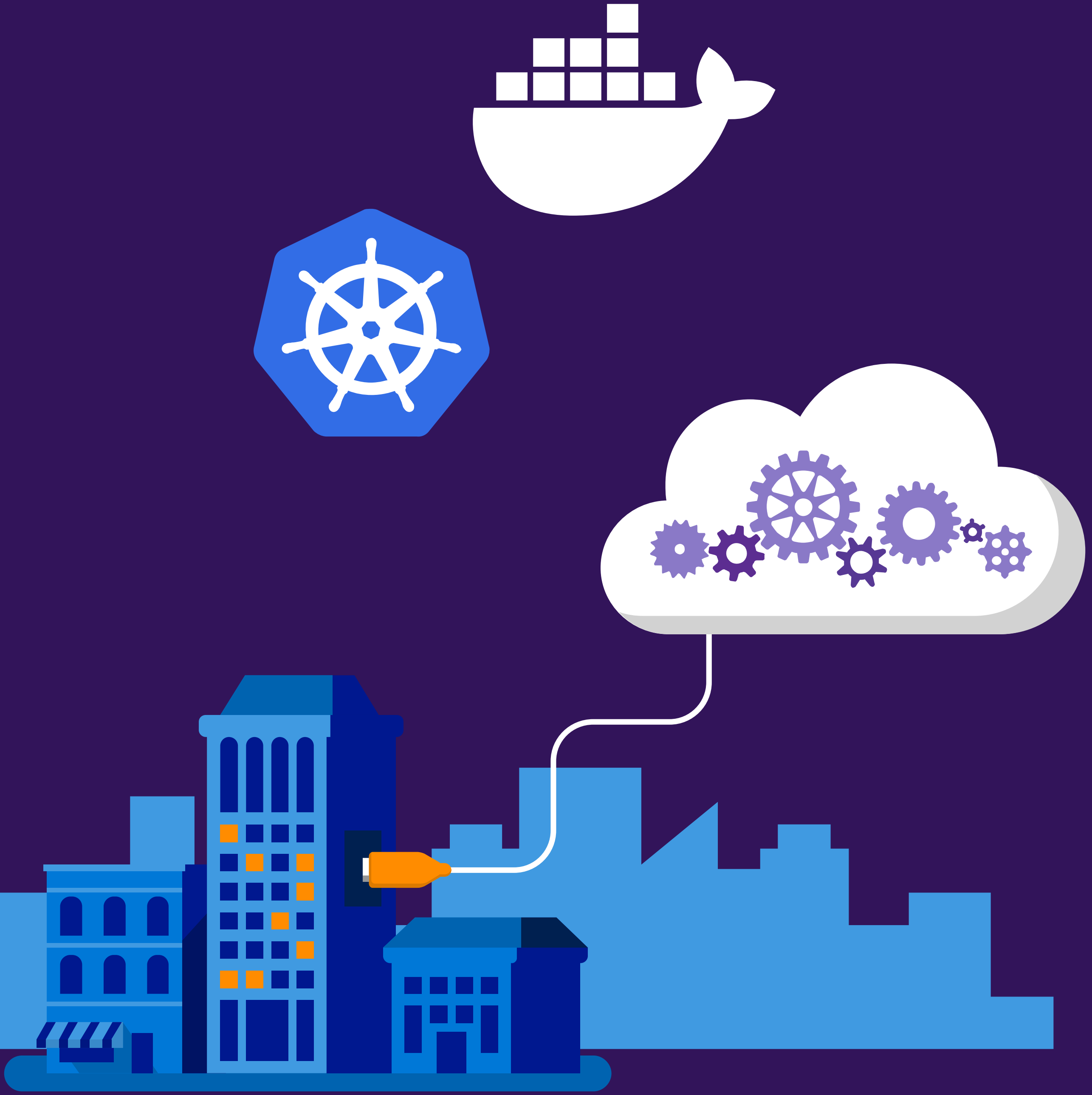
Container Orchestration

Kubernetes and Azure Container Service

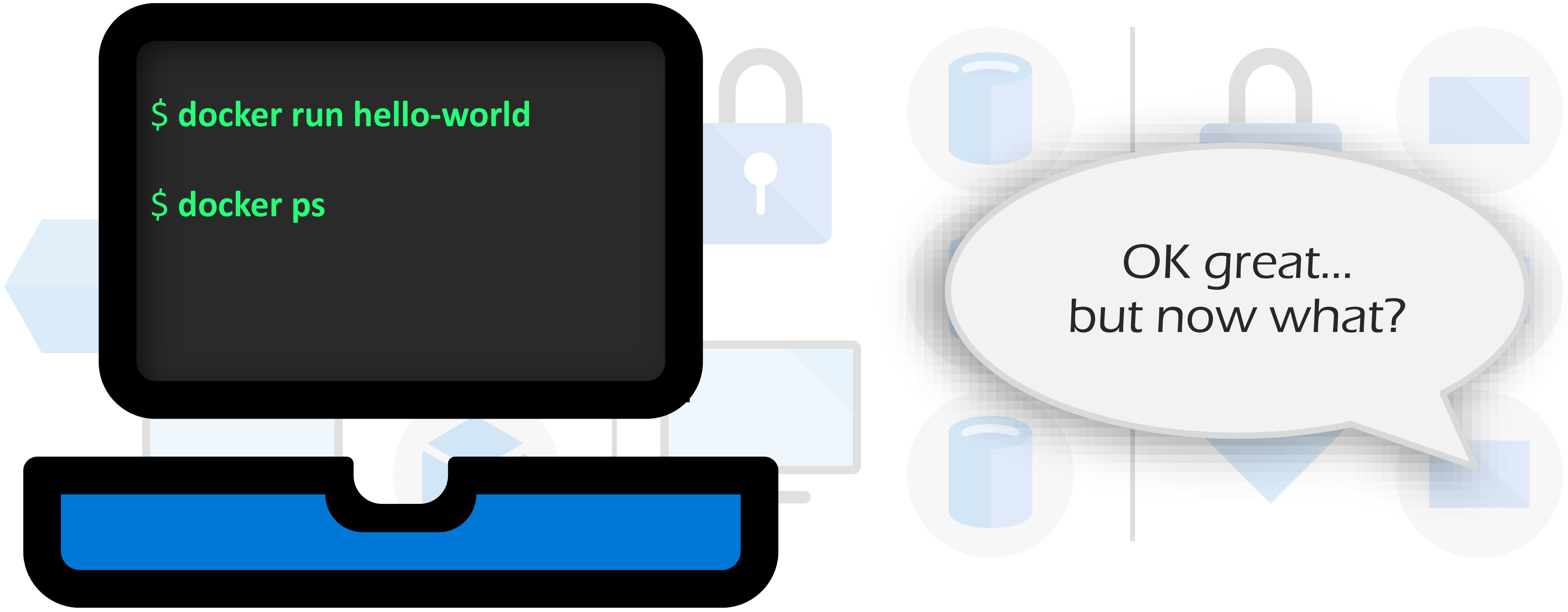


Ben Coleman
Cloud Architect & Evangelist
@BenCodeGeek

Microsoft

Need For Orchestration

- Docker is easy to get working on a **single host** or your **dev machine**



```
$ docker run hello-world

$ docker ps
```

Need For Orchestration

- Docker is easy to get working on a **single host** or your **dev machine**
- How do we manage containers across **multiple hosts**
- Complexity rapidly arises…

NAME RESOLUTION

MANGING PLACEMENT

RESOURCE MANAGEMENT

SECRETS

NETWORKING & LOAD BALANCING

SCHEDULING

SECURITY

SCALING UP/DOWN

HEALTH CHECKS

FAILOVER

SERVICE DISCOVERY

Container orchestrators solve these problems

Container Orchestrator Tools & Projects

Kubernetes

Mesosphere DC/OS

Docker Swarm

CloudFoundry

Service Fabric

Kubernetes

Production-Grade Container Orchestration
Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

- Optimize workload placement (aka ‘binpacking’)
- Horizontal scaling
- Service discovery and load balancing
- Automated rollouts and rollbacks
- Secret and configuration management
- Storage orchestration
- Batch execution
- Self-healing

