Container Orchestrators

- K8s, AKS

Rajesh Kolla

Full-stack development ,Azure Architect

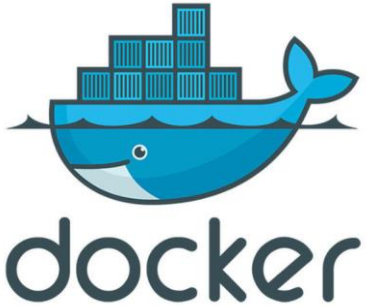
Email: rajesh.kolla01@outlook.com

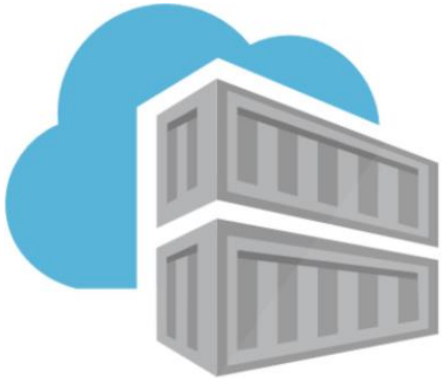
Twitter: @RajeshKolla18

LinkedIn: https://be.linkedin.com/in/razeshkolla

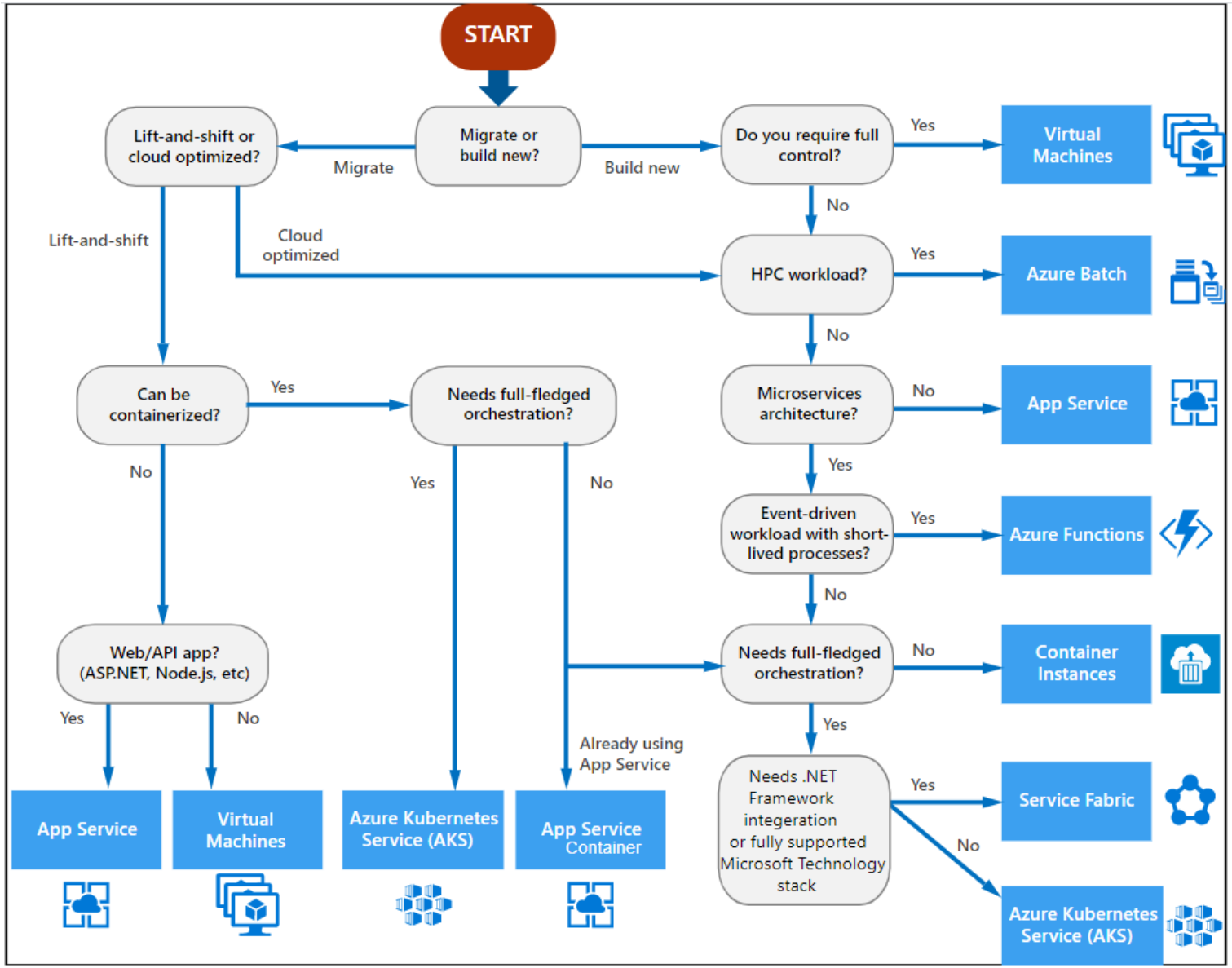
Agenda

- Why do need Orchestrator
