
Visual Studio LIVE!
EXPERT SOLUTIONS FOR .NET DEVELOPERS
San Diego
W07 Docker Containers on Azure – Let Me Count the Ways
Michele Leroux Bustamante
Microservices / Security Architect
Solliance
Intermediate
Code Again for the First Time!
Visual Studio 25 YEARS OF CODING INNOVATION

Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS | San Diego

# W07 Docker Containers on Azure – Let Me Count the Ways


## Michele Leroux Bustamante

Microservices / Security Architect

Solliance

michelebusta@solliance.net

@michelebusta


solliance your solution experts. PolicyServer

MVP Microsoft® Most Valuable Professional

Developing Microsoft Azure Solutions Exam Ref 70-532 SECOND EDITION

Microsoft Regional Director

docker CAPTAIN

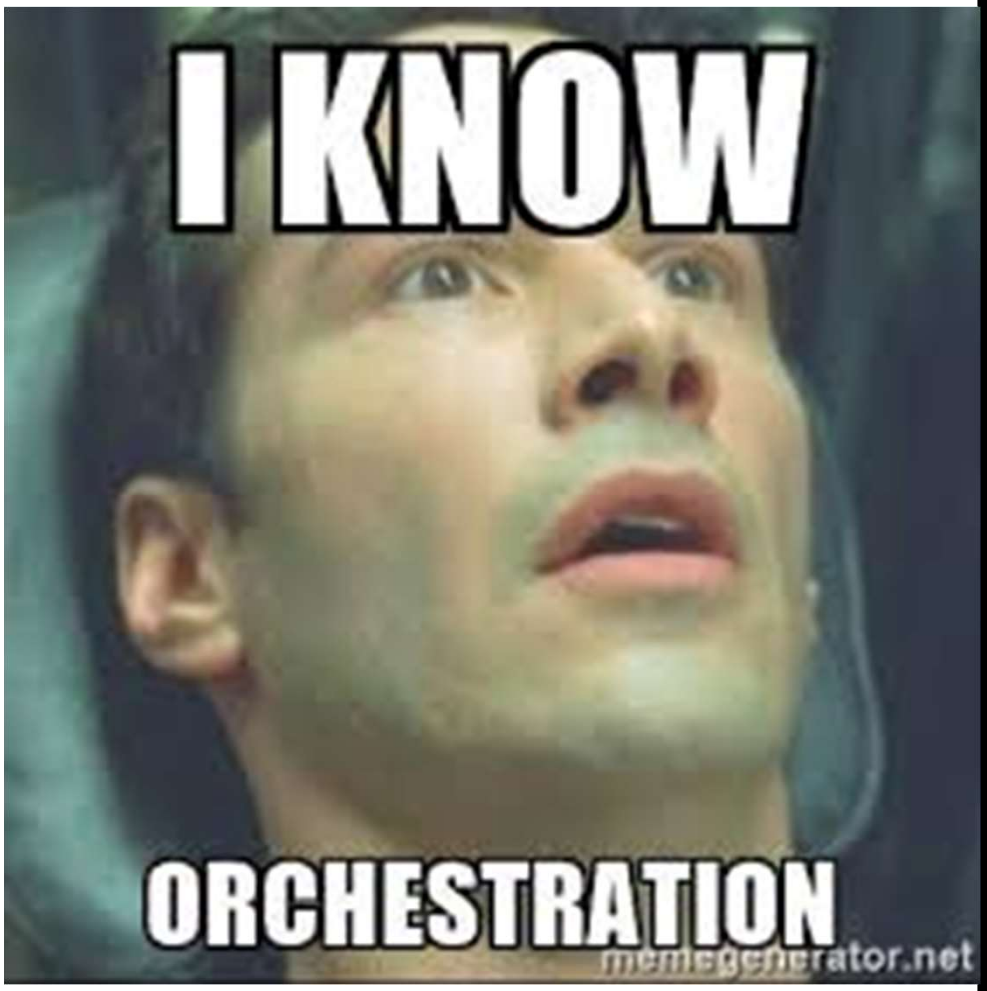
Intermediate

Code Again for the First Time!

Visual Studio 25 YEARS OF CODING INNOVATION

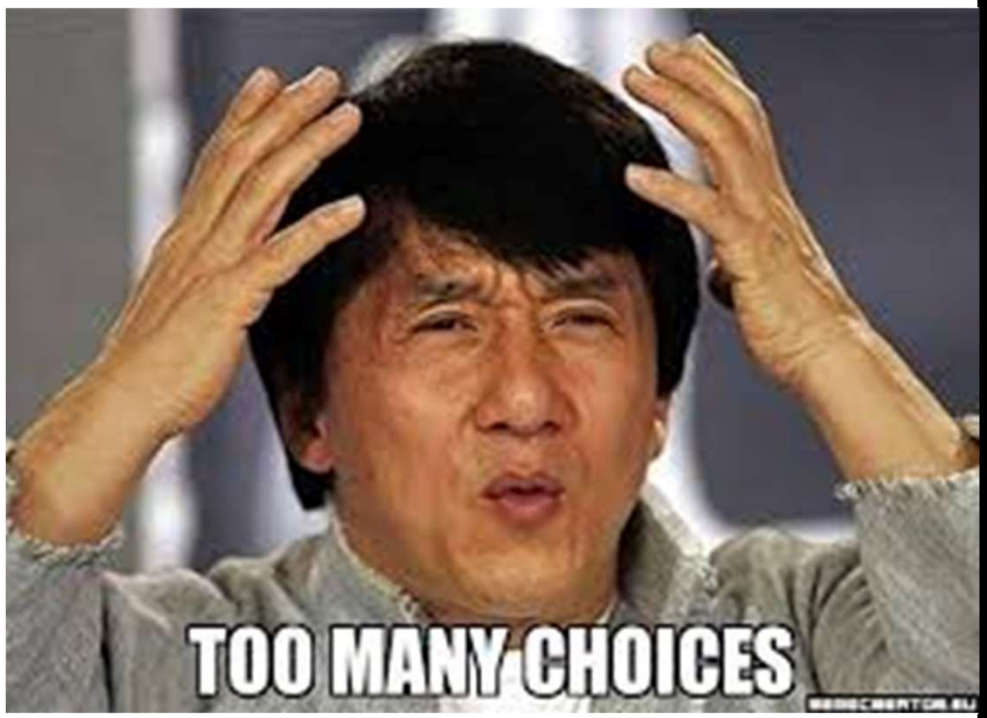
# 2

# Frequently Asked Questions




# #1

## How do I know if I need an orchestration platform?




# #2

## How do I CHOOSE the right approach or platform?

# Microservices Compute on Azure

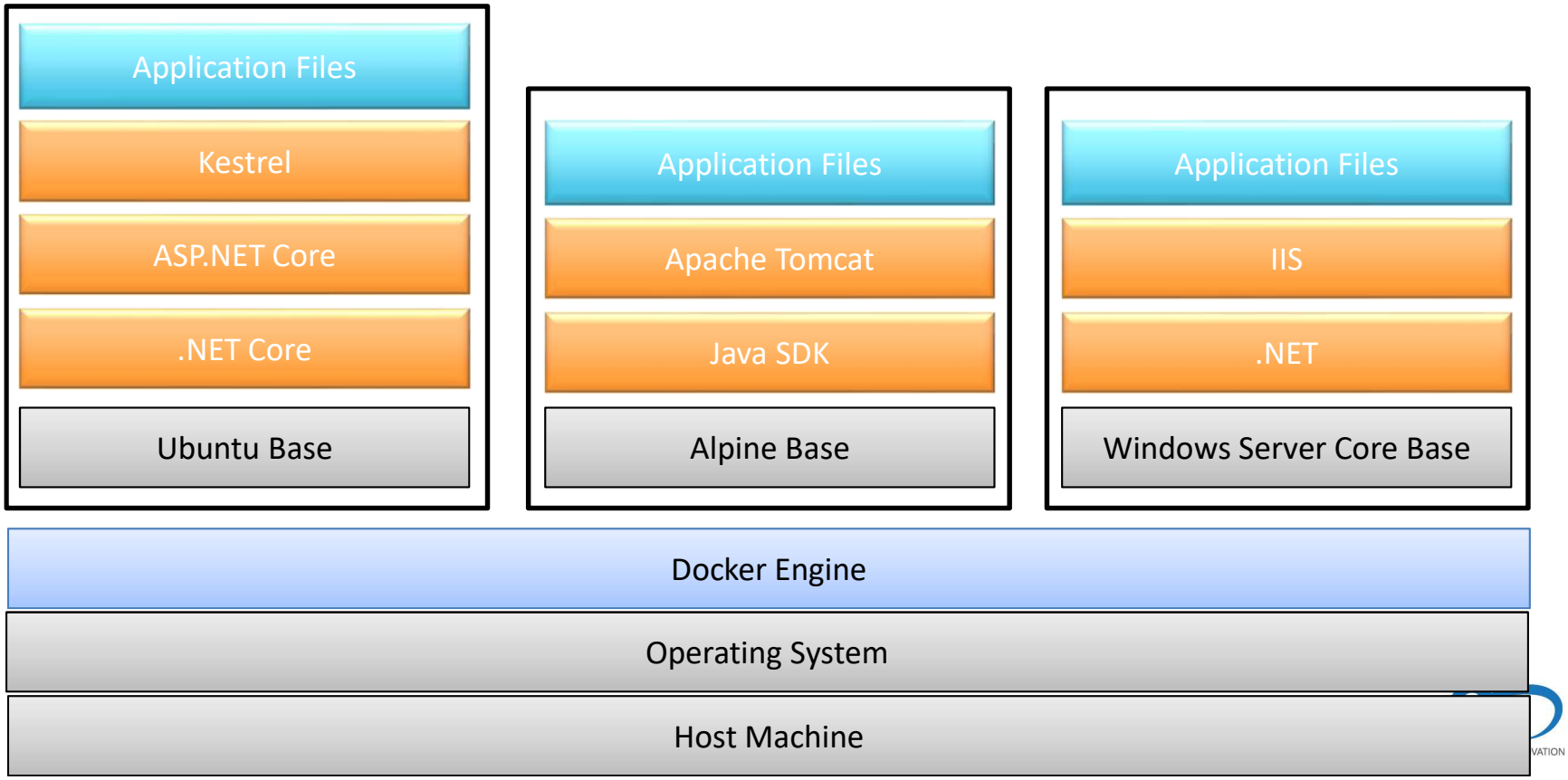
- Serverless
  - Functions
  - Azure Container Instances
- PaaS
  - App Services for Containers
  - Azure Kubernetes Service
  - Azure Service Fabric
- IaaS
  - Deploy any orchestration platform including Docker CE, Docker EE, Mesophere DC/OS, Kubernetes
  - Marketplace templates