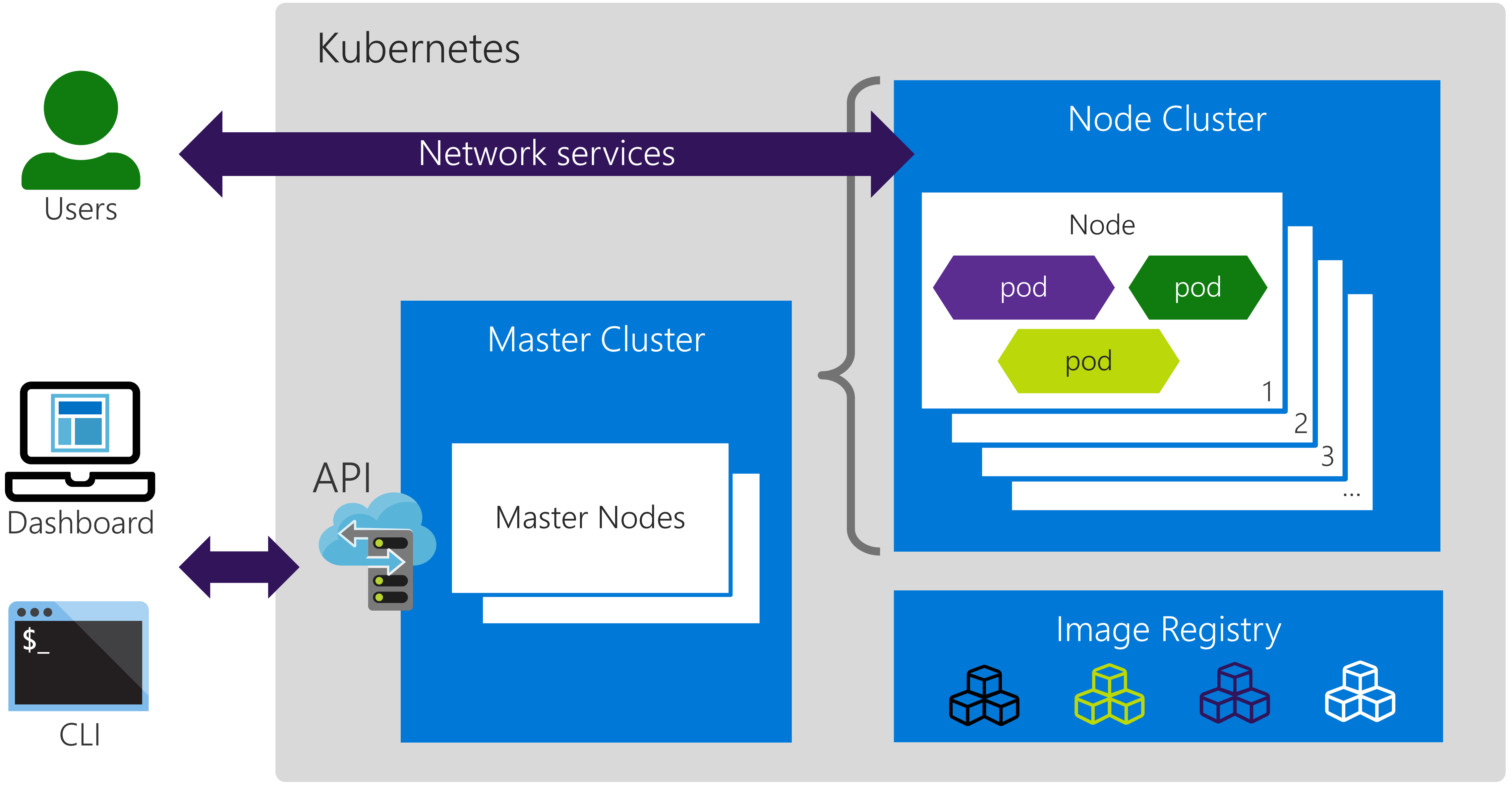
CLOUD NATIVE
COMPUTING FOUNDATION

Kubernetes - Growth in 2017

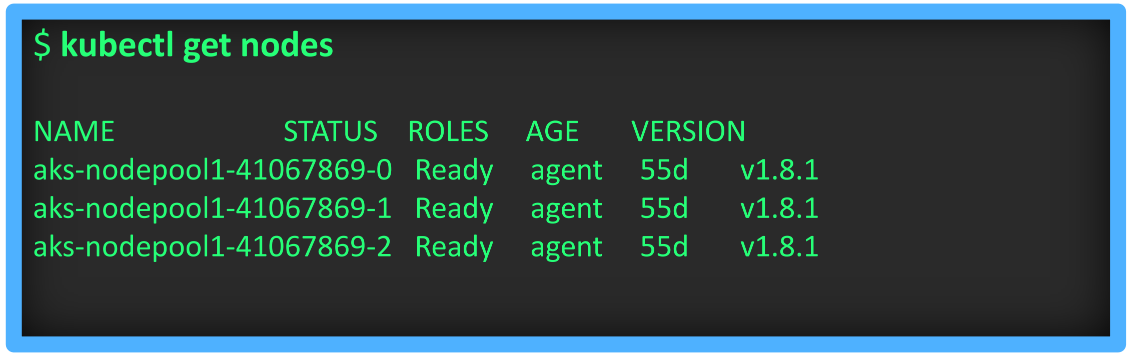
Feb 2017	- CoreOS replaces their Fleet product with Kubernetes
July 2017	- Microsoft joins Cloud Native Computing Foundation (CNCF)
Sept 2017	- Oracle joins CNCF
Oct 2017	- Docker Enterprise announces support for Kubernetes
Oct 2017	- Azure launch Container Service managed Kubernetes (AKS)
Nov 2017	- CNCF & 36 companies agree on Kubernetes standard
Nov 2017	- AWS launch Kubernetes service

Now widely considered the defacto solution for container orchestration

Kubernetes (Very!) Simplified Architecture



Interacting with Kubernetes

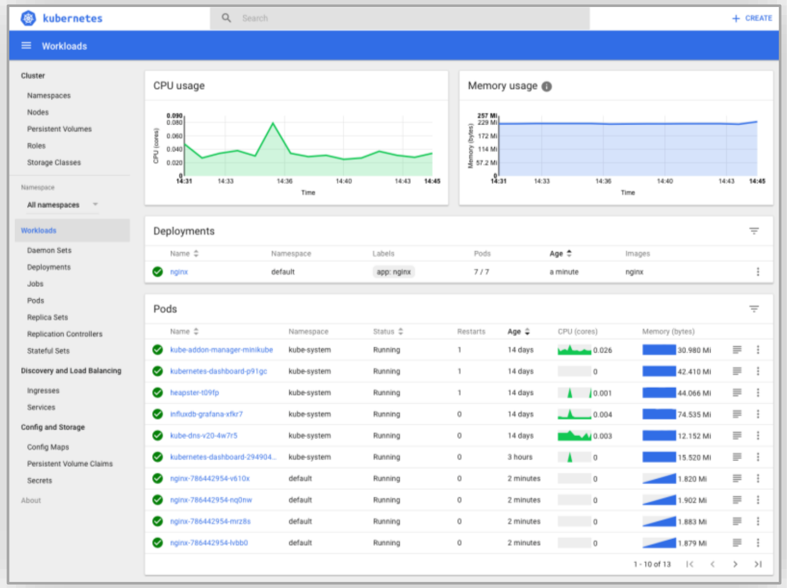


```
$ kubectl get nodes

NAME                      STATUS   ROLES   AGE   VERSION
aks-nodepool1-41067869-0  Ready    agent   55d   v1.8.1
aks-nodepool1-41067869-1  Ready    agent   55d   v1.8.1