- Kubernetes Overview
- Demo-
- Azure Kubernetes Service (AKS)
- Demos
  - Create an AKS Cluster
  - Deploy apps with kubectl
- Q&A





Decision Tree for Compute



Reference: https://docs.microsoft.com/en-us/azure/architecture/guide/technology-choices/compute-decision-tree

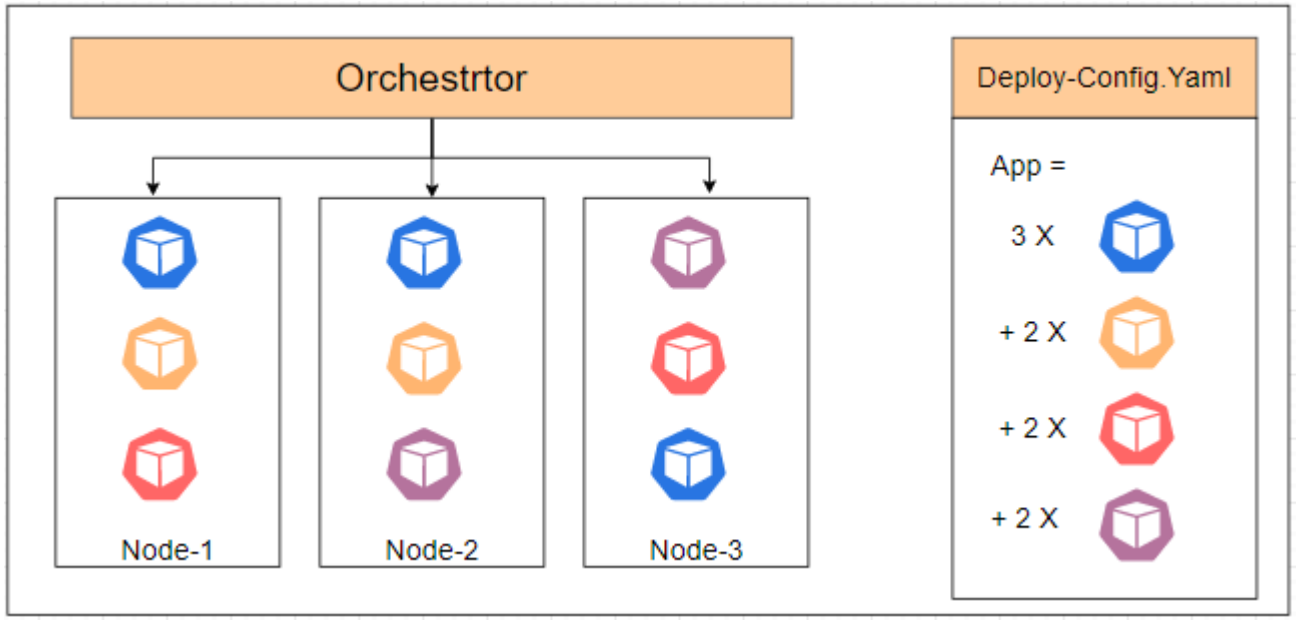
What is the orchestrator ?