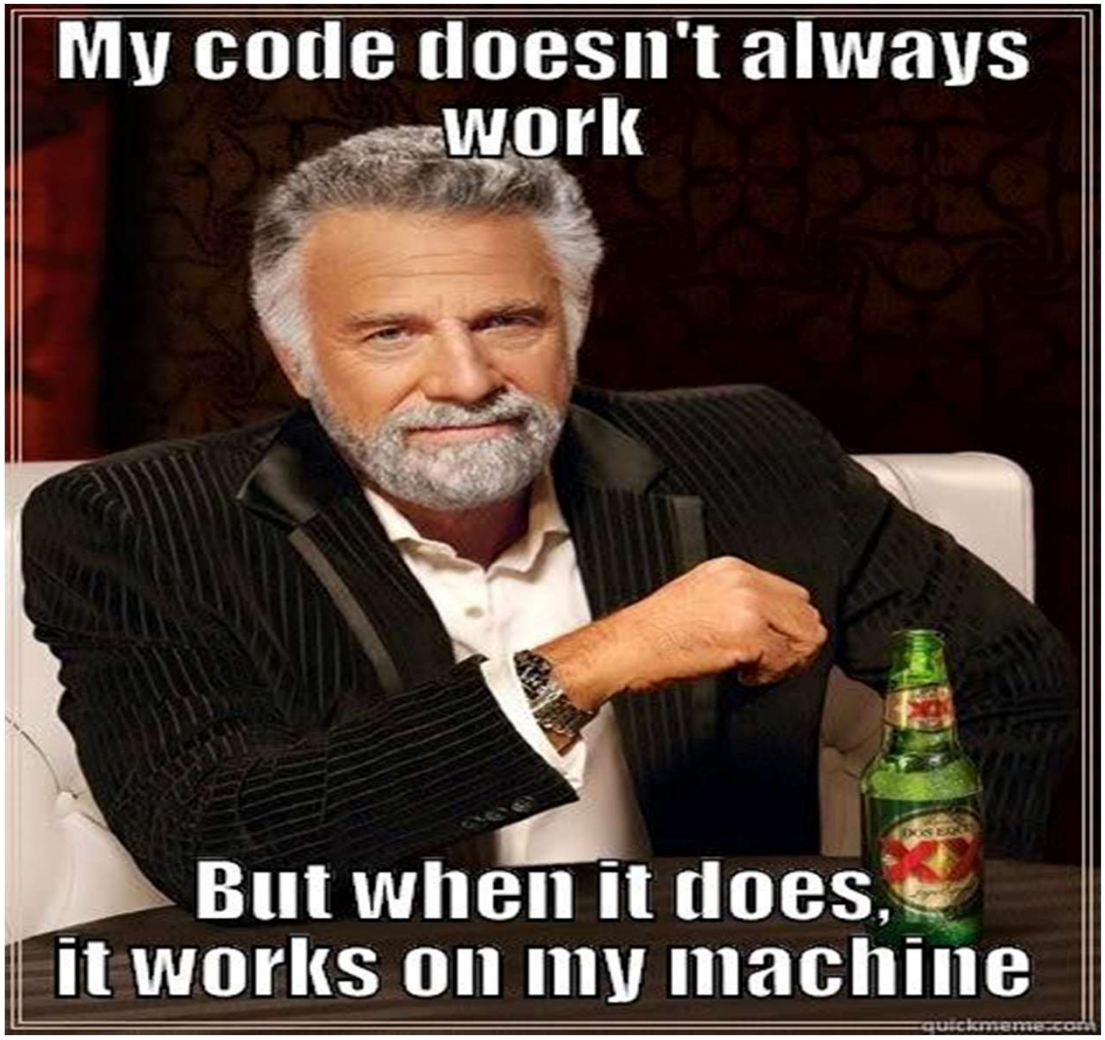
# Docker

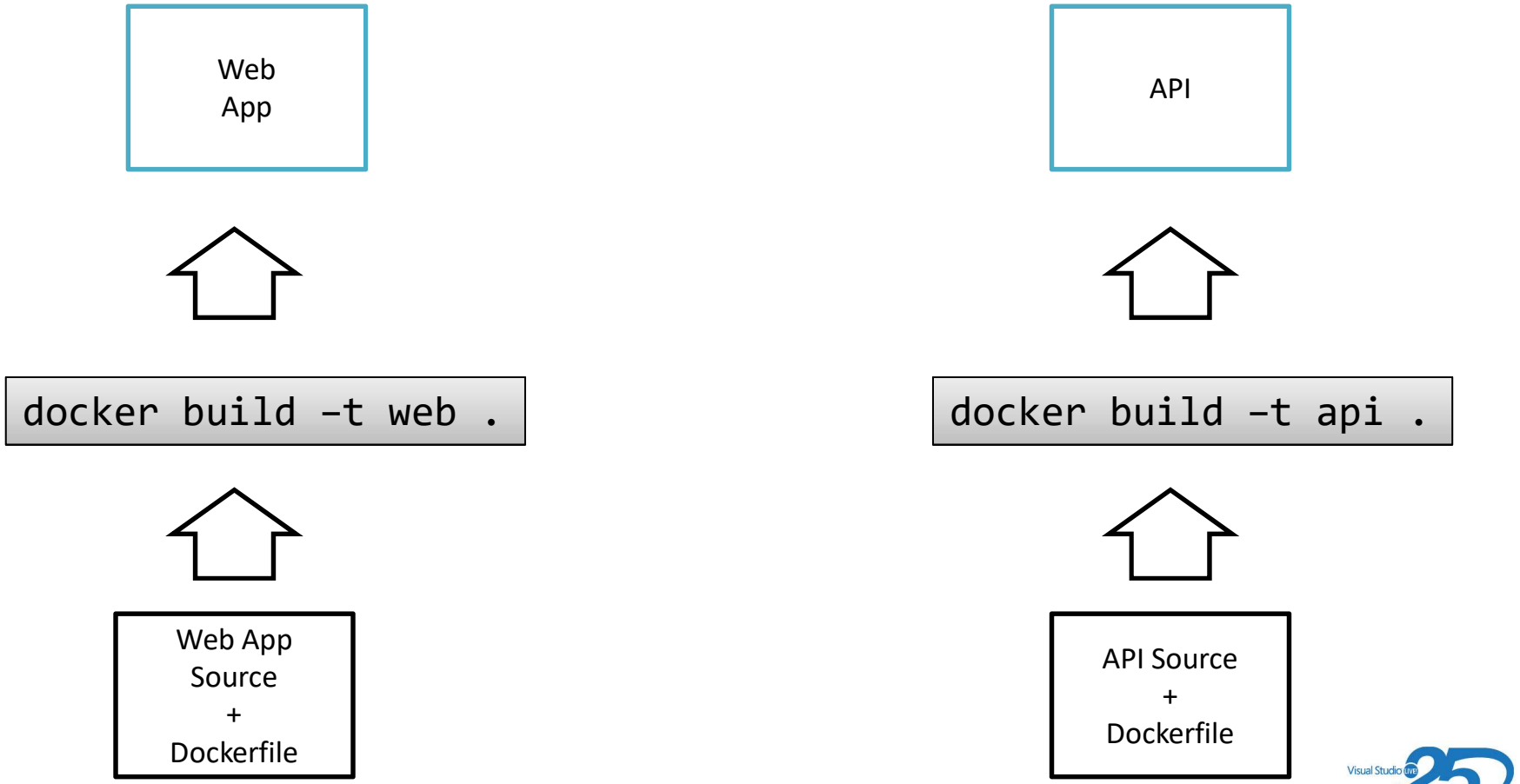
# Docker

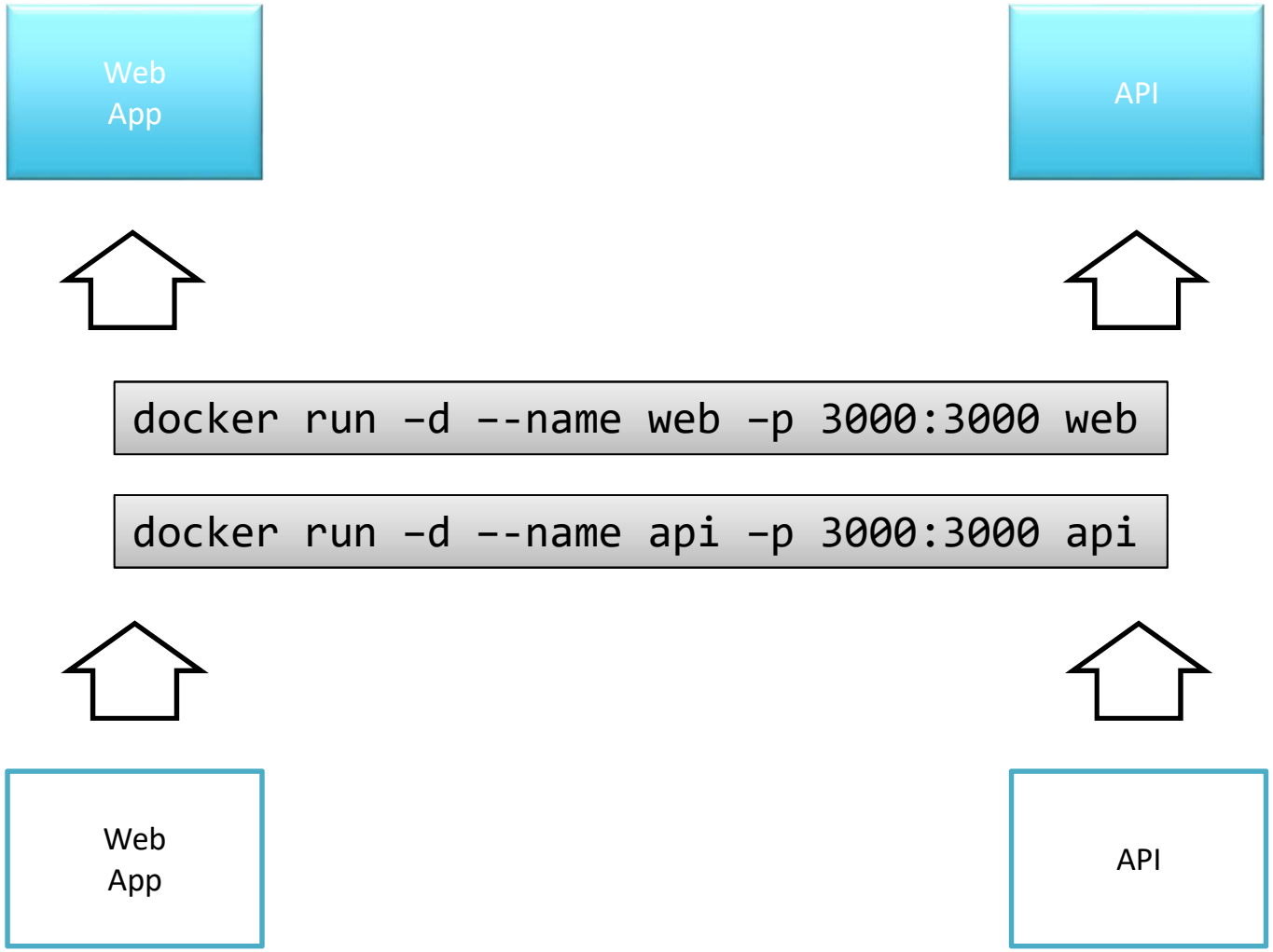
- Containerization technology
- Package applications / runtime dependencies for deployment
- Package multi-container deployments with networking
- Runtime environment
- Develop once, run anywhere
- Supported by orchestration platforms, for distribution, management and scale

# Full stack encapsulation



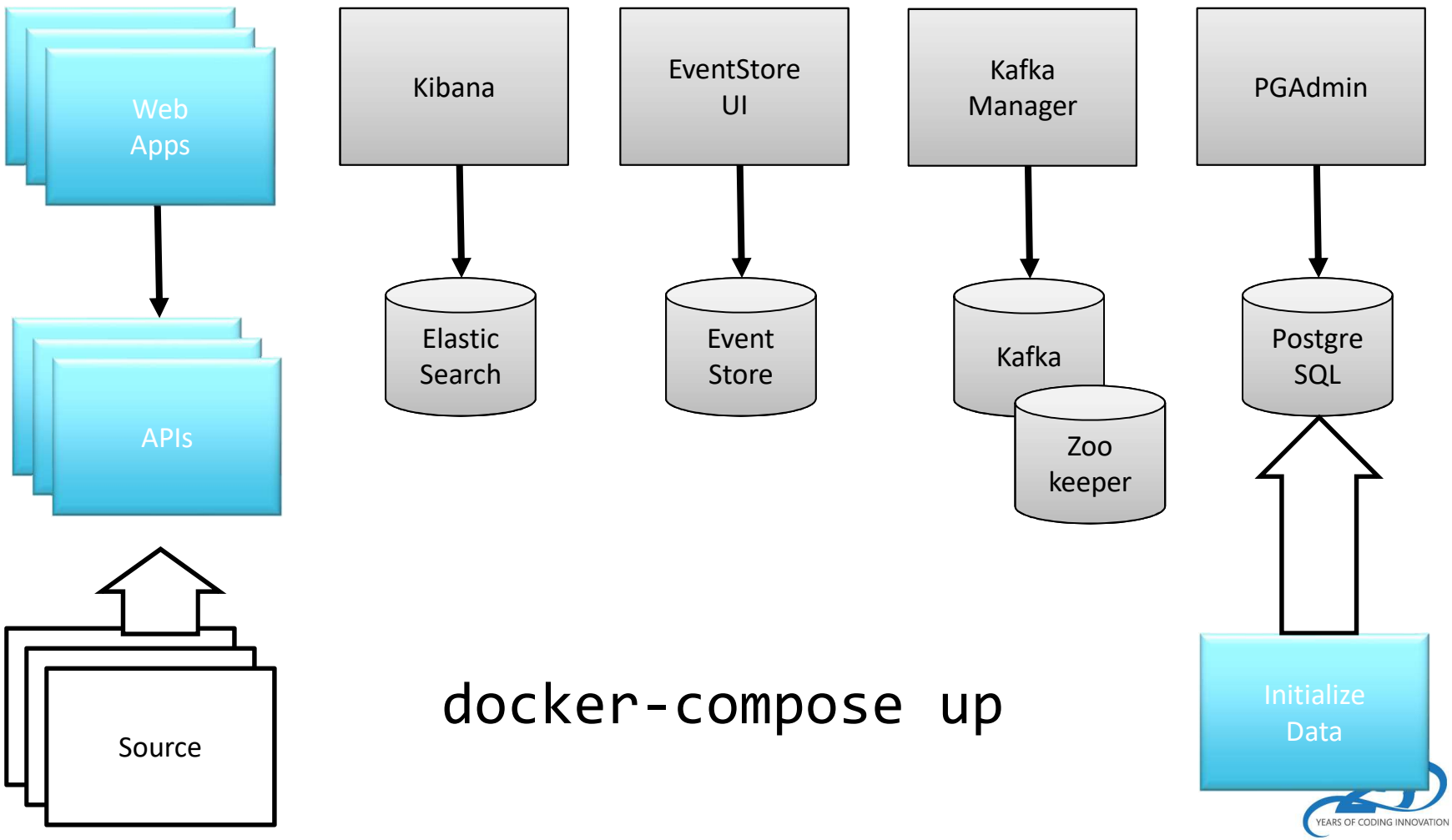



Web
App
API
docker build -t web .
docker build -t api .
Web App
Source
+
Dockerfile
API Source
+
Dockerfile


Web
App
API
docker run -d --name web -p 3000:3000 web
docker run -d --name api -p 3000:3000 api
Web
App
API

# Docker Compose

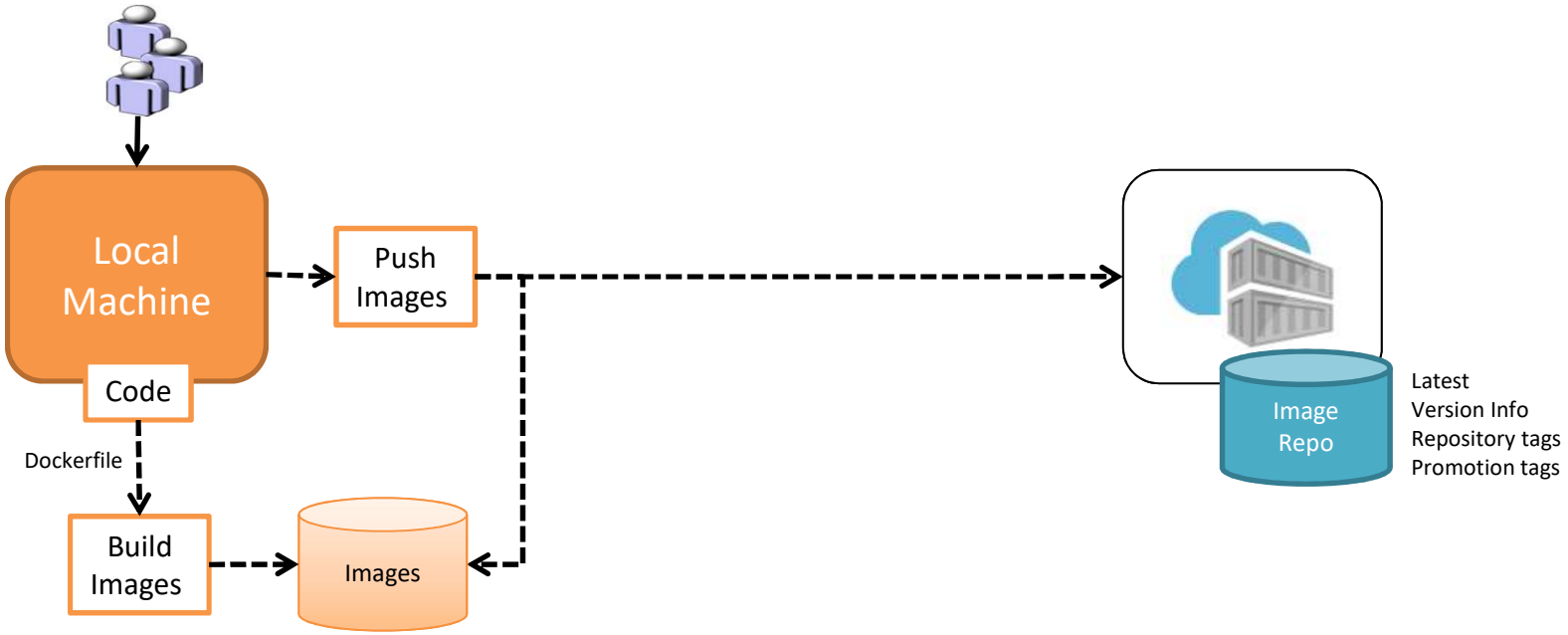
- Run multi-container applications
- Create a service definition file
  - YML (YAML) file “docker-compose.yml”
- Defines one or more service
  - Container image
  - Ports
  - Network
  - Volumes
  - Environment variables
  - Secrets
  - Configuration


Web Apps
APIs
Source
Kibana
Elastic Search
EventStore UI
Event Store
Kafka Manager
Kafka
Zoo keeper
PGAdmin
Postgre SQL
Initialize Data
docker-compose up