aks-nodepool1-41067869-2  Ready    agent   55d   v1.8.1
```

Command Line: **kubectl**

REST API



Deployments

Name	Namespace	Labels	Pods	Age	Images
nginx	default	app: nginx	7 / 7	a minute	nginx

Pods

Name	Namespace	Status	Restarts	Age	CPU (cores)	Memory (bytes)
kube-addon-manager-minikube	kube-system	Running	1	14 days	0.026	30.980 Mi
kubernetes-dashboard-p91gc	kube-system	Running	1	14 days	0	42.410 Mi
heapster-t09fp	kube-system	Running	1	14 days	0.001	44.066 Mi
influxdb-grafana-xfkr7	kube-system	Running	0	14 days	0.004	74.535 Mi
kube-dns-v20-4w7r5	kube-system	Running	0	14 days	0.003	12.152 Mi
kubernetes-dashboard-294904...	kube-system	Running	0	3 hours	0	15.520 Mi
nginx-786442954-v610x	default	Running	0	2 minutes	0	1.820 Mi
nginx-786442954-nq0nw	default	Running	0	2 minutes	0	1.902 Mi
nginx-786442954-mrz8s	default	Running	0	2 minutes	0	1.883 Mi
nginx-786442954-lvbb0	default	Running	0	2 minutes	0	1.879 Mi