Orchestrator is utility that is designed to easily manage complex containers and containerization deployments across multiple host from one central location .

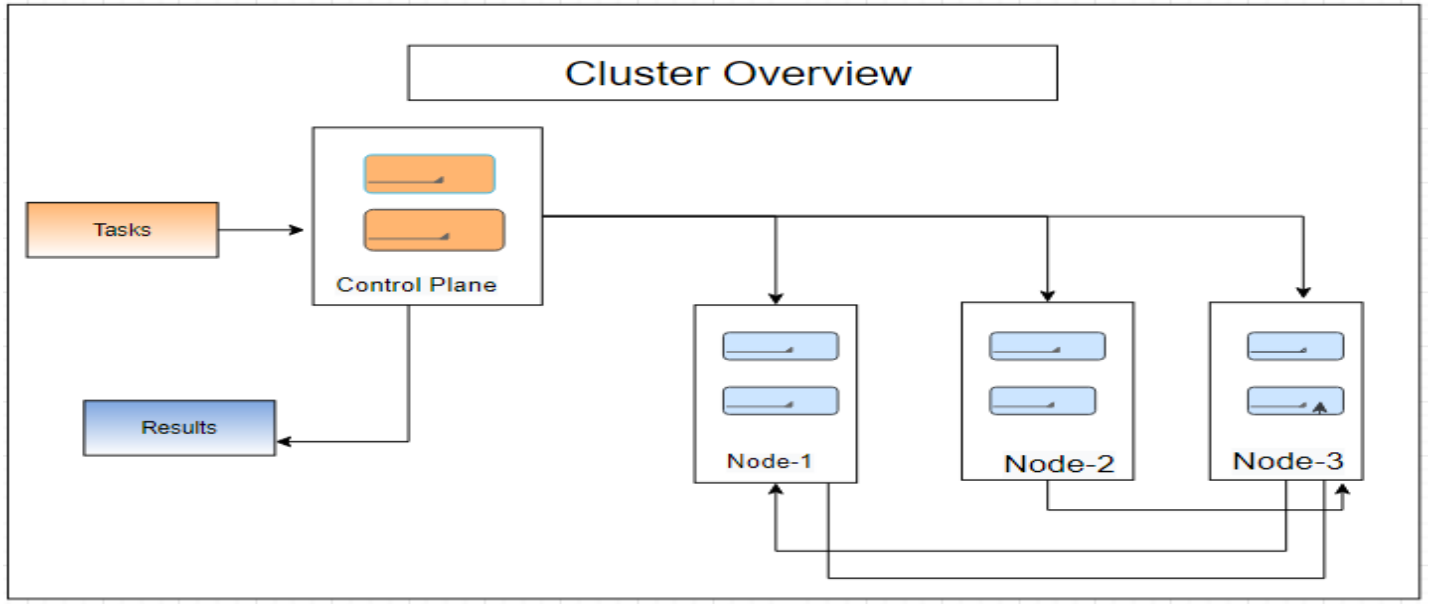
- Most popular Orchestrator :Kubernetes (K8s) and Docker- swarm

Why do we need orchestrator ?

- Self healing
- Autoscaling
- Health monitoring
- Upgrade strategies
- Resource constraints
- Networking
- Load balancer
- Service discovery
- Ingress



Kubernetes Overview



Kubernetes (K8s) is production level most popular orchestration system from Google.

