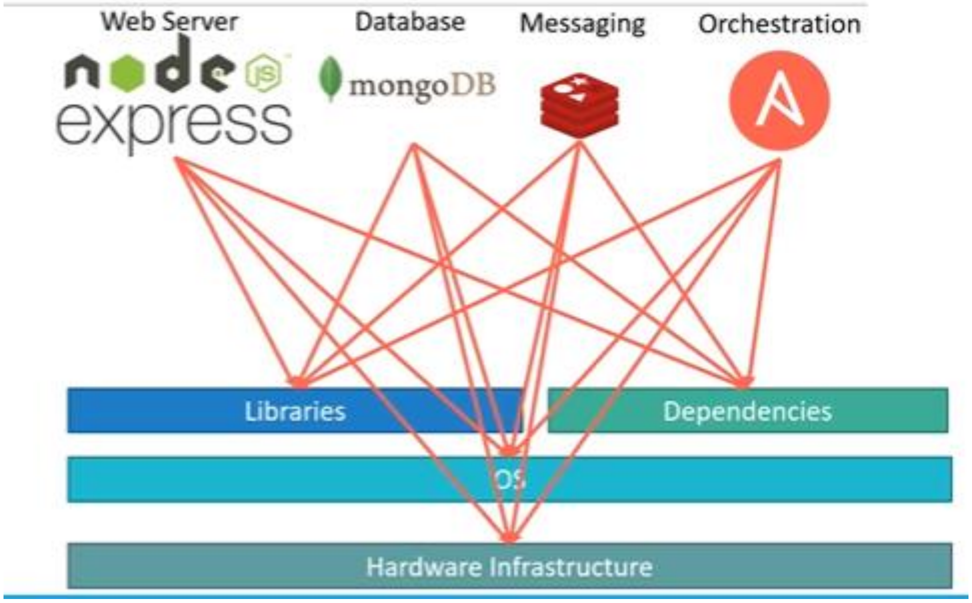
Agenda

1. Overview of Container
2. Azure Kubernetes Architecture
3. Introduction to Deployment & YAML Manifest File
4. Create aks cluster and identity in sandbox environment
5. Demo : Deploy a multicontainer application to AKS
6. Useful Resources