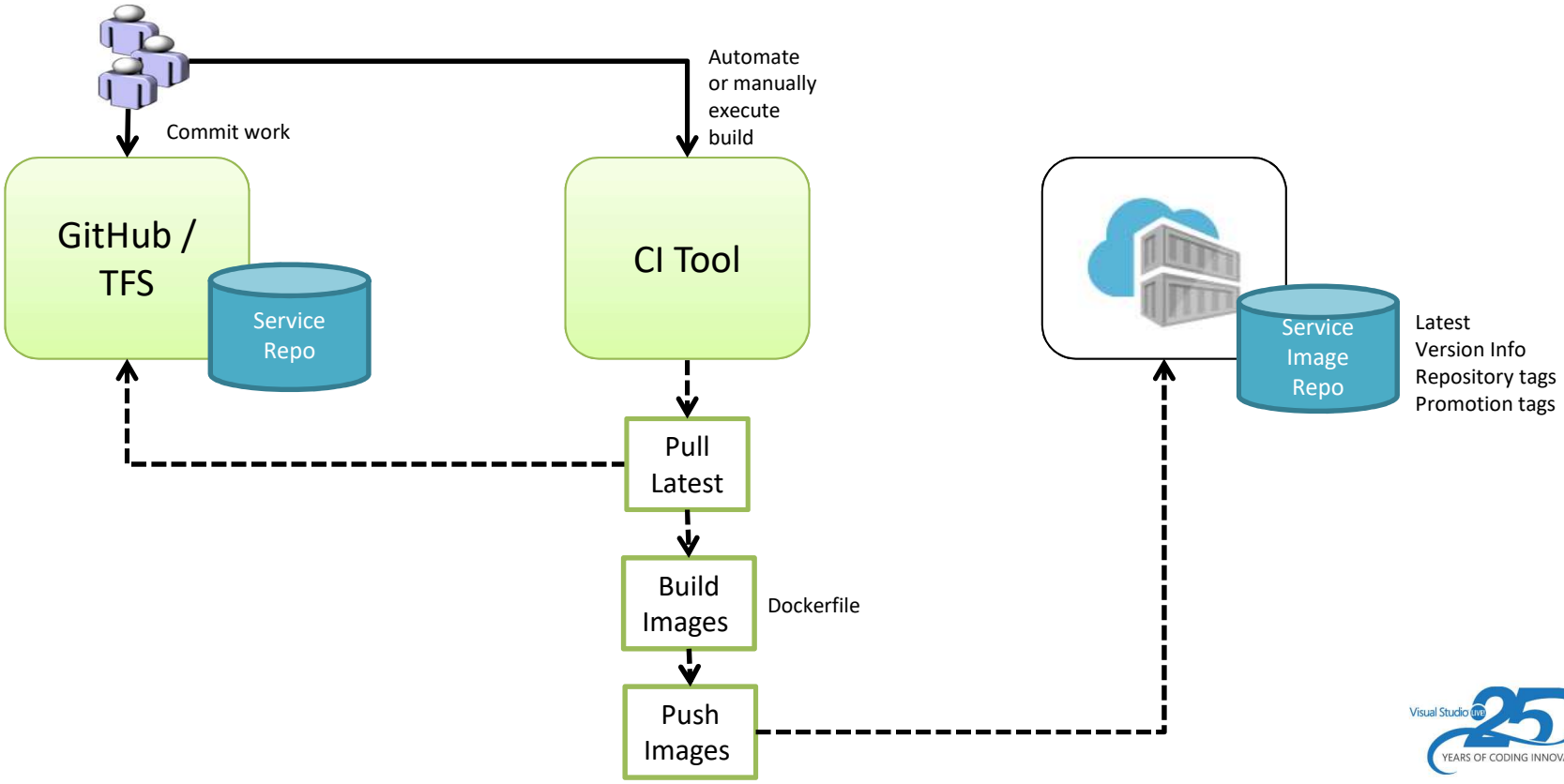
Docker Compose

# DEMO

# Image Registry



# Automated Image Builds / Tagging



# Automated Image Builds / Tagging

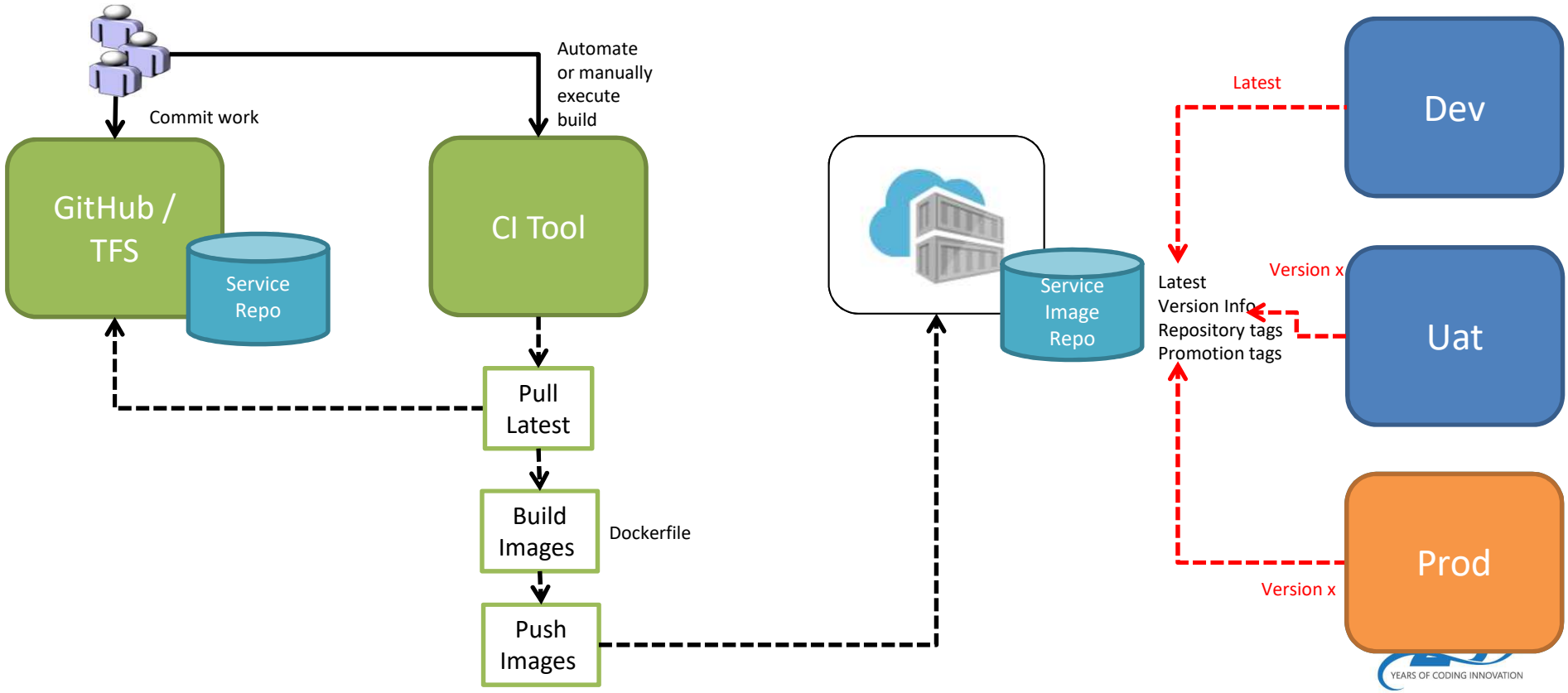
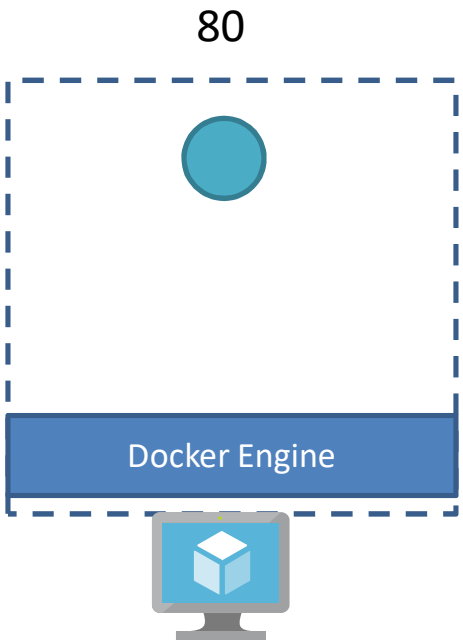


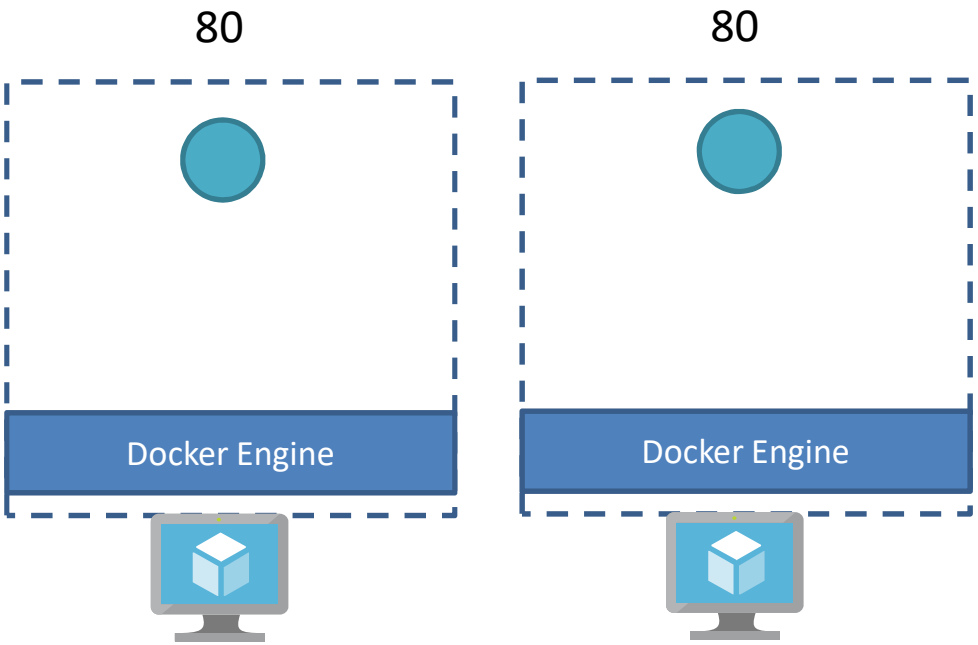
Image Builds, CI, Azure Container Registry

# DEMO

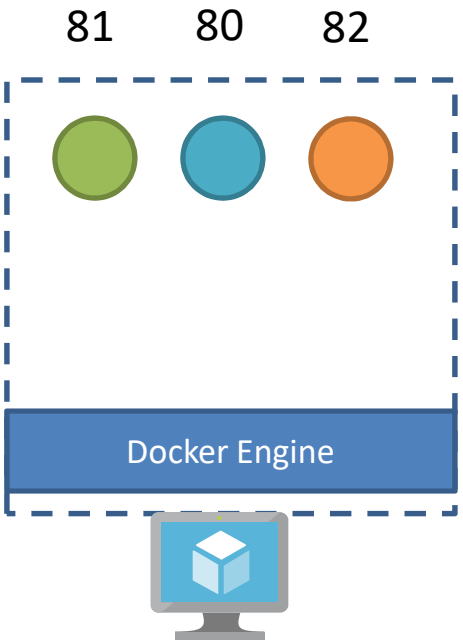
# Single VM



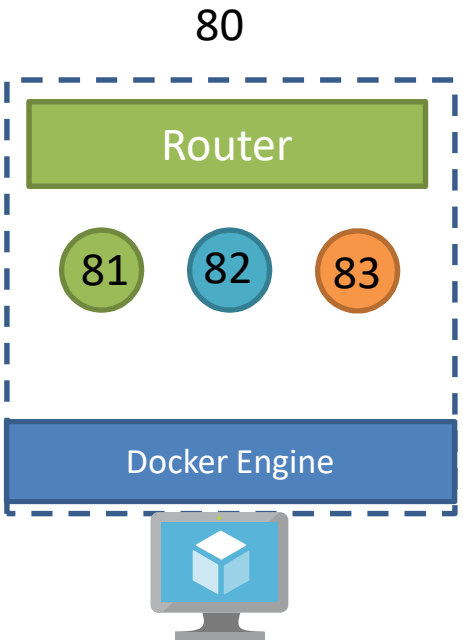
# Load Balanced VMs



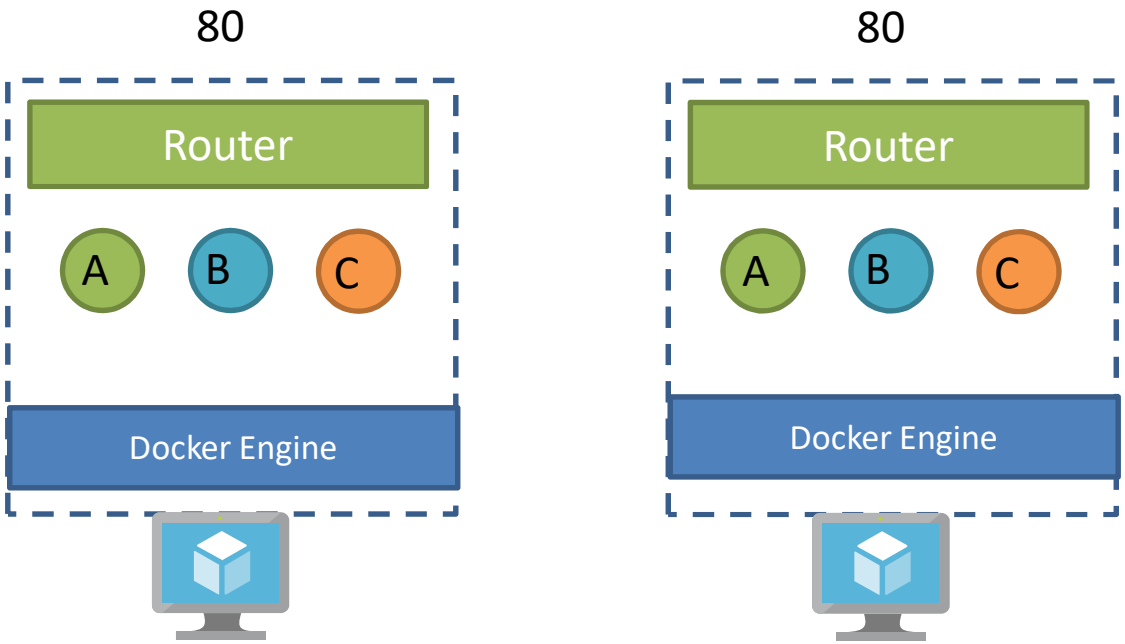
# Multiple Container Per VM



# Multiple Container Routing



# Web Apps for Containers



# Web Apps for Containers

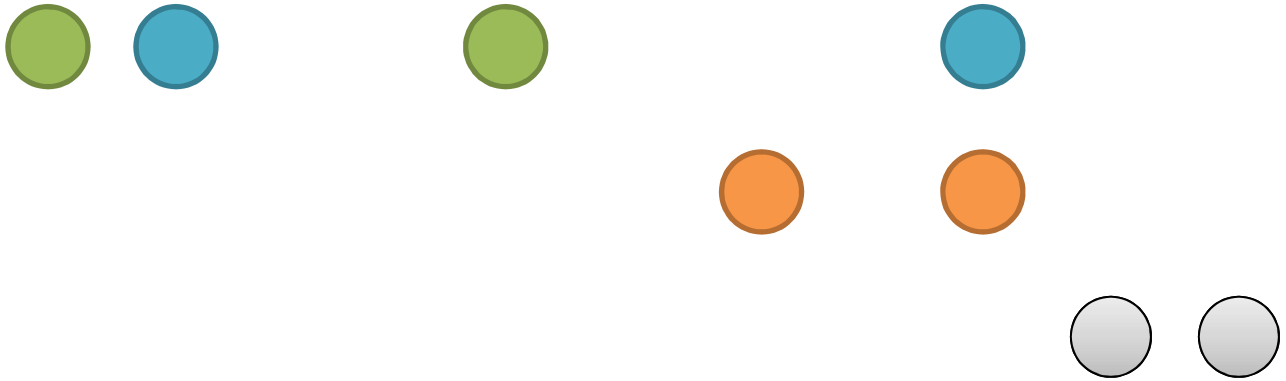
# Web Apps for Containers

- PaaS host environment
- Scale up / out the same as Web Apps / App Services
- Each container (app) assigned DNS
  - At port 80 / 443