Kubernetes (K8s) cluster contains

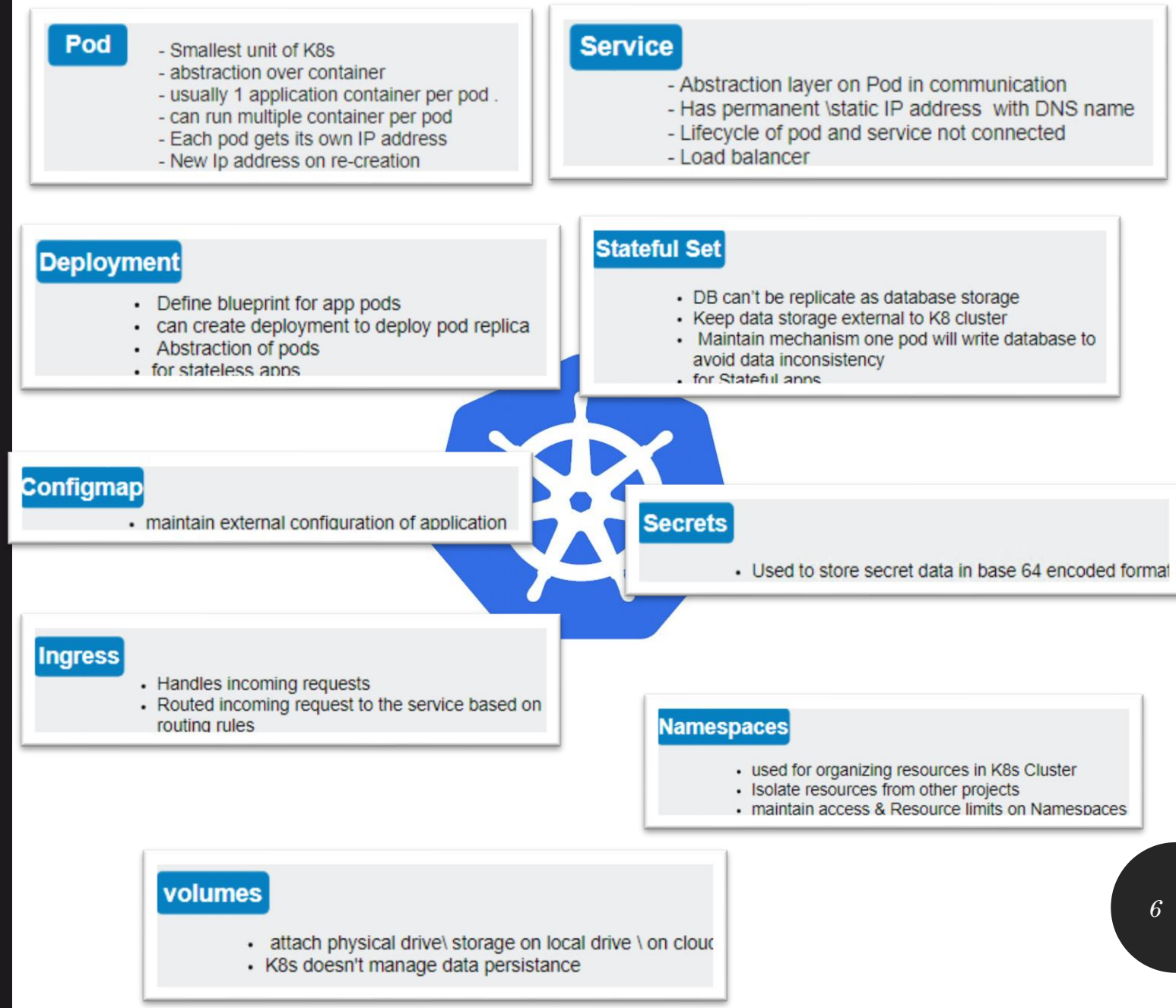
- Master Node- scheduler container
  - Master node needs less resources compare to worker node
- worker Node- running containers

- Default host OS in K8s is Linux for linux containers
- Can run windows container by windows 2019 or later

- Cluster is set of computers that configure to work together and view as one system
- A cluster uses centralized software which is responsible for scheduling and controlling tasks . Computers that run scheduling software is called control planes
- Computers in the cluster that run the tasks are called nodes
- K8s cluster contains at least one main plane and one or more nodes
- Both main plane &node can be physical devices , vms

K8s Components

- Pod
- Service
- Deployment
- StatefulSet
- Configmap
- Secrets
- Ingress
- Namespaces
- Volumes



K8s Architecture

1. Kube-Api-Server: This is cluster gateway and act as a gatekeeper for authentication . This is the only one entry point for cluster

2. Controller Manager:- This will detects cluster state changes. keep on monitor changes of pods in nodes . if any pod went fault state then notify scheduler about change