1 - 10 of 13

Dashboard

Kubernetes Concepts and Terms

Node
A worker machine (VM) normally clustered, each capable of running pods

Deployment
A logical object for managing a replicated application (i.e. set of pods)

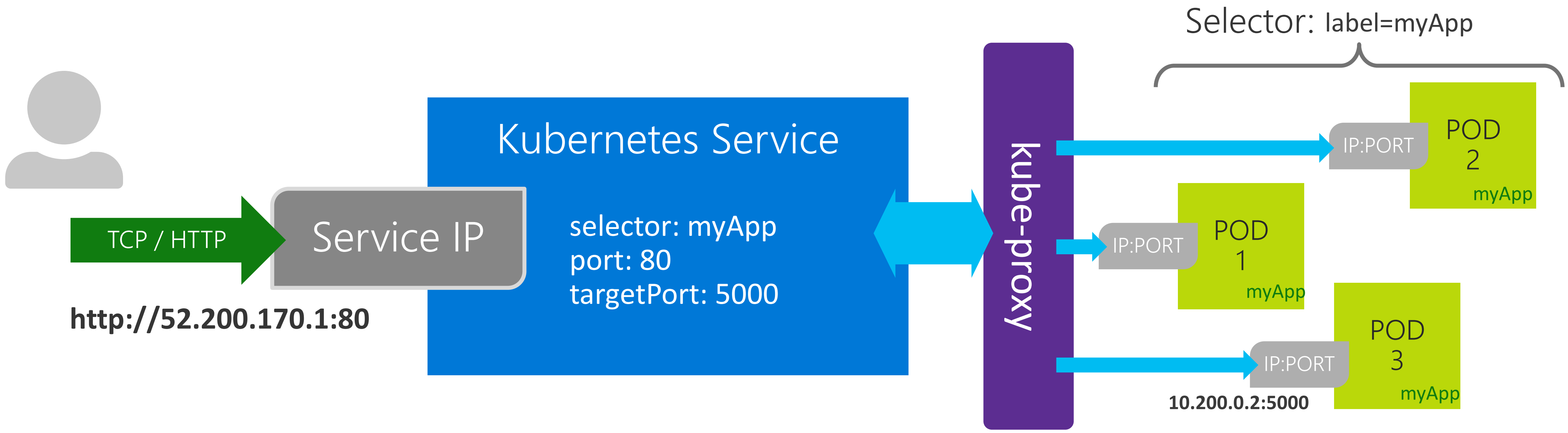
Label
Metadata attached to any object for configuration and selection

Pod
A group of one or more running containers that is managed through a lifecycle

Service
Network access to a resource, e.g. pod or port. Typically load balanced

Replica Set
A set of one or more pods that is distributed and managed across Nodes

Kubernetes Networking (Simplified!)



- LoadBalancer - Uses cloud provider to present load balanced service (e.g. Azure Load Balancer)
- ClusterIP - Internal cluster virtual IP
- NodePort - Map a range of ports
- ExternalName - DNS CNAME redirection

Kubernetes Deployments

- Described in YAML or JSON
- Define Kubernetes objects; e.g. deployment, pod, replica-set, service, etc.
- Tied closely to the API (changes with Kubernetes version)
- Deploy using CLI or dashboard
- Similar to Docker Compose

```yaml
apiVersion: apps/v1beta2   # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
 name: dotnetcore-deployment
spec:
 selector:
  matchLabels:
   app: dotnet-app
 replicas: 2   # tells deployment to run 2 pods matching the template
 template:     # create pods using pod definition in this template
  metadata:
   # A unique name for pod is generated from the deployment name
   labels:
    app: dotnet-app
  spec:
   containers:
   - name: dotnet-container
    image: microsoft/aspnetcore:2.0.5
    ports:
    - containerPort: 5000
```

Azure Container Service**s**

"Containers everywhere"