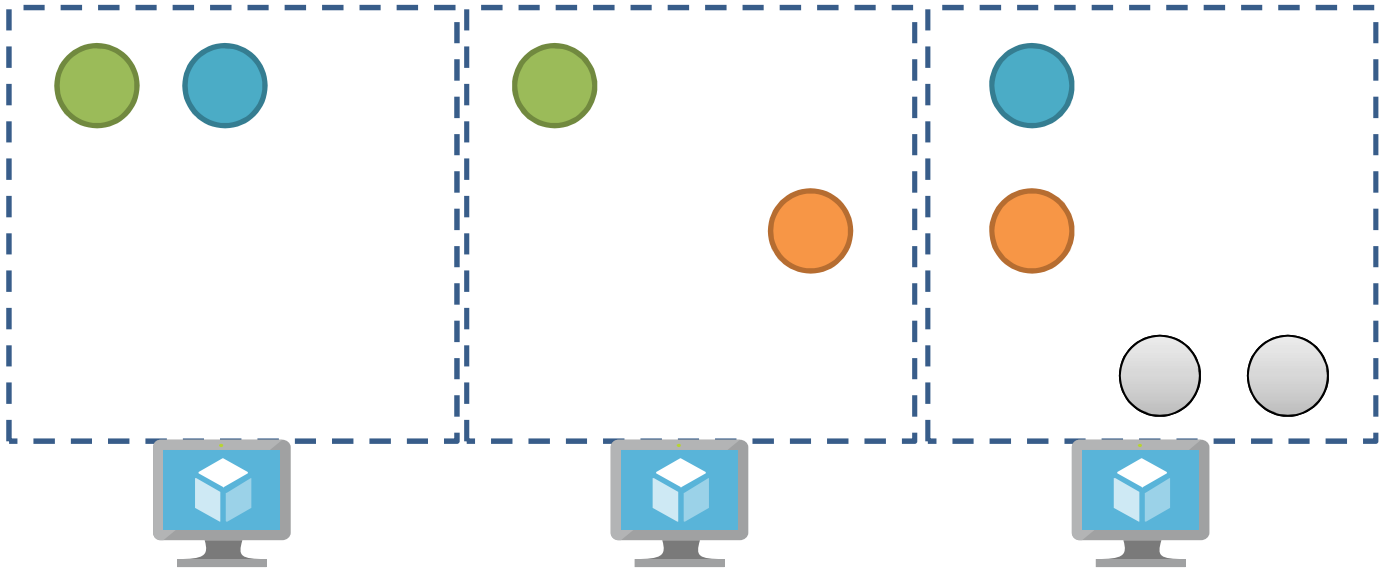
Web Apps for Containers

# DEMO

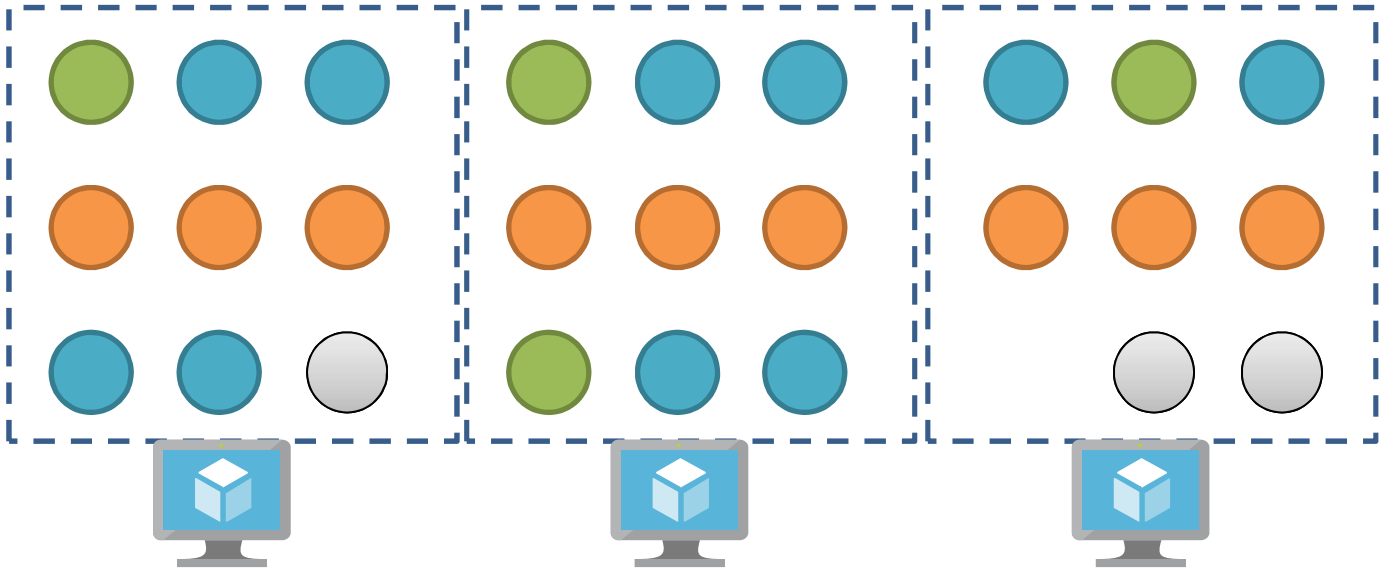
# Serverless



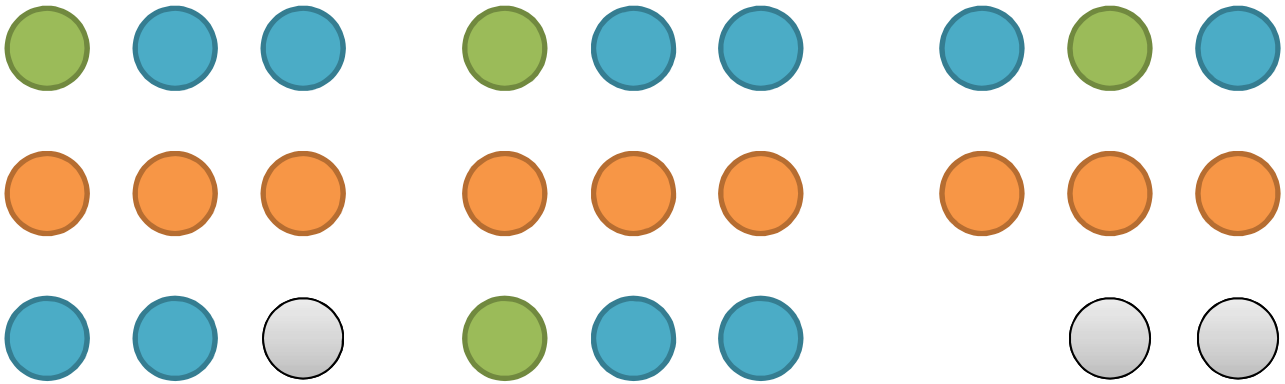
# Serverless



# Serverless

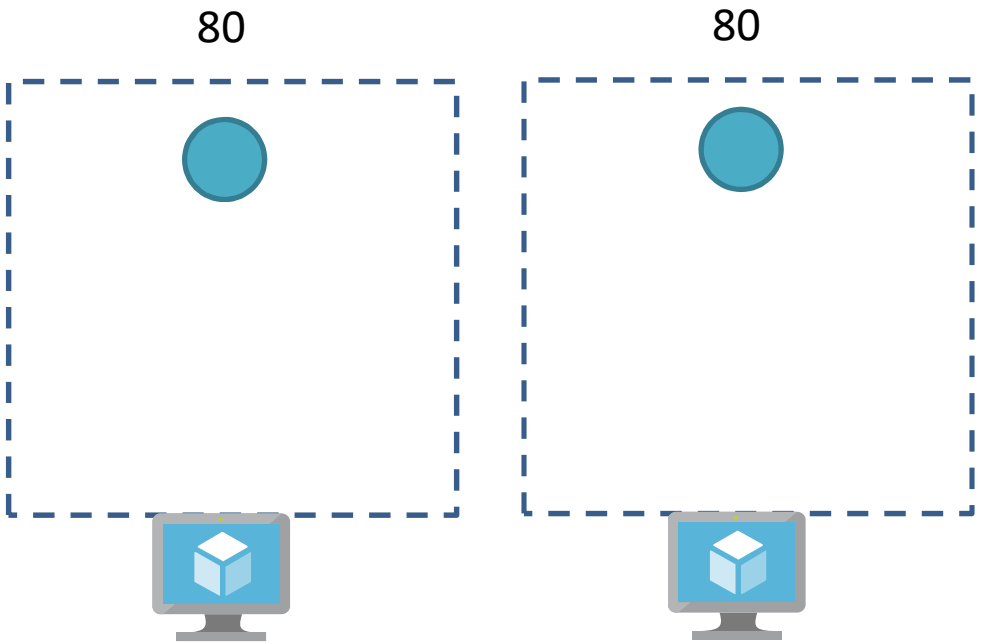


# Serverless

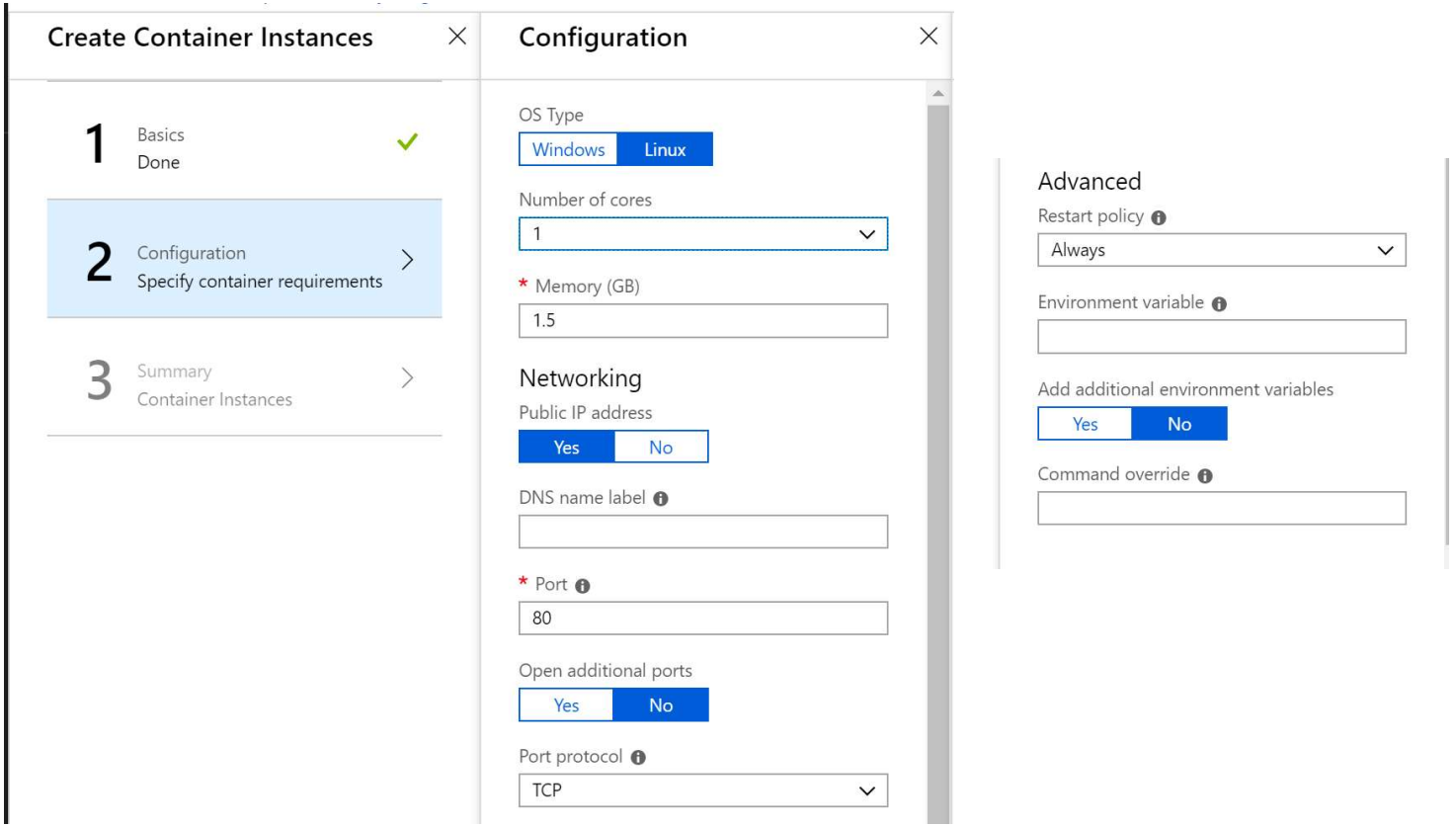


# Azure Container Instances

# Azure Container Instance



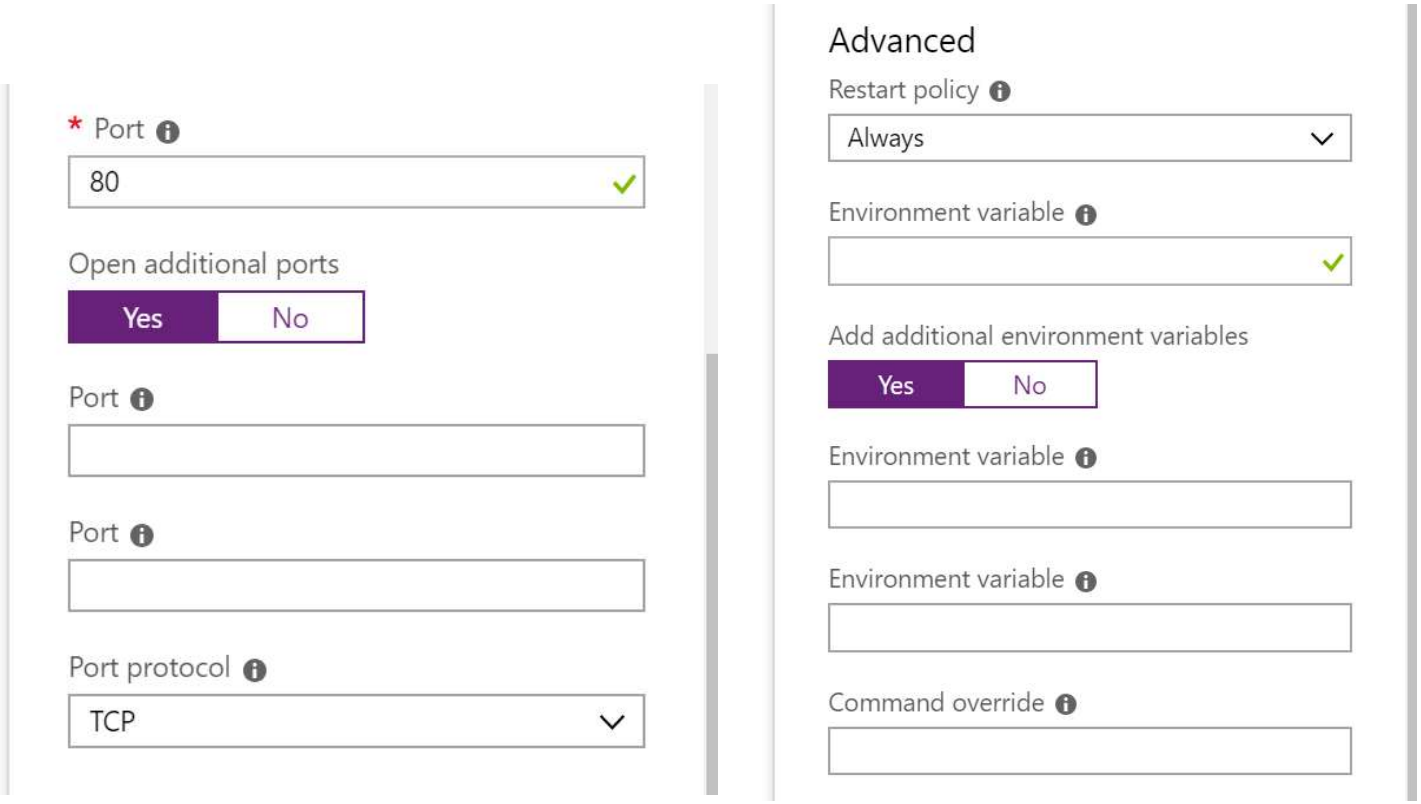
# Azure Container Instance



Azure Container Instances

# DEMO

# Azure Container Instance



Now the fun begins…

# Container Platforms / Considerations

## Host

- Azure Kubernetes Service
- IaaS
  - Any
- On premise / hybrid
  - Any
- Azure Service Fabric
  - Native + Containers

## Platforms

- Kubernetes
- Mesosphere DC/OS
  - Kubernetes
- Docker CE / EE
  - Swarm
  - Kubernetes
- Service Fabric

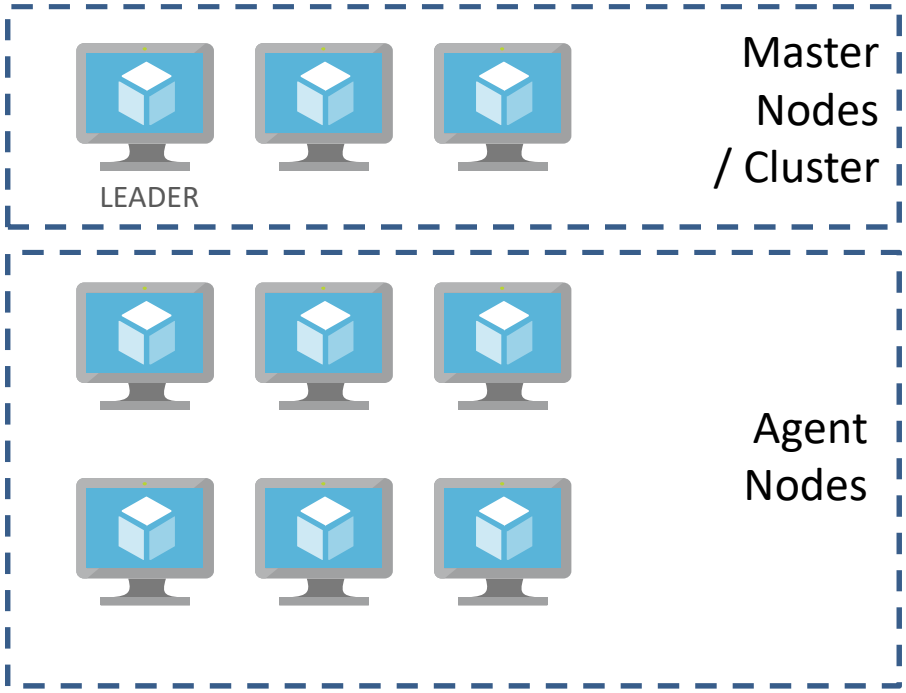
## Infrastructure

- Management cluster
- Agent node cluster
- Networking / Routing / DNS
- Docker registry