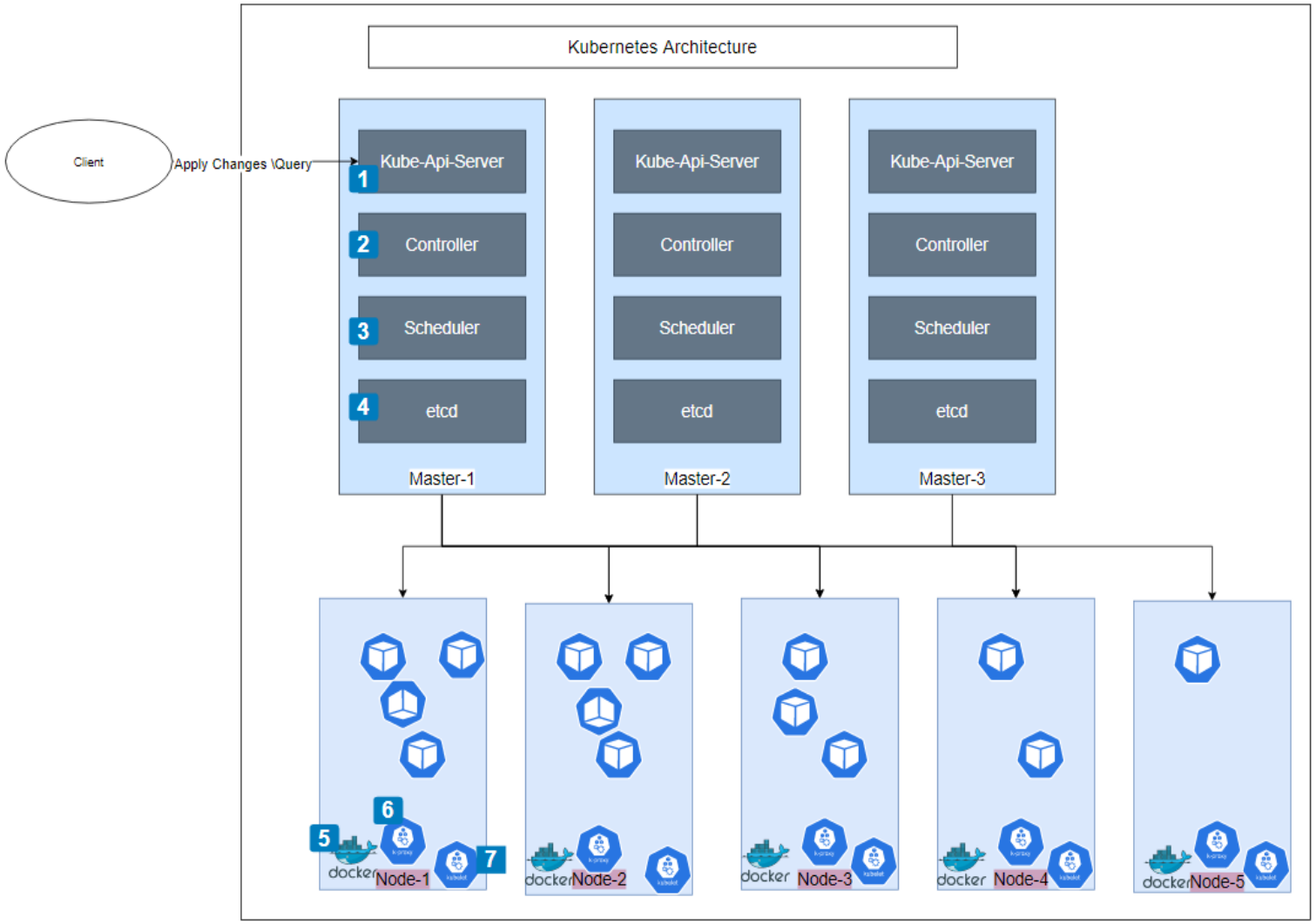
3. Scheduler:- This just decides on which node new pod should be scheduled based resource usage in node.

4. etcd:- This is internal storage
   - cluster changes get stored in the key value store
   - Manages Secrets and configmap
   - called it as cluster-brain

5. Container Runtime:- This is one of processor in worker node .
   - This will be used to run containers in pods

6. Kube-Proxy: This is one of processor in worker node
   - Communicate with other pods in the same node and other nodes
   - Forwards requests services

7. Kubelet: This is one of processor in worker node .
   - Interacts with both container and node .
   - Starts the pod with container inside.