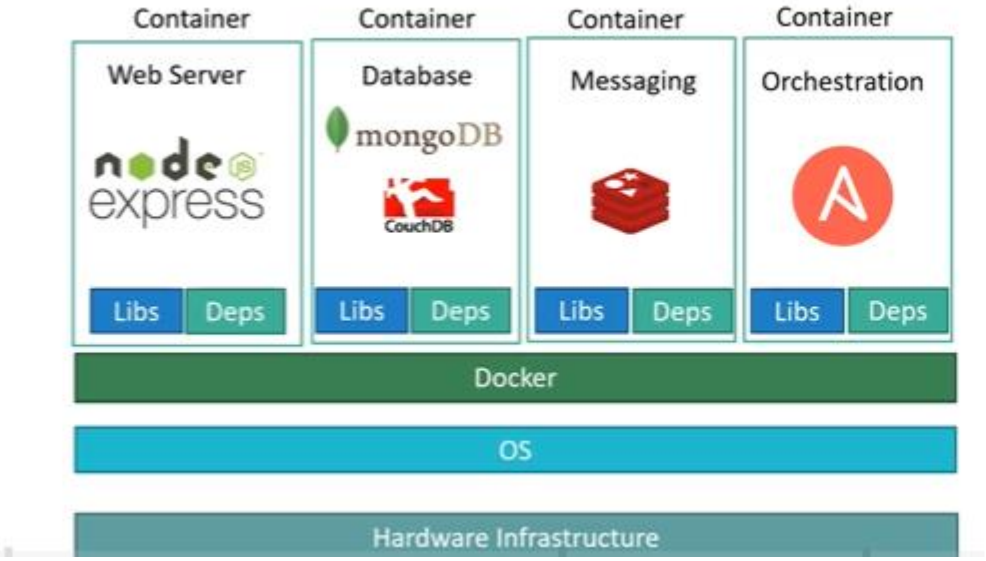
Overview of Kubernetes

- **Why we need container?**

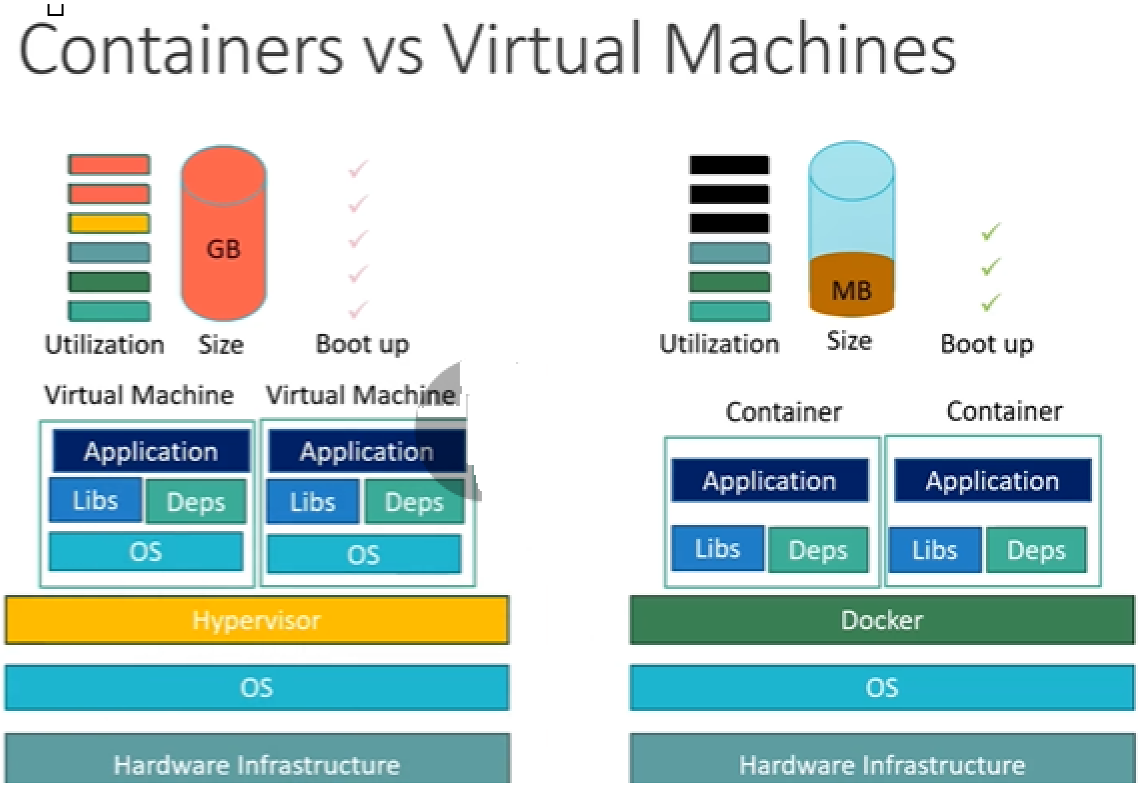


- Compatibility of different applications & packages with OS
- Different services requires different version of libraries
- Architectural changes
- Long setup time
- Different DTAP env

