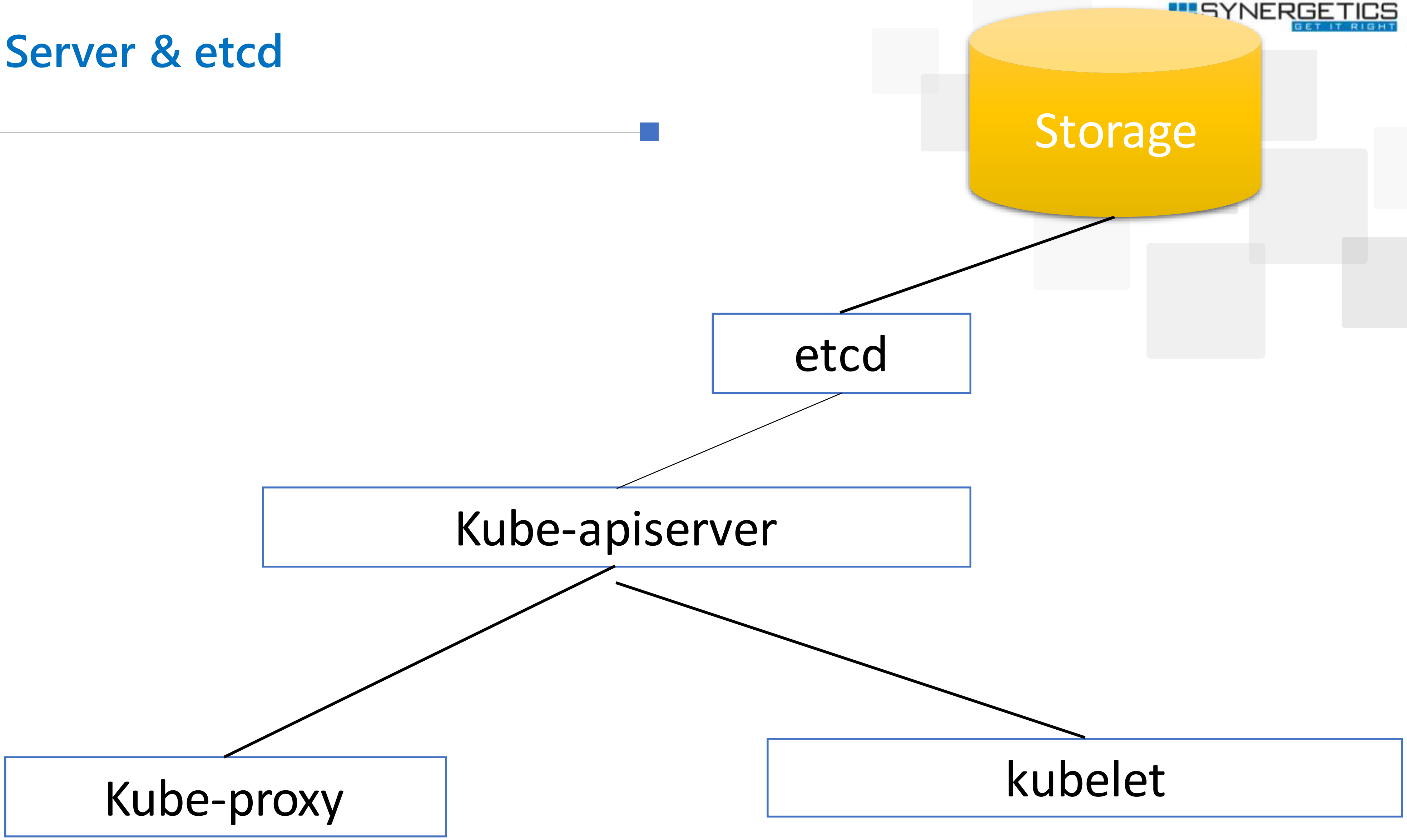
Kubernetes REST API

- The API Object Definition
  - Major fields: apiVersion, kind, metadatam spec and status
  - Metadata includes name, namespace, label, annotation etc
  - Spec describes desired state, while status describes current state



```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
```

API Server & etcd



Kube-scheduler

Sequential Scheduler

Schedule 30,000 pods on 1,000 nodes within 587 seconds (version 1.2)

High Availability via leader election.

Kubelets can reject pods in case of conflict, and trigger reschedule.

Kubelet

Node’s worker

Ensure Pods run as described by spec

Report pod and node readiness

Perfomance Node level admission control

Reserve resource, limit # of pods etc

Report node status to API Server

Machine info

Resource usage

Health

Images

Not a container but a service under system

Kube proxy

Daemon on every node

Watch on changes of service and endpoints

Manipulate iptables rules on host to achieve service load balancing