How do we create test \dev K8 cluster ?

- MiniKube:- is an open-source project to create test \dev k8 cluster
- Docker Desktop:- can enable K8s local cluster using Docker Desktop in local DEV environment

How do communicate with K8 cluster?

Can communicate with K8 cluster using API , UI \dashboard , CLI (kubectl)

Kubectl:- is command line utility which enables communication with kube-Api-Server in K8s Cluster

Commands	description
kubectl version	To see k8s client & server version
kubectl get pod	List of running pods in default namespace
kubectl get replicaset	List of running replicasets in default namespace
kubectl get deployment	List of deployments in default namespace
kubectl get Service	List of services in default namespace
kubectl create deployment <Name> –Image=Image	Create deployment default namespace
kubectl describe pod <podName>	Describe details of pod in default namespace
kubectl logs <podName>	See logs of pod
Kubectl exec –it <podName> --bin /bash	Debugging pod
kubectl apply -f deployment-config.yaml	Deployment using deployment configuration file

Deployment configuration :-

- can use Yaml file for deployment of resources in K8s cluster
- Yaml is human readable declarative language and serializable
- Deployment file has 3 parts

Metadata

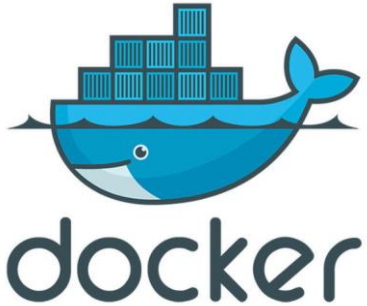
Specification

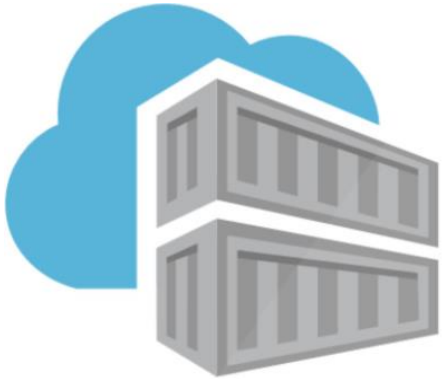
State
Desired?= Actual?

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginxapp
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginxapp
  template:
    metadata:
      labels:
        app: nginxapp
    spec:
      containers:
      - name: nginx
        image: nginx:1.16
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: 5050
```

Demo-1

Deployment resources
using Yaml files in local miniKube cluster
\docker desktop cluster

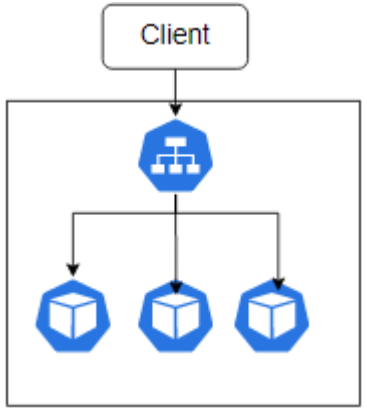




- Each Pod has its own IP address
- Pods are destroyed \created frequently

Service:-

- Stable IP address
- Load Balancing
- Abstraction layer to communicate to pods within & outside cluster



Types of Services :-

Cluster-IP Service

- It is default service, and it will generate internal IP address .
- Pods are identified via selector to forward the request
- target port much match the port container is listening at Cluster IP only accessible within cluster

Head less Service

Head-less Service

- There are some scenarios Pod wants to directly interact with specific pod
- Use case : Stateful applications like database
- Cluster-IP should be None
- Client needs to find out IP address using DSN lookup
  - DSN lookup for Service - returns single IP (Cluster IP)
  - when Set Cluster-IP to "None" - returns Pod IP address

Node Service

Node-Port Service

- can accessible outside cluster
- Port Range for Node-Port Services: 30000 -32767
- This is extension of Cluster-IP Service
- This is not secured as client directly access to node ip

Load-balancer Service

Load-balancer Service

- Becomes accessible externally through cloud providers load-balancer
- Load-balancer service is an extension of Node-Port Service

What is Helm :-

- Package manager for K8s
- Used to package YAML files for application and distribute them in public and private repositories in Helm Hub

Helm charts:-

- Bundle of YAML files
- Can create own Helm Charts with Helm
- Push them to Helm Repository
- Download and use existing ones

Helm Chart Structure