Azure Container Service

Azure Container Instances

Azure Container Registry

Web App for Containers

Docker Machine Driver

Azure Service Fabric

Docker VM Extensions

Azure Batch

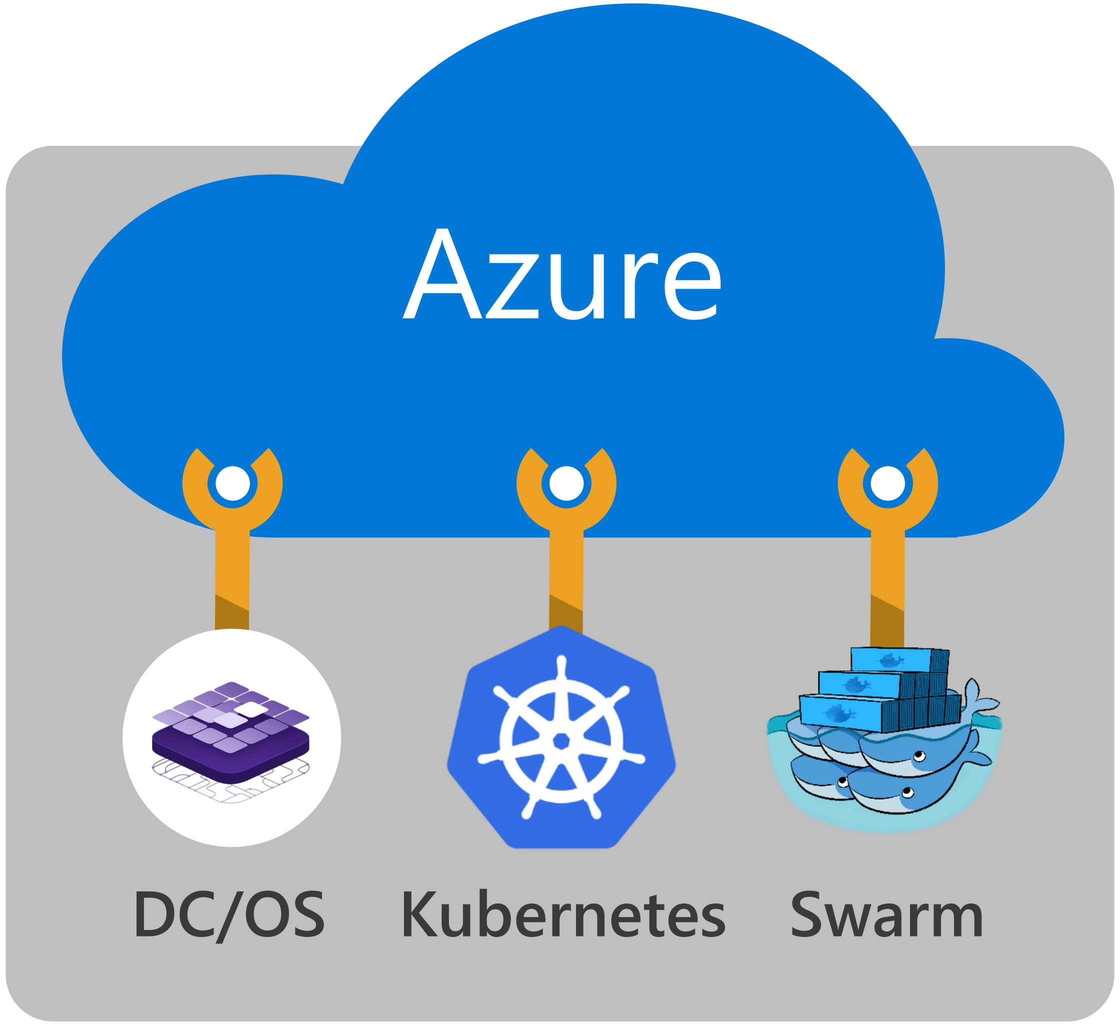
Windows Server 2016

Azure Marketplace

Some Azure Container Service (ACS) History

- Original Azure Container Service - ACS
- Unmanaged clusters

- **Will deprecate to AKS Q1/Q2 CY18**
- DC/OS and Swarm to be available in Azure Marketplace



Azure Container Service (AKS)