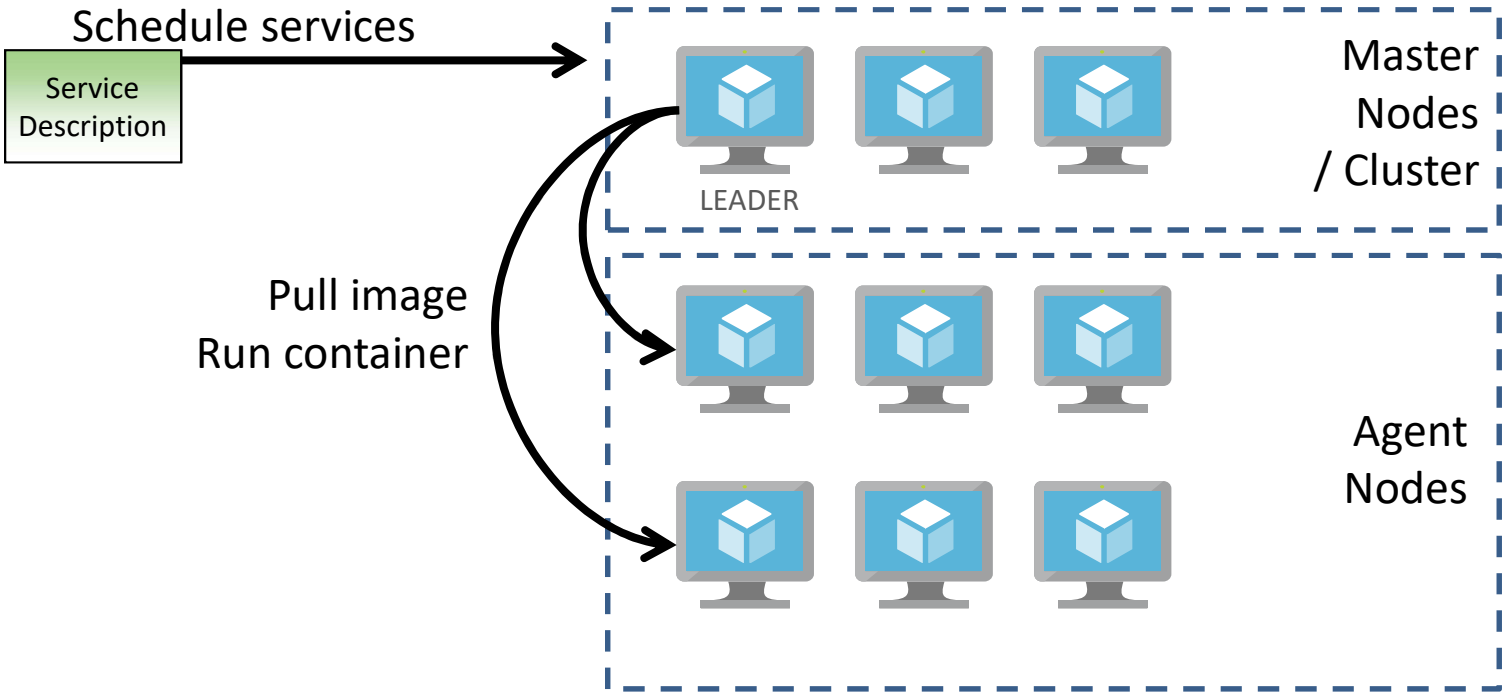
## Core Features

- Service registration and discovery
- Load balancing
- Routing
- Auto-scaling
- Self-healing
- Upgrade, rollback, recovery
- Secret management

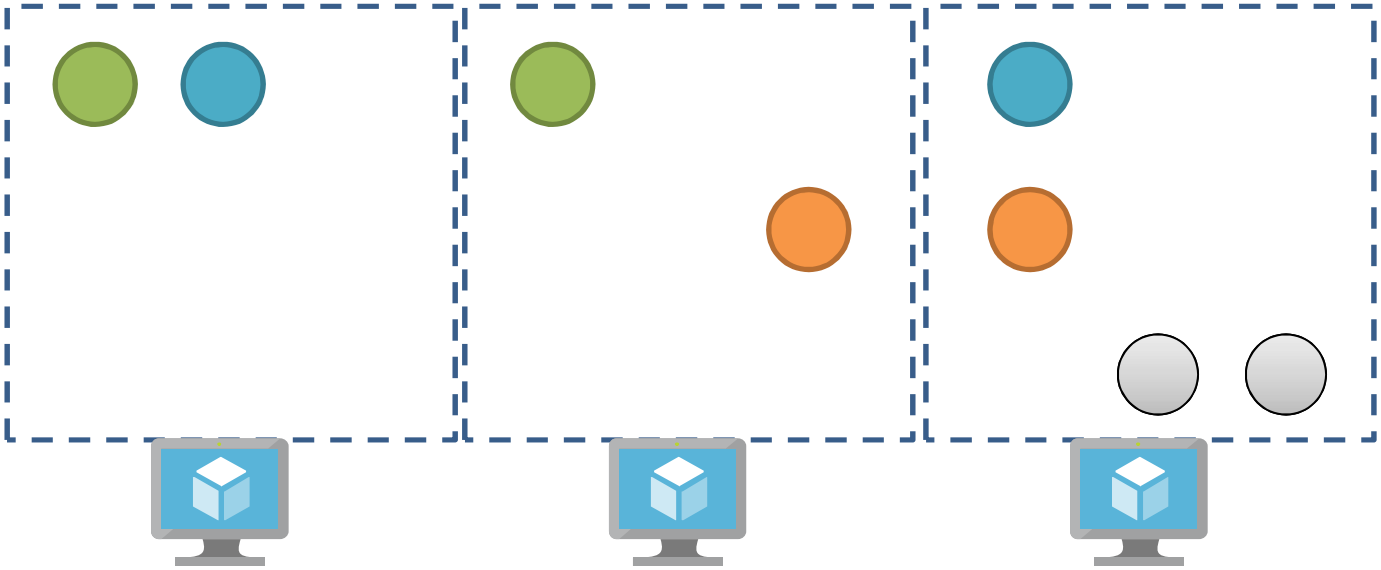
# Platform Master and Agent clusters



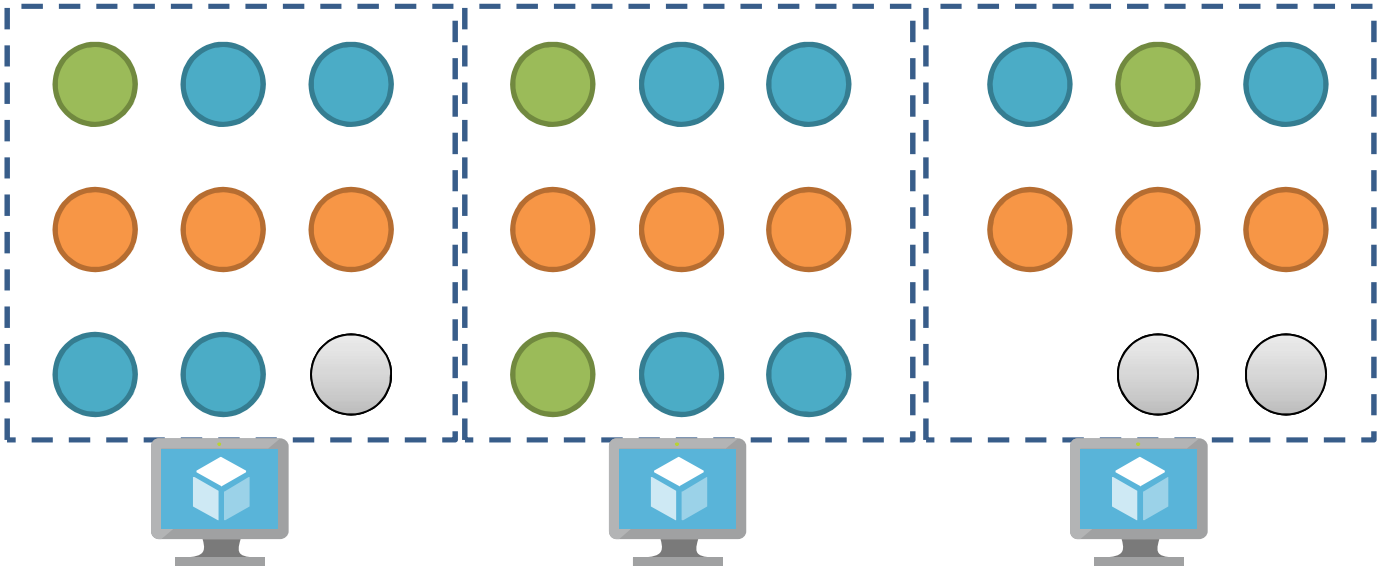
# Scheduling


Schedule services
Service Description
LEADER
Master Nodes / Cluster
Pull image
Run container
Agent Nodes