```
AppChart/
  Chart.yaml
   values.yaml
   charts/
   templates/
```

Top level AppChart folder – name of the chart

Chart.yaml – meta info about chart

Values.yaml - key & values for the template files

Charts folder -> chart dependencies

Template folder -> the actual template files

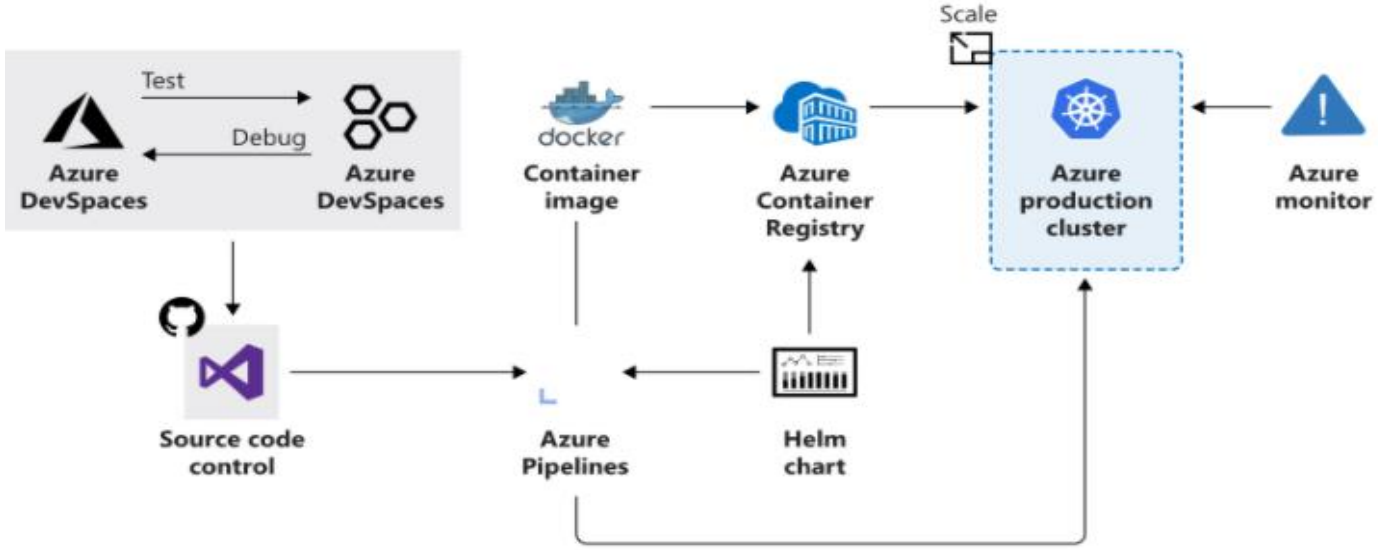
Helm install <chartname>

Helm install - -values=app-values.yaml <chartname>

Azure Kubernetes Services (AKS)

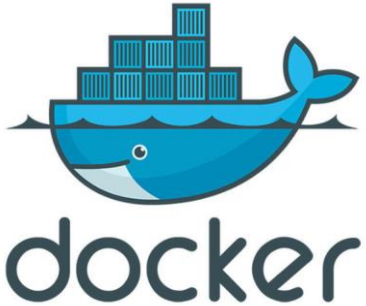
- *Managed Kubernetes cluster in azure*
- *Control plane is free*
- *Only pay for work nodes*
- *Simplified version upgrades*
- *100% upstream Kubernetes*

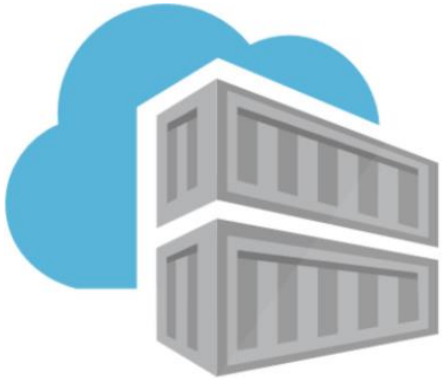
- Azure-Monitor monitor azure files or disks
- Identity and Security management with AAD
- Integrated logging and monitoring with Azure-Monitor
- Auto cluster node and pod scaling
- Virtual network integration
- Ingress with Http application routing support
- Integration with public & private container registry (ACR)
- Elastic scale with ACI
- Deployment Centre -Azure DevOps project automatically create azure resources and enable Azure monitor for container to monitor performance
- Develop and debug with Dev Spaces
  - Develop your code in isolation and do integrated testing with other components without replicating or mocking up dependencies