PREVIEW

"Kubernetes as a Service"

- Next generation of **Azure Container Service**
- Bringing together best of the Azure core platform and Kubernetes
- Managed Kubernetes clusters
  - Manager nodes - controlled by Microsoft & Azure
  - PaaS "lite"
- Kubernetes 1.7 and 1.8

Kubernetes As A Service
AKS

Azure Container Service (AKS)

PREVIEW

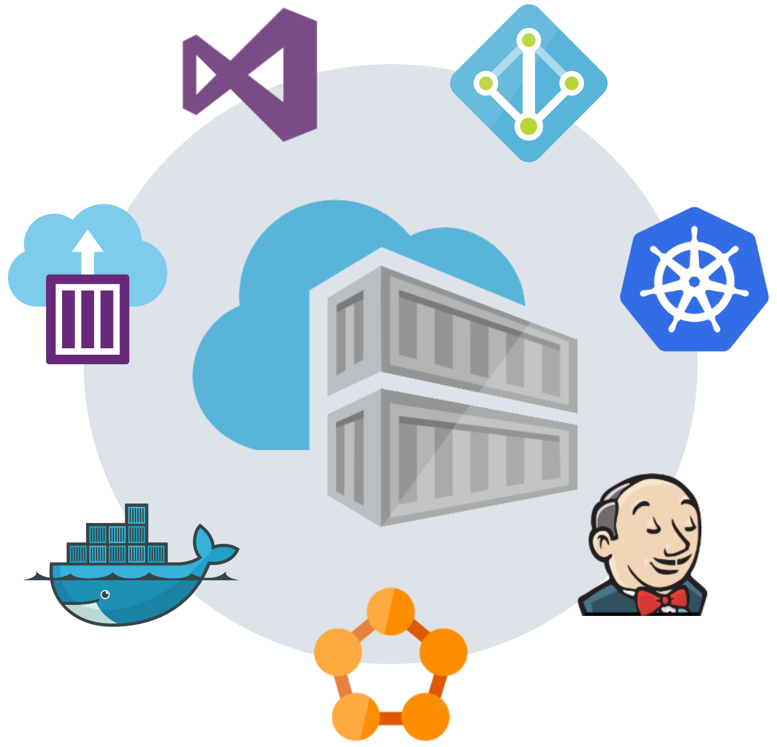
- Standard open-source distribution of Kubernetes
- Scale nodes up & down
- Standard Kubernetes APIs and tooling & dashboard
- Free service
  - Master nodes free of charge & managed for you
  - Pay for compute nodes per normal consumption rates

DEMO INTRO

Azure Container Registry

"Docker Registry as a service"

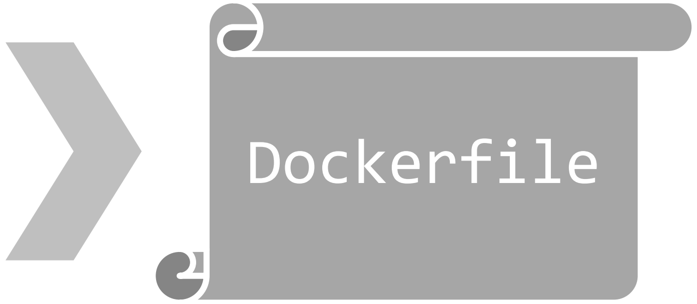
- Secure private Docker v2 registry **as a service**
- Easily shared across Azure & other services
- Use **standard** Docker tools & APIs
- Backed with Azure Active Directory for **access management**
- **Webhooks** for integration & DevOps