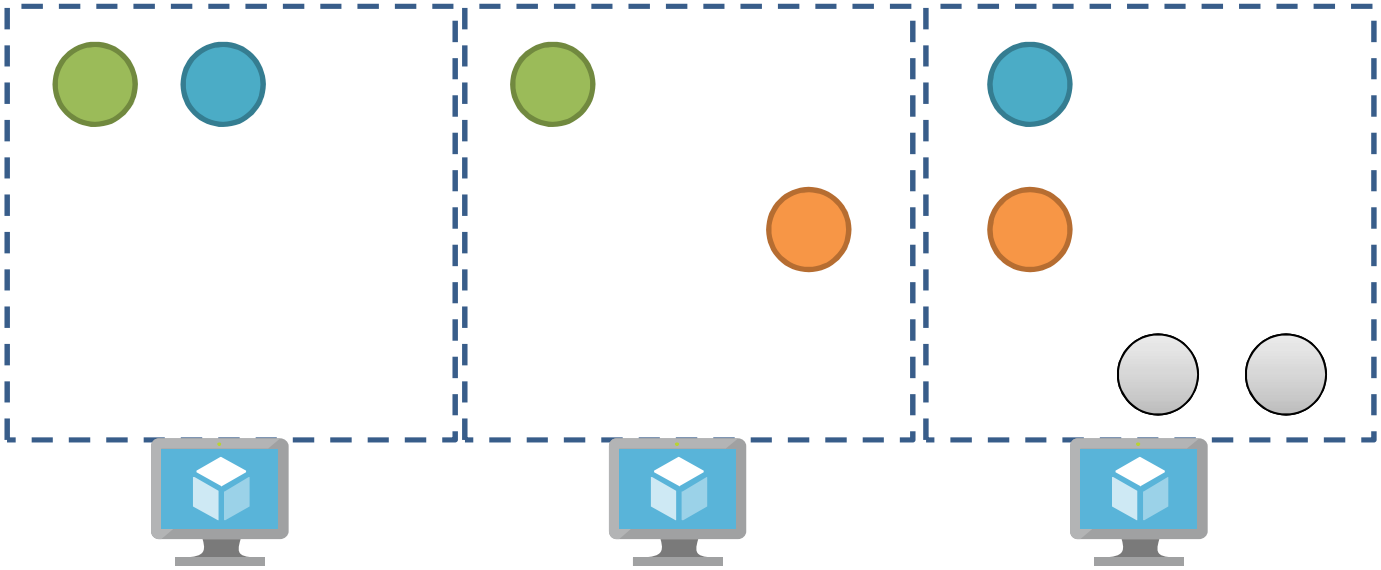
# Scheduling Services



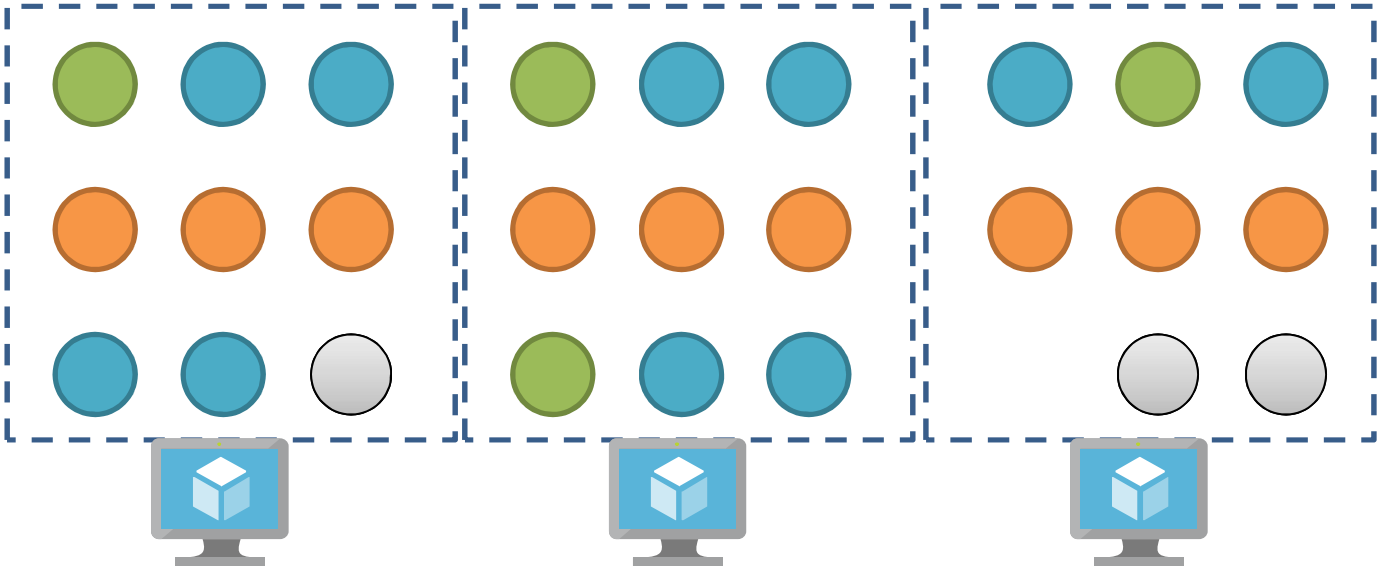
# Scheduling Services



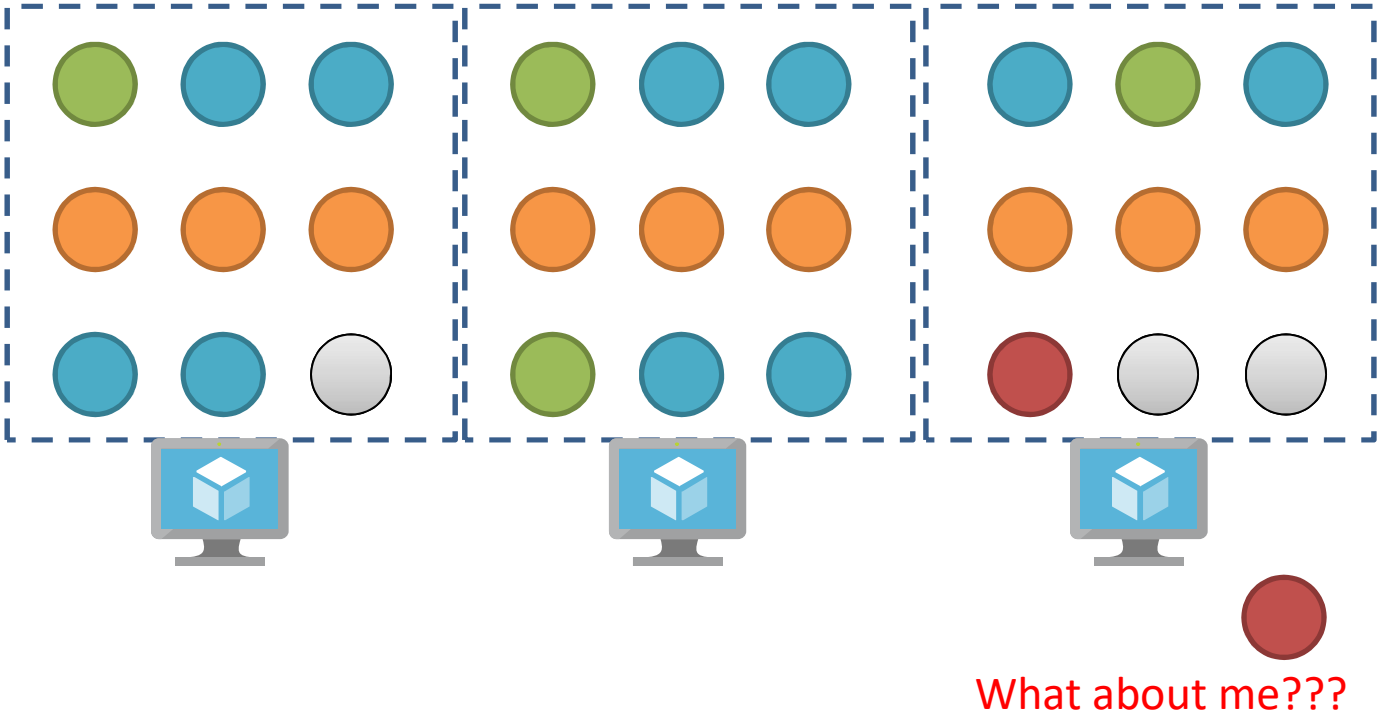
# Server Density



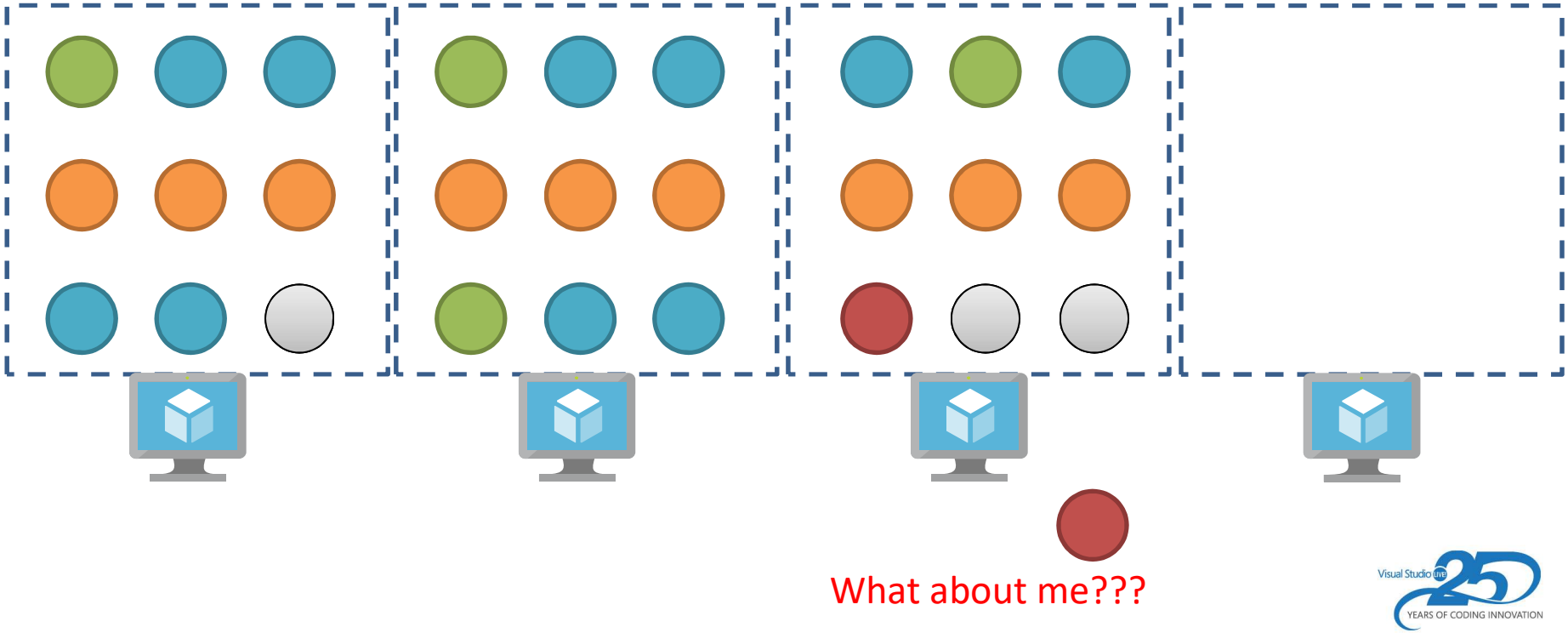
# Server Density



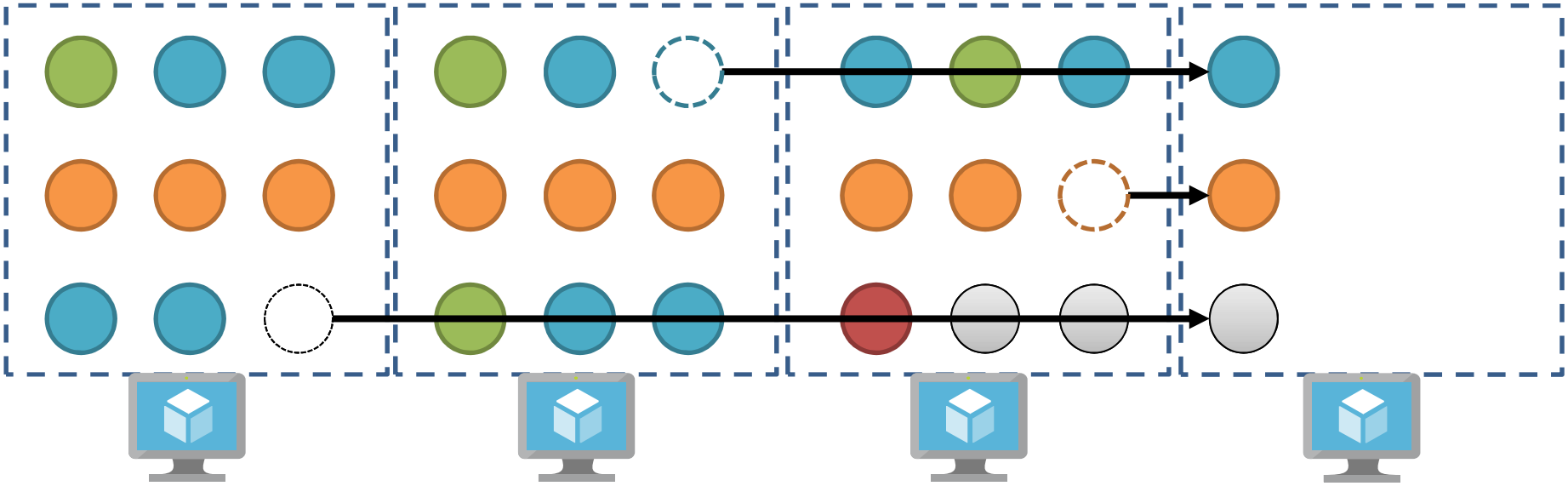
# Room for Upgrades, Restarts



# Adding Nodes

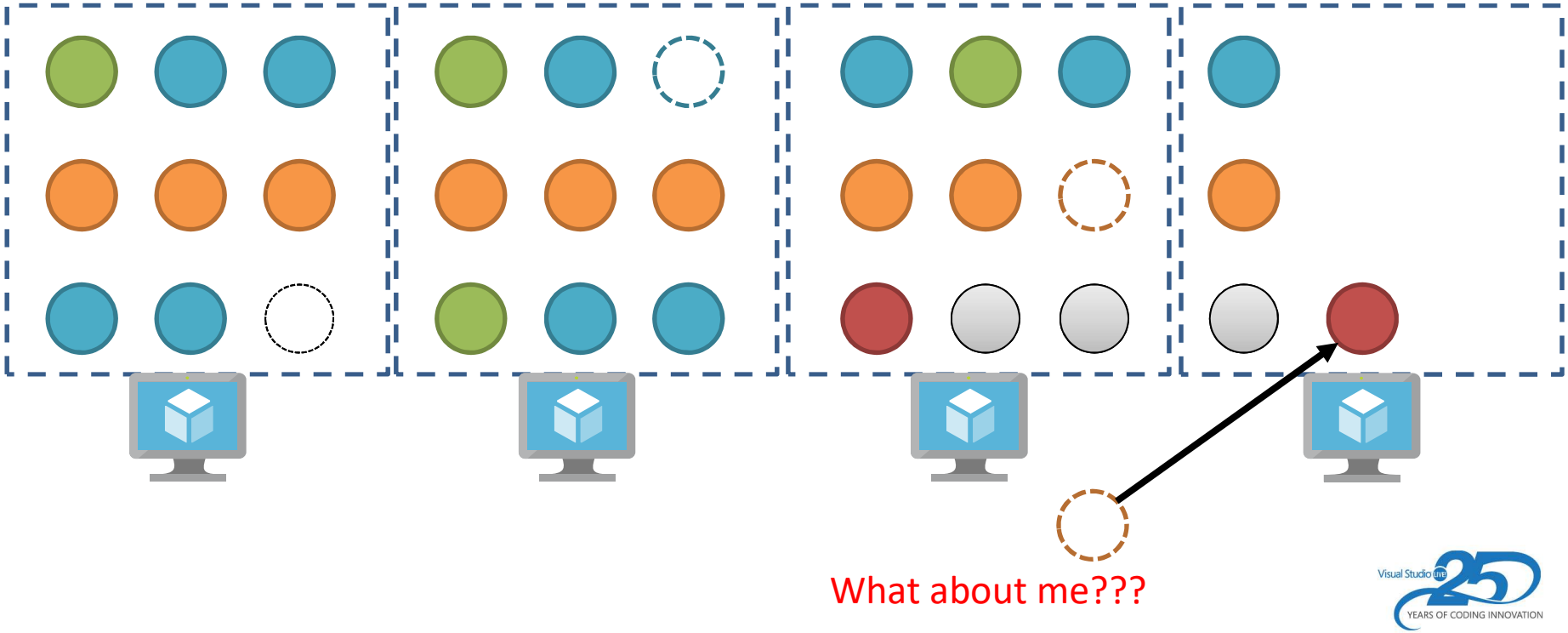


# Redistribution of Instances



What about me???