Demo-2

- AKS cluster creation

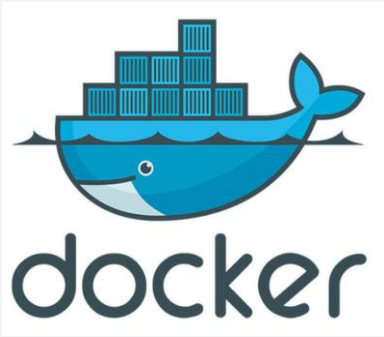


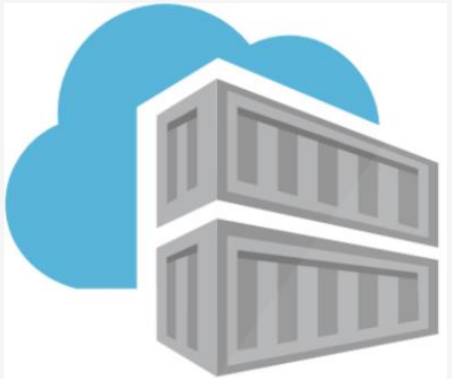


Demo-3

Deploy services in AKS Cluster

- kubectl





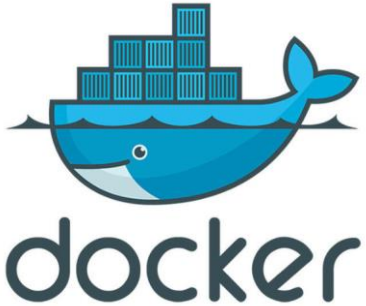
Demo-4

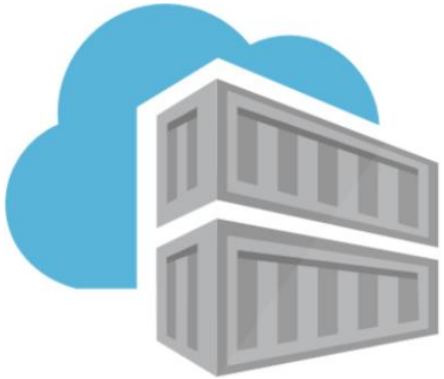
Scaling In AKS:-

Horizontal Pod scalar

Horizontal Node scaling

- kubectl





ASF vs AKS

- *Common features*
- *Scheduling*
- *Upgrades*
- *Health monitoring*
- *Service discovery*

Service Fabric:- Service Fabric is open-source technology it can be installed on Linux and windows and run on physical or virtual machine or cloud. Support running application in containers or native executables

Azure Service Fabric :- is a service in azure which assist with creating VMs, Networking , installing & configuring service fabric . It supports ARM templates to provisioning resources and manages SF cluster

- Windows
- Stateful services
- Serverless (Service fabric mesh)

- Open-source tooling ecosystem
- Other clouds
- Virtual kubelet
- Dev spaces

(it is going replaced by Bridge to K8s)

Q&A

Email: razesh.kolla@gmail.com
Twitter: @RajeshKolla18
LinkedIn: https://be.linkedin.com/in/razeshkolla

Thank you!

Email: razesh.kolla@gmail.com
Twitter: @RajeshKolla18
LinkedIn: https://be.linkedin.com/in/razeshkolla