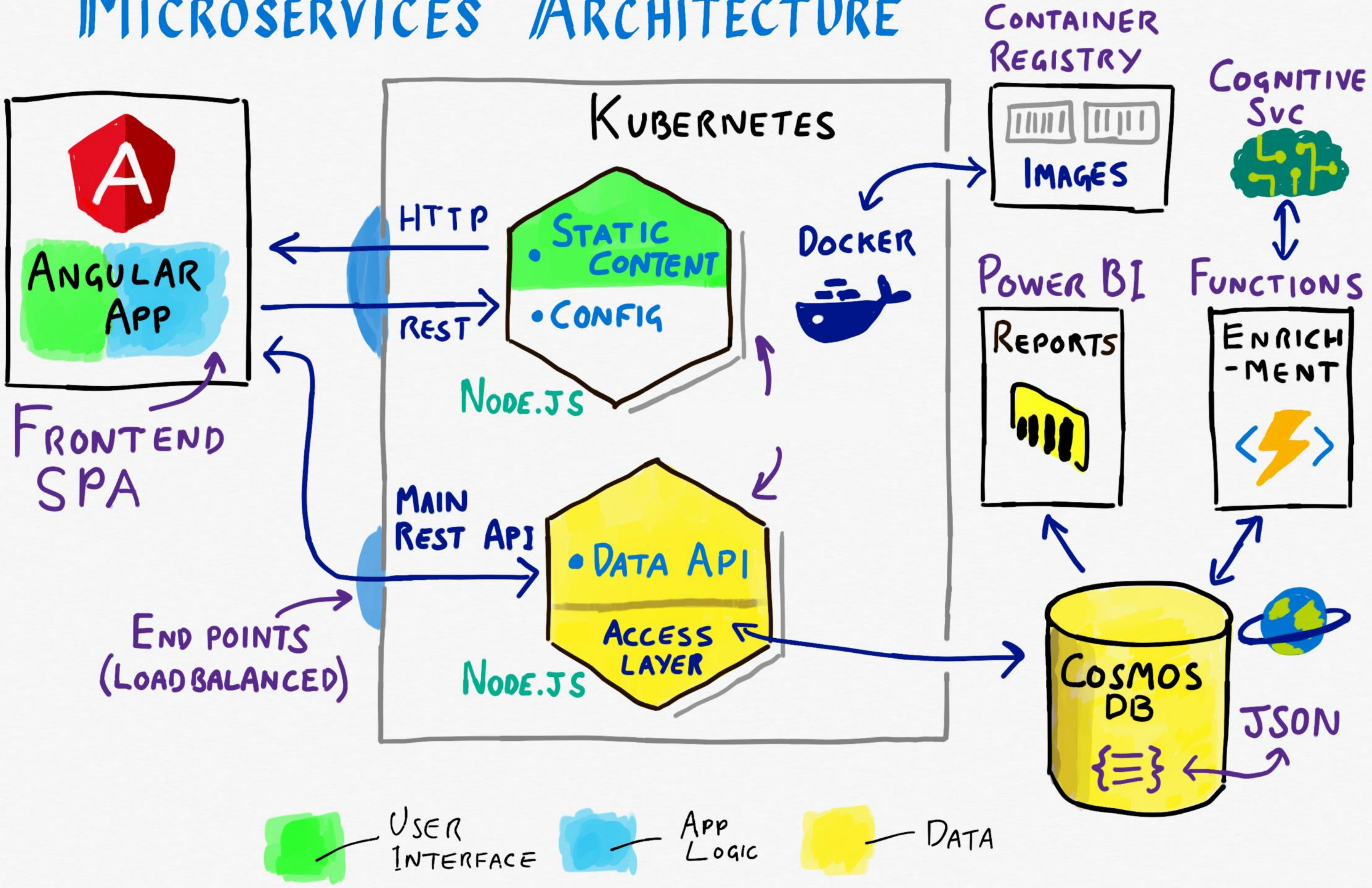
Building Images - Basics

- A file called a **Dockerfile** is used to build images.
  Note. The default name of this file is typically just `Dockerfile` (no extension)
- A **Dockerfile** is simply a set of instructions on how to build the image, much like a script.

  - Laying out the filesystem & directory structure the app is expecting
  - Copying in binaries and configs
  - Installing any pre-req packages, libraries and software
  - Running any custom set-up commands
  - Setting environmental variables
  - Defining the command/executable to start your application



Microservices Architecture



DEMO

END

Dockerfile

```
FROM microsoft/aspnetcore-build:2.0.3
LABEL description="A test docker image"

# Run Dotnet build
WORKDIR /build
COPY src/ .
RUN dotnet restore
RUN dotnet publish --configuration release

# Copy published binaries
WORKDIR /app
RUN cp -R /build/bin/release/netcoreapp2.0/publish/* .

# Kestrel port 5000 and start the app
EXPOSE 5000
CMD ["dotnet", "dotnet-demoapp.dll"]
```