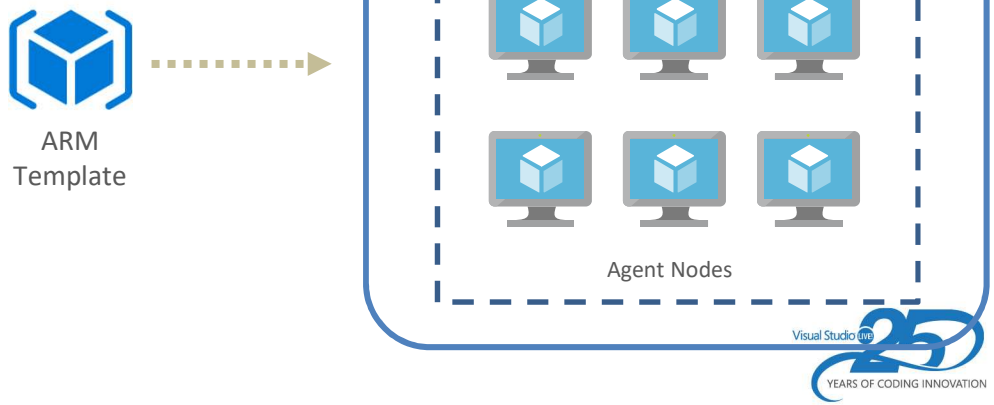
# Room for New Instances



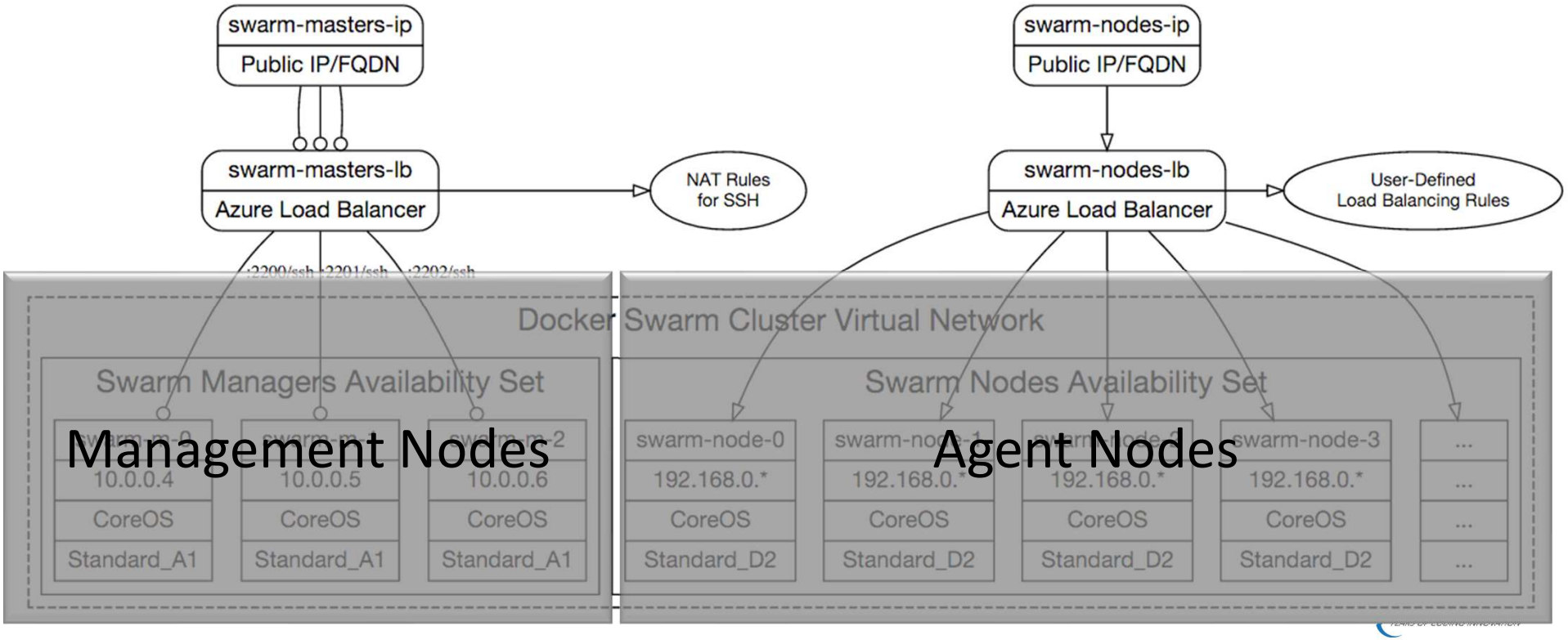
# IaaS Templates

- Choose from existing ARM templates
  - Or, don't and deploy your own

- Docker Swarm
- Docker EE
- Mesosphere DC/OS
- Kubernetes
- Others…



# Docker Swarm



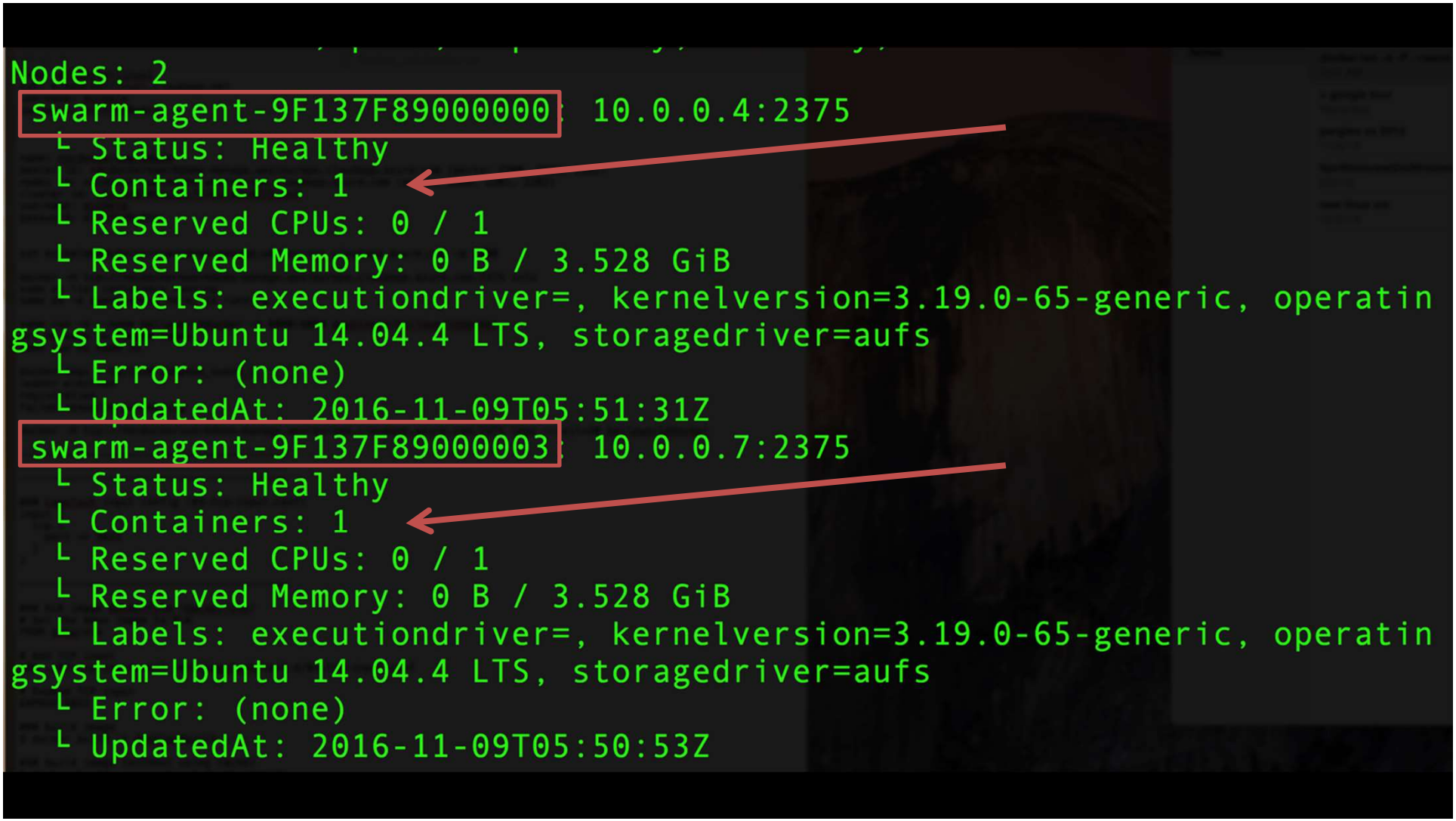

Management Nodes

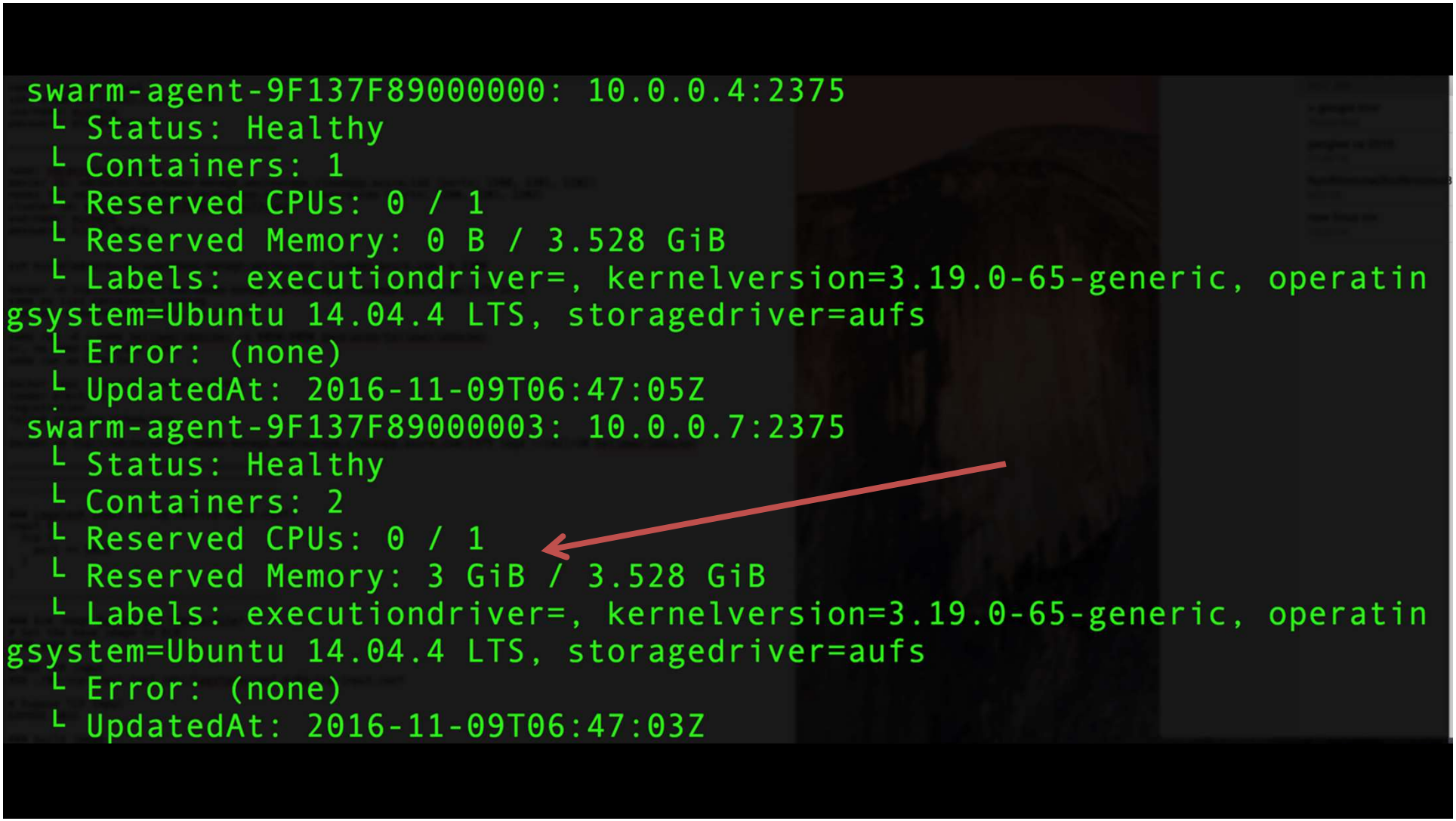
Agent Nodes

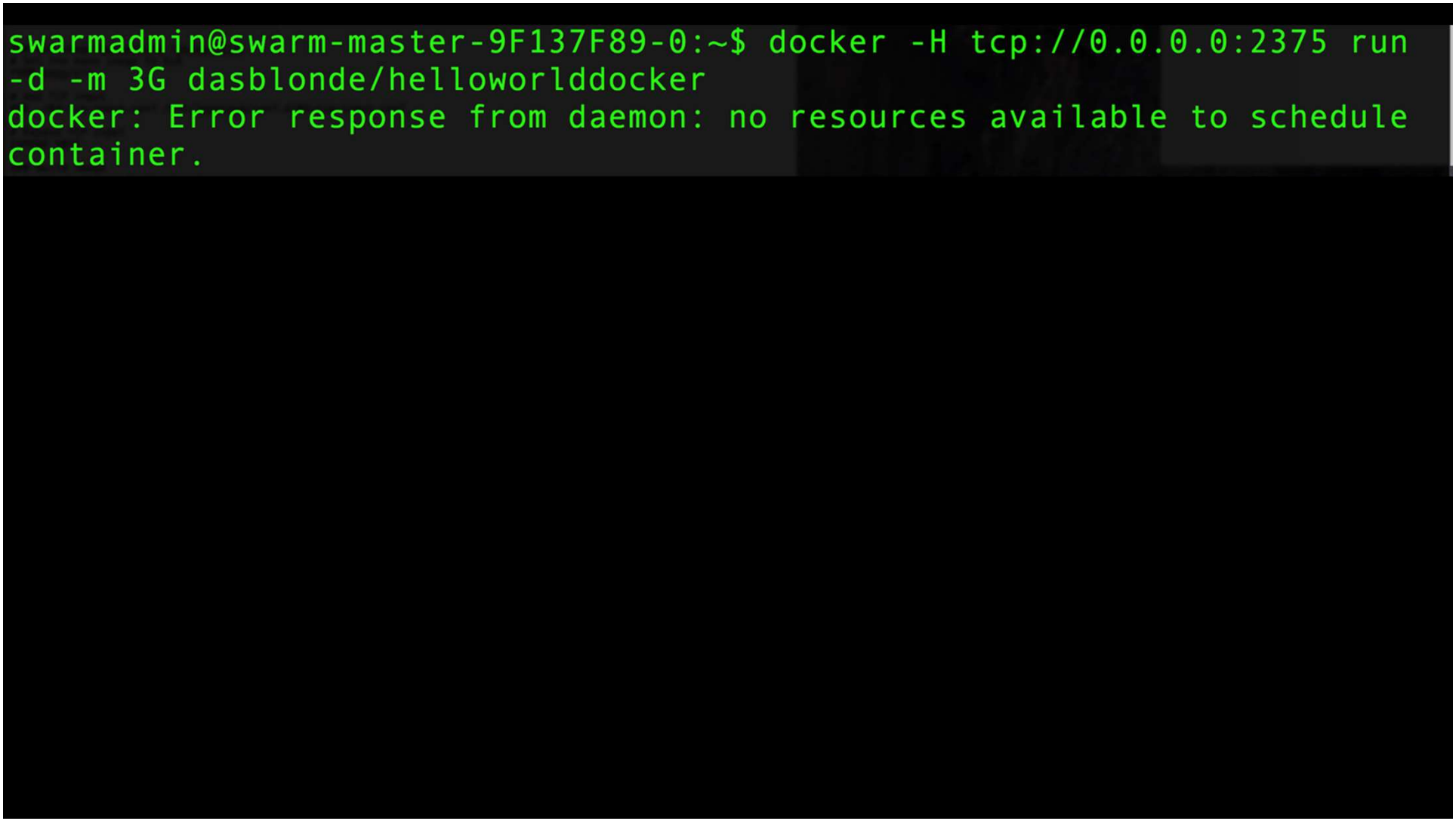
Docker Swarm

Docker EE UI

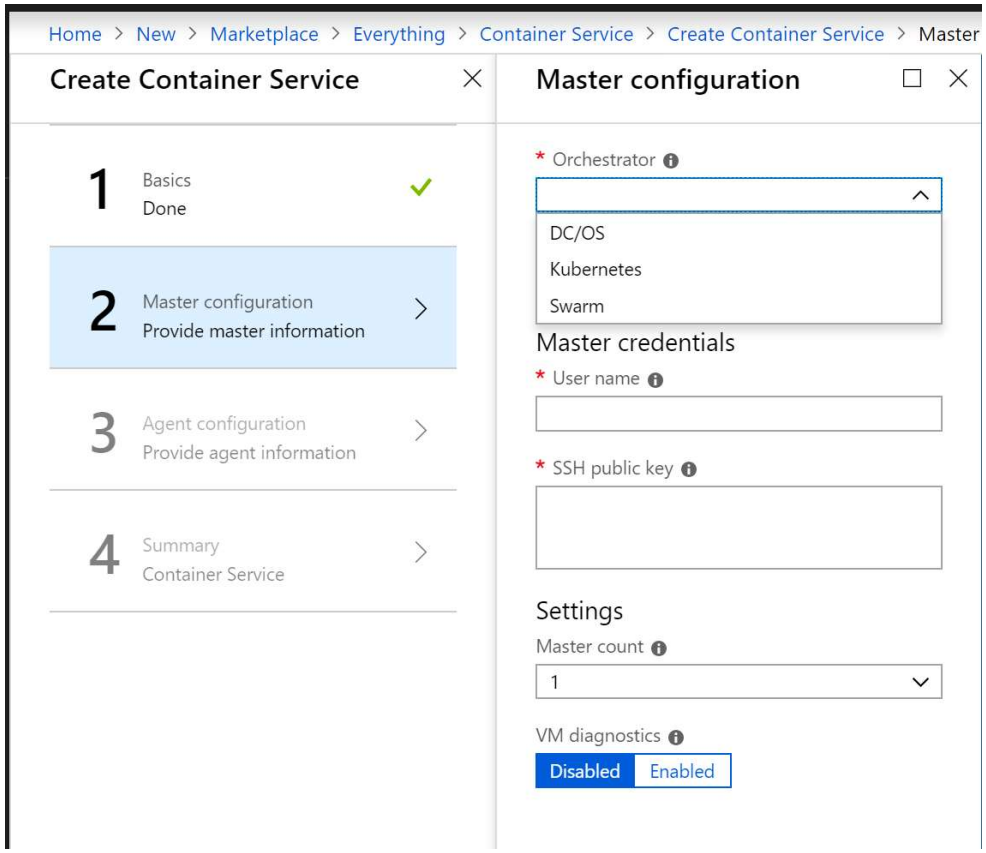
# DEMO


Nodes: 2
swarm-agent-9F137F89000000: 10.0.0.4:2375
└ Status: Healthy
└ Containers: 1
└ Reserved CPUs: 0 / 1
└ Reserved Memory: 0 B / 3.528 GiB
└ Labels: executiondriver=, kernelversion=3.19.0-65-generic, operatingsystem=Ubuntu 14.04.4 LTS, storagedriver=aufs
└ Error: (none)
└ UpdatedAt: 2016-11-09T05:51:31Z
swarm-agent-9F137F89000003: 10.0.0.7:2375
└ Status: Healthy
└ Containers: 1
└ Reserved CPUs: 0 / 1
└ Reserved Memory: 0 B / 3.528 GiB
└ Labels: executiondriver=, kernelversion=3.19.0-65-generic, operatingsystem=Ubuntu 14.04.4 LTS, storagedriver=aufs
└ Error: (none)
└ UpdatedAt: 2016-11-09T05:50:53Z

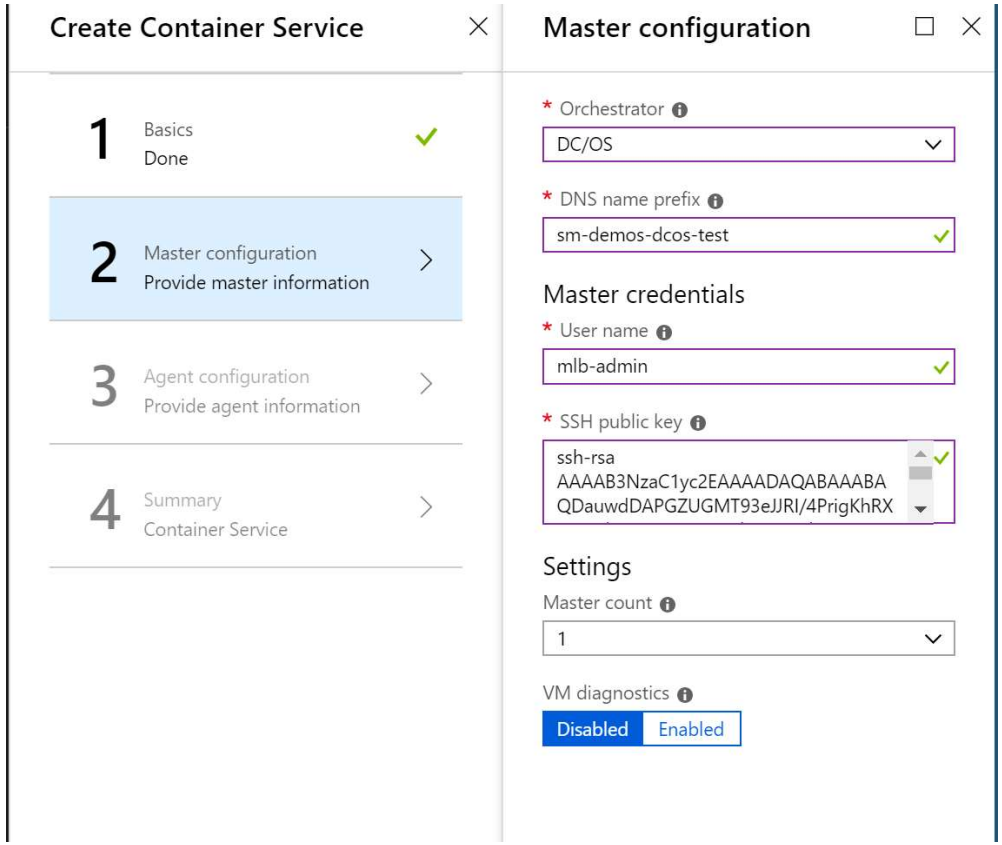

swarm-agent-9F137F89000000: 10.0.0.4:2375
└ Status: Healthy
└ Containers: 1
└ Reserved CPUs: 0 / 1
└ Reserved Memory: 0 B / 3.528 GiB
└ Labels: executiondriver=, kernelversion=3.19.0-65-generic, operatingsystem=Ubuntu 14.04.4 LTS, storagedriver=aufs
└ Error: (none)
└ UpdatedAt: 2016-11-09T06:47:05Z
swarm-agent-9F137F89000003: 10.0.0.7:2375
└ Status: Healthy
└ Containers: 2
└ Reserved CPUs: 0 / 1
└ Reserved Memory: 3 GiB / 3.528 GiB
└ Labels: executiondriver=, kernelversion=3.19.0-65-generic, operatingsystem=Ubuntu 14.04.4 LTS, storagedriver=aufs
└ Error: (none)
└ UpdatedAt: 2016-11-09T06:47:03Z


swarmadmin@swarm-master-9F137F89-0:~$ docker -H tcp://0.0.0.0:2375 run -d -m 3G dasblonde/helloworlddocker
docker: Error response from daemon: no resources available to schedule container.