- **Solution is Containerized application**



- **Container Vs VM**



Container Vs Image

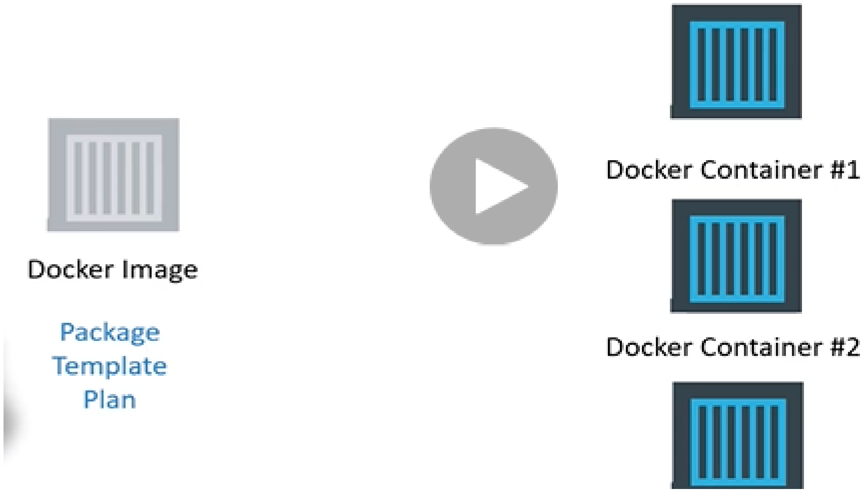


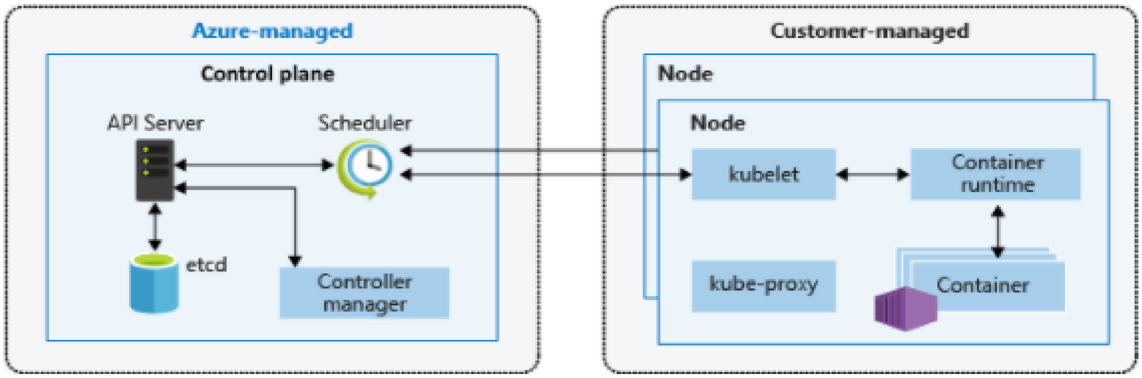
Image : Package or Template like we have different images in VM
Container : Images are used to create containers

Azure Kubernetes & Architecture

Kubernetes

- Open-source system for automating deployment, scaling, and management of containerized applications.
- [Azure Kubernetes Service (AKS)]() makes it simple to deploy a managed Kubernetes cluster in Azure.

Kubernetes Architecture



- Control plane

  - core Kubernetes services and orchestration of application workloads
  - Control plane is automatically created and configured during AKS cluster creation
  - Components of Kubernetes :

    1. *kube-apiserver* - Provides the interaction for management tools such as `kubectl` or the Kubernetes dashboard
    2. *etcd* - key value store to maintain the state of your Kubernetes cluster and configuration
    3. *kube-scheduler* - determines what nodes can run the workload during application creation or scaling and starts them
    4. *kube-controller-manager* - Oversees a number of smaller Controllers that perform actions such as replicating pods and handling node operations

Node

An AKS cluster has one or more nodes, which is an Azure VM that runs the Kubernetes node components and container runtime.

Node Pools

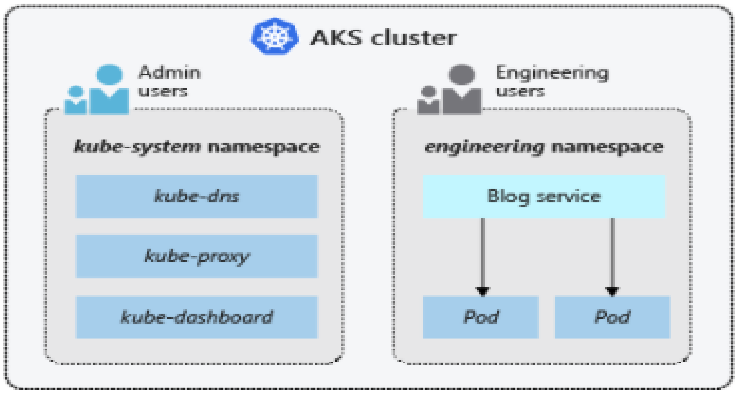
Nodes of the same configuration are grouped together into *node pools*.

Pods

A pod represents a single instance of your application. Pods typically have a 1:1 mapping with a container.

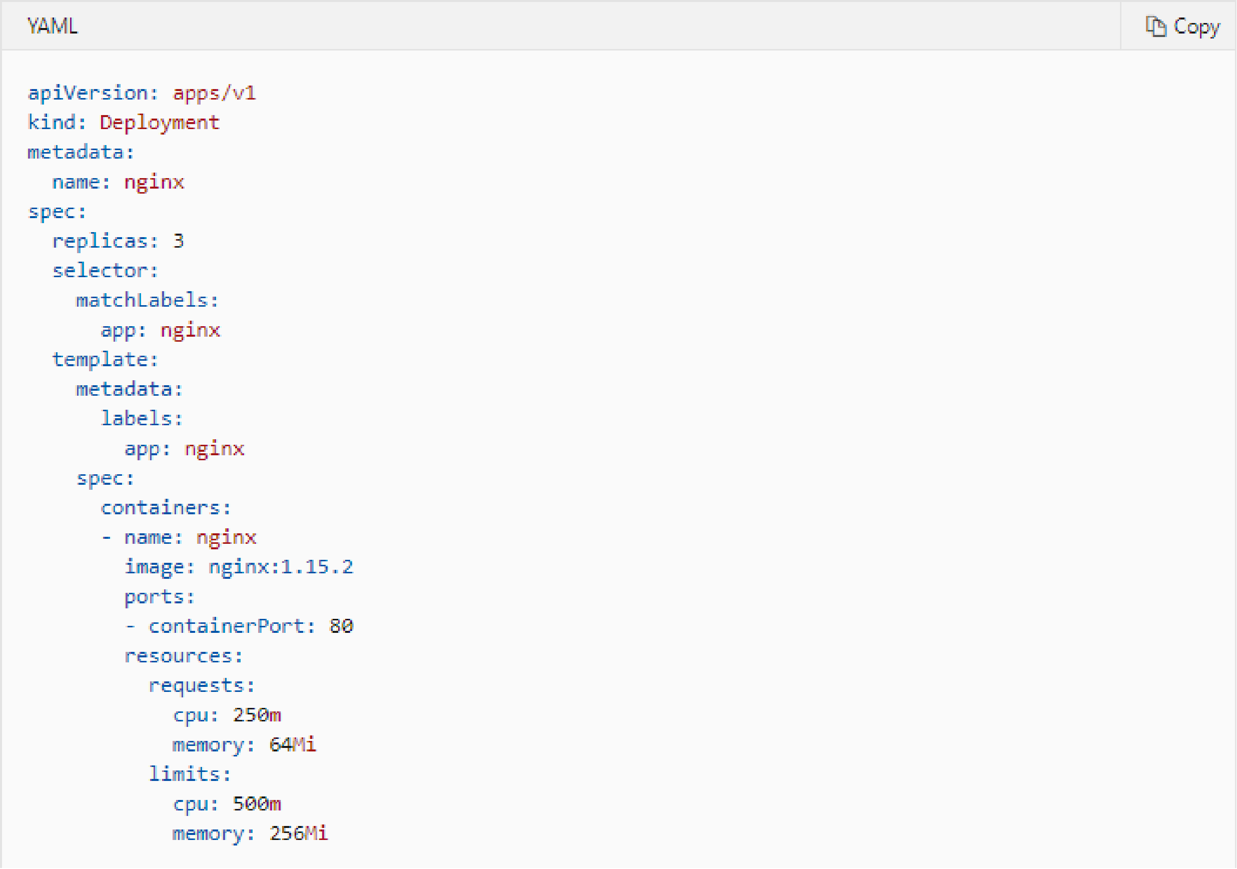
Package management with Helm

Namespace



Introduction to Deployment & YAML Manifest File

Deployments are typically created and managed with kubectl create or kubectl apply

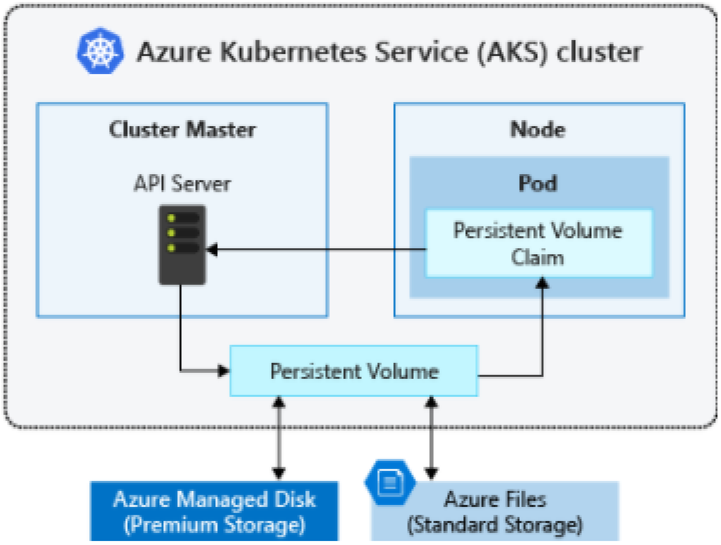


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.15.2
        ports:
        - containerPort: 80
        resources:
          requests:
            cpu: 250m
            memory: 64Mi
          limits:
            cpu: 500m
            memory: 256Mi
```

Deploy application on AKS with Helm

1. Azure Container Registry to store your container images
2. Create AKS Cluster
3. Connect to AKS Cluster
4. Download your application
5. Create a Docker File
6. Build & Push Application ACR
7. Create Helm Chart
8. Run your helm chart

- **Storage with AKS**


Azure Kubernetes Service (AKS) cluster
Cluster Master
API Server
Node
Pod
Persistent Volume Claim
Persistent Volume
Azure Managed Disk (Premium Storage)
Azure Files (Standard Storage)

Useful Resources:

- Monitoring : https://www.youtube.com/watch?v=RjsNmapggPU&t=935s
- Beginners : https://www.udemy.com/course/learn-kubernetes/
- AKS Workshop Microsoft : https://docs.microsoft.com/en-us/learn/modules/aks-workshop/
- Udemy AKS : https://www.udemy.com/course/azure-kubernetes-service-aks/
- AKS onboarding in ABN AMRO : https://confluence.aws.abnamro.org/pages/viewpage.action?spaceKey=COESD&title=Getting+Started+Guide+on+AKS