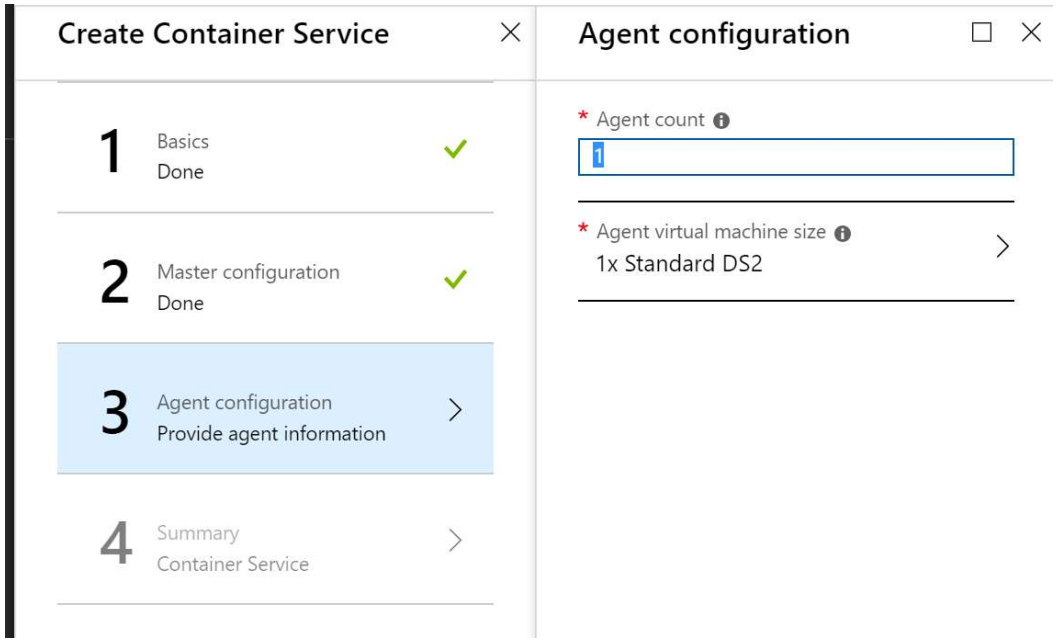
# Container Service on Azure



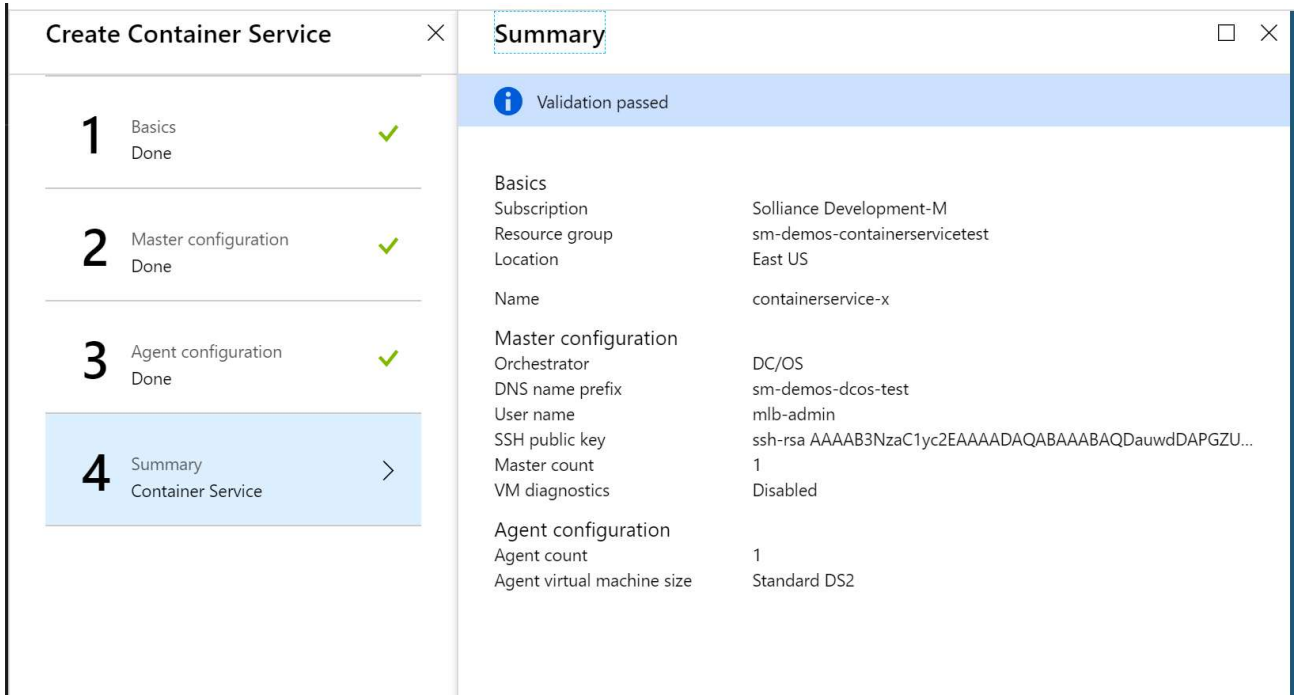
# Container Service on Azure



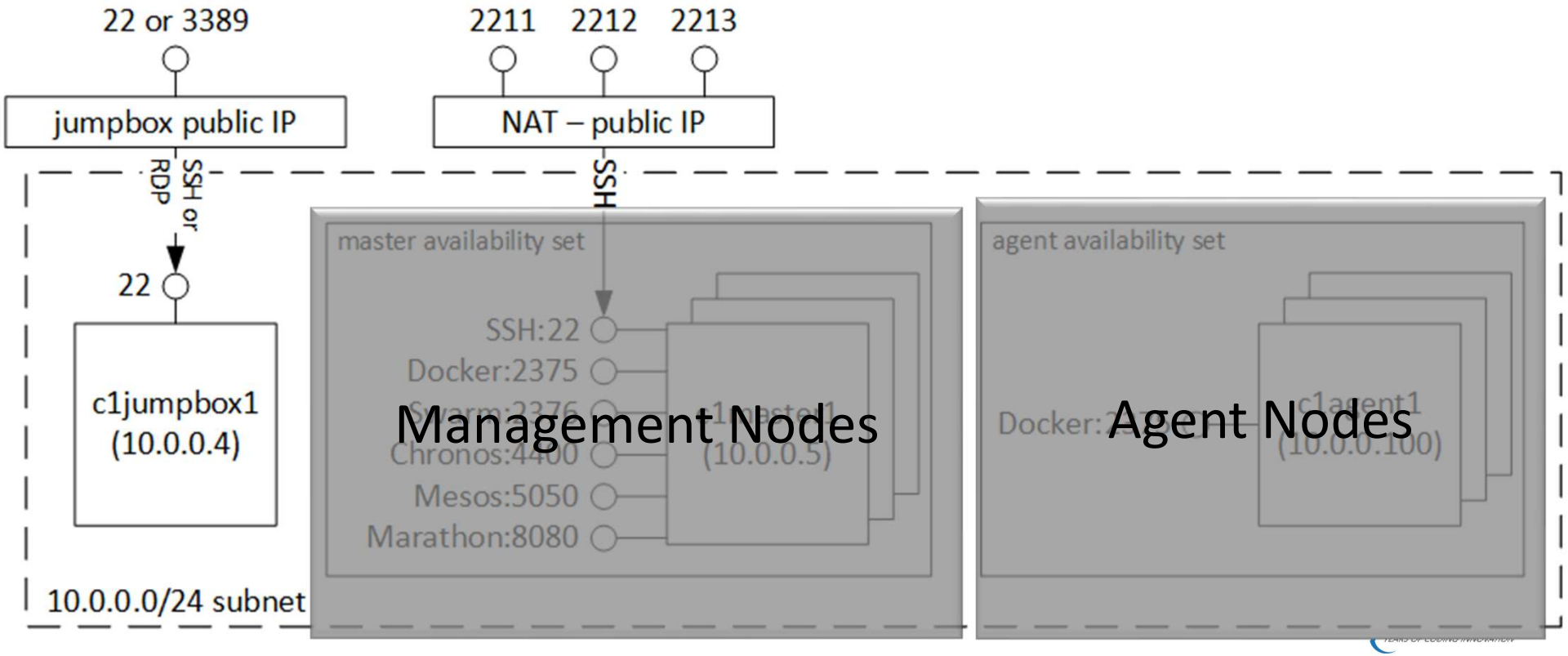
# Container Service on Azure



# Container Service on Azure



# Mesosphere DC/OS

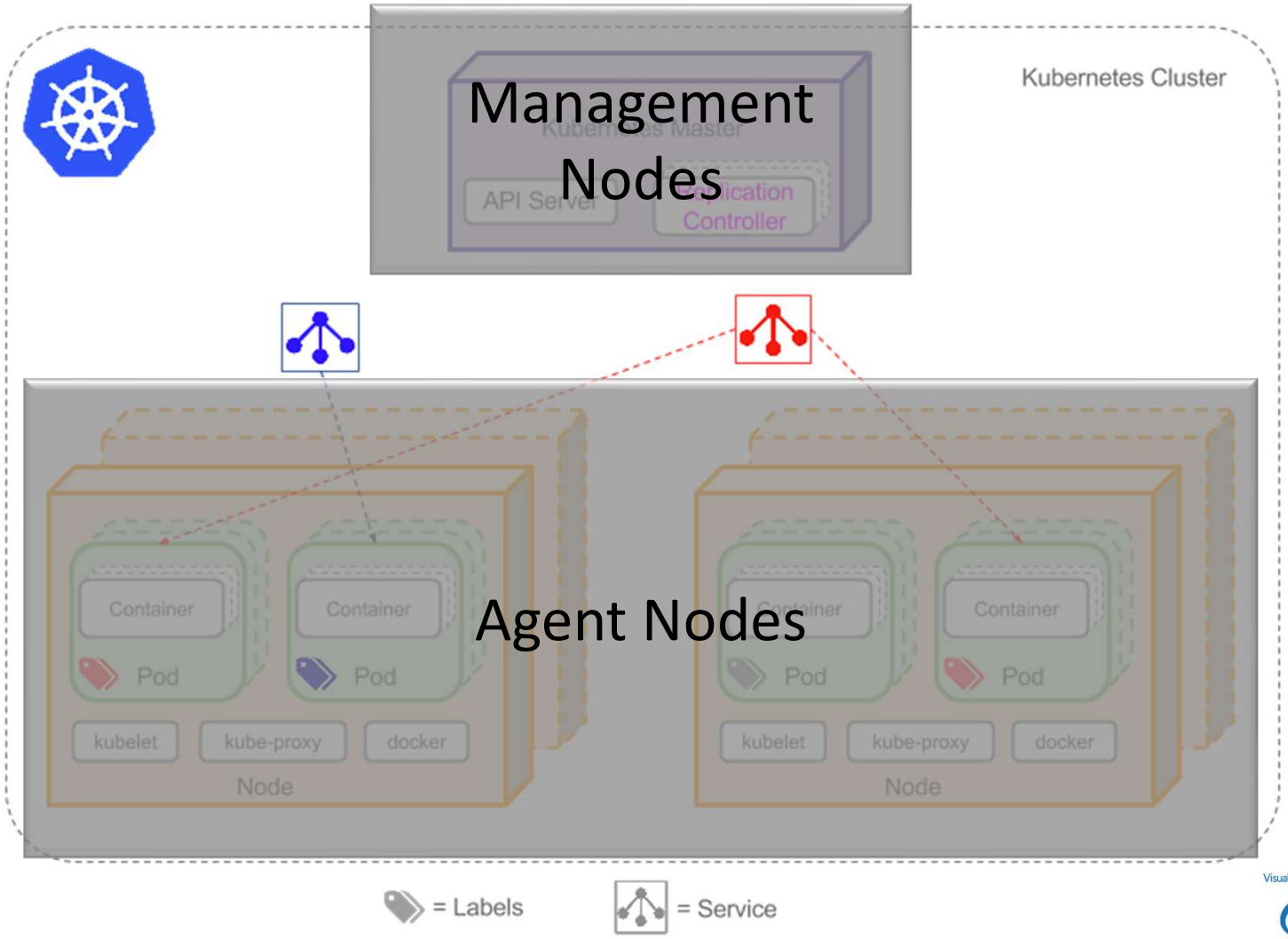


# DC / OS

- Variety of container runtimes
- Can schedule executables, without containers
- Strong concept of public/private node segregation
- Built in application packages such as load balancers or productized apps to schedule

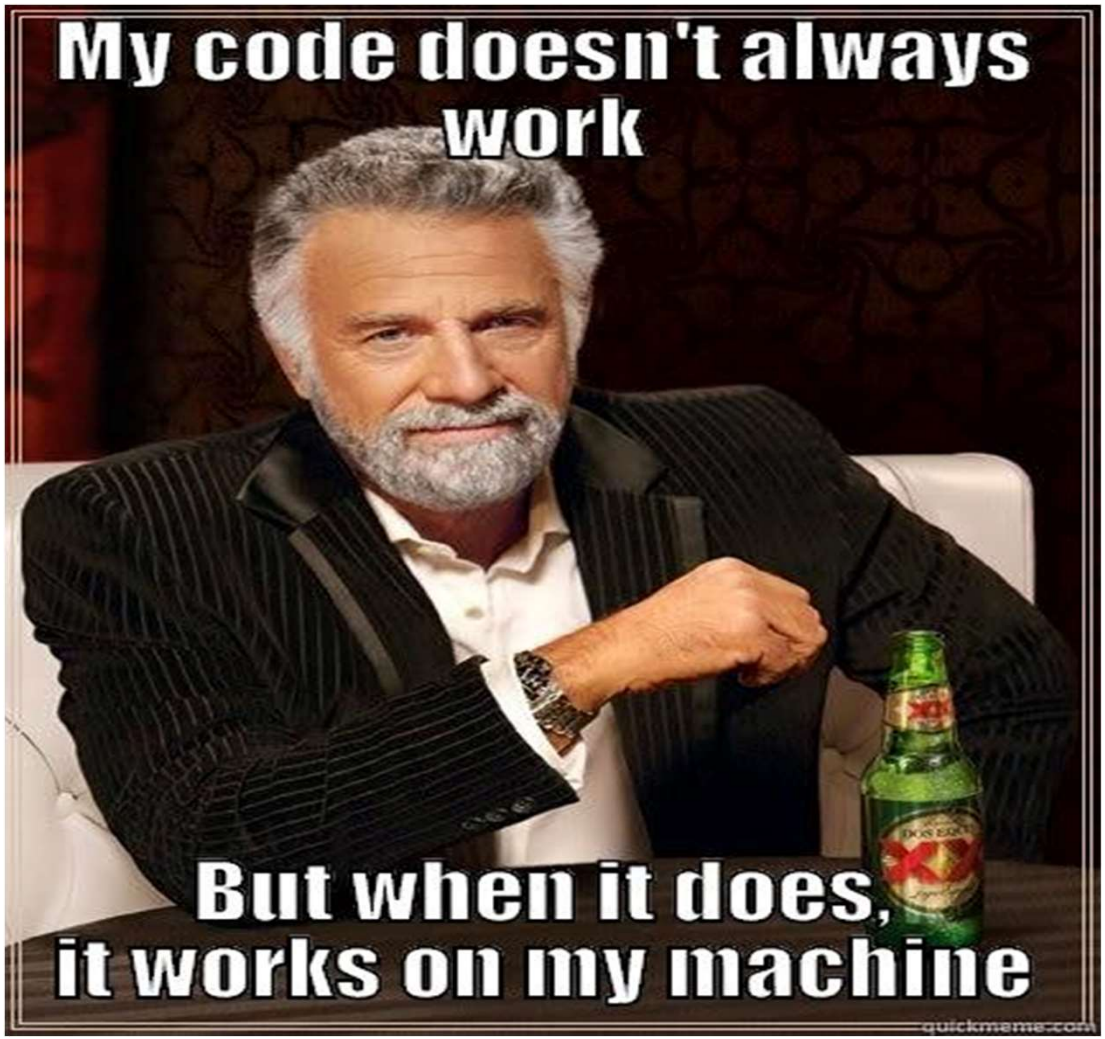
Mesosphere DC/OS

# DEMO


Kubernetes Cluster
Management Nodes
API Server
Controller
Agent Nodes
Container
Container
Pod
Pod
kubelet
kube-proxy
docker
Node
Container
Container
Pod
Pod
kubelet
kube-proxy
docker
Node
= Labels
= Service

Azure Kubernetes Service

# DEMO



# Resources

- We produced this container lab for Microsoft workshops walking you through Docker and AKS setup and deployments including load balancing:
  - https://github.com/Microsoft/MCW-Containers-and-DevOps
- We also helped with many of these other MCW labs:
  - https://github.com/Microsoft?utf8=%E2%9C%93&q=mcw