# Basics of Container

VM:

Install java

Maven package

Copy jar

Java -jar app.jar

Node app:

1. Source code
2. Dockerfile
3. Docker build -> container image
4. Container (docker run -dt node-app)

Source code(developer) -> build(jar) -> **Dockerfile** -> Container Image -> Container

docker <operations>

docker images

docker <module> <operation>

docker image ls

# Kubernetes Architecture

AWS

User Interface(UI) - Management Console

Command Line Interface(CLI) - Aws cli

Application Programming Interface(API) - Terraform, Cloud Formation

Kubernetes(k8s)

User Interface(UI) - k8s Dashboard, Rancher, Lens(IDE)

Command Line Interface(CLI) - kubeadm, kubectl

Application Programming Interface(API) - Terraform

Install Kubeadm, kubectl:

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#installing-kubeadm-kubelet-and-kubectl>

Container Runtime Interface (CRI)

Docker, **containerd**, podman, rkt

Container Networking Interface (CNI)

**Calico**, Flannel, Weavenet

kubeadm init - Stages

[preflight] Pull image:tag = apiserver:1.28.2 - CR - registry.k8s.io

certs

configuration

API Server -> etcd / scheduler

etcd hostname / username / certificate

control-plane -> kubelet

API Server

Controlplane - API Server, CM, Scheduler, Etcd

# 

## Upgrade K8S cluster:

Cluster ->

* on-prem/non-cloud(control-plane owned)
* cloud(service provider manages control-plane) - AKS

Non-cloud Cluster:

1) Pick a version

1.28.14-2.1

2) Upgrade control-plane

**Kubeadm**

sudo apt-mark unhold kubeadm

sudo apt-get update && sudo apt-get install -y kubeadm='1.28.14-2.1'

sudo apt-mark hold kubeadm

**Kubectl & Kubelet**

sudo apt-mark unhold kubelet kubectl

sudo apt-get update && sudo apt-get install -y kubelet='1.28.14-2.1' kubectl='1.28.14-2.1'

sudo apt-mark hold kubelet kubectl

# Pod

Pod Status

1/1

Ready Containers / Total no. of Containers

Init:0/2

Completed Containers/Total no. of Containers

## Auto Scaling

VM:

Launch Template:

VM -> OS, Size, User script, Volumes

Auto Scaling:

Min node = 3 and max nodes = 5

Metrics -> CPU, mem, Network traffic

If cpu >80%; create a VM

If cpu <80%; Delete a VM

Load Balancer:

User -> LB -> Any one of the pod

## Deployment

Kubernetes

Pod Template -> container name, size, image etc..

Deployment

Rolling Update (default) - Canary Deployments

Recreate- Blue/Green Deployments

1 -> nginx:1.14.2

2 -> nginx:1.16.1

3 -> nginx:xxxx

undo = 3-1 = 2

1 -> nginx:1.14.2

3 -> nginx:xxxx

4 -> nginx:1.16.1

Undo = 1

# Configuration

Container

Container Image:tag

Configuration (based on env)

Env

ConfigMap

Secret

V1 -> 10 Tables in DB

V2 -> 15 Tables in DB (5 New tables added)

V1 -> Update DB with 5 new tables(Job) -> V2 (set image)

Job - 1 tasks gets executed once

Pod -> Container -> Command: mysql -u -p migration.db -> Pod gets completed

## 

## CronJob:

schedule: every Sunday

Job - 1 tasks gets executed once

Pod -> Container -> Command: mysqldump -u -p backup.db -> Pod gets completed

1st Sunday

Job 1 -> Pod 1.A

2nd Sunday

Job 2 -> Pod 2.A

Example: MySQL Database Backup using CronJob

apiVersion: batch/v1beta1

kind: CronJob

metadata:

name: backup

spec:

schedule: "@daily"

jobTemplate:

spec:

template:

spec:

containers:

- name: mysql

image: mysql

imagePullPolicy: IfNotPresent

command:

- /bin/sh

- -c

- mysqldump drupal-db -p$MYSQL\_ROOT\_PASSWORD > dump.sql;

- date; echo Drupal database has been backed up

restartPolicy: OnFailure

Metaverse: Dev Cluster [10 workerNodes - 100Gb Mem]

facebook - 10 deployments, 2 ds, 3 cm, 2 secrets, 10 svc

instagram

whatsapp

meta-pay

Kube-system

# Scheduler

1. Node name
2. Group of nodes - specific config

Node 1

Node 2

Node 3

Node 4

disk=ssd

Node 5

disk=ssd

1. Pod -> Specific Availability Zone
2. Creating pods in Master node

Flight Booking System:

MUST:

Leg Room

MAY:

1. Window Seat
2. Near to restroom

Seat == Node

Passenger == Pod

Affinity

if condition is TRUE -> Pod get created

condition(node selectors) -> Node Affinity

condition(pod selectors) -> Pod Affinity

Anti-Affinity

if condition is TRUE -> Pod should not get created

condition(pod selectors) -> Pod Anti-Affinity

Node Affinity

Required(must)

network=fast

Preferred(may)

disktype=ssd

Node1

network=fast

disktype=ssd

Node2

network=fast

**Goal:**

| **node-1** | **node-2** | **node-3** |
| --- | --- | --- |
| *webserver-1* | *webserver-2* | *webserver-3* |
| *redis-cache-1* | *redis-cache-2* | *redis-cache-3* |

Architect’s condition:

**Goal**: Dedicated redis-cache for each webserver

1) Colocate webserver and redis-cache [pod affinity]

2) No identical pods on same node [pod anti-affinity]

Node 1

Redis

webserver

Node 2

Node 3

Node 4

Node 5

# Helm

Package? == Charts

Oct 24

facebook charts - 10 deployments(fb:15.2), 2 ds, 3 cm, 2 secrets, 10 svc

v1.0

Jan 25

facebook charts - 10 deployments(fb:24.6), 5 cm, 5 secrets, 10 svc

V1.5

Chart(package)

Tools

Application

# Networking

Types of Network:

1. Management or Cluster Network: 172.31.
2. Pod Network: 172.16.
3. Service Network: 10.96.0.0/12

Container Network Interface(CNI)

Calico

Pod CIDR - 172.16.0.0/16

Calico-node - k get ds calico-node -n kube-system

Calico-controller - k get deploy calico-kube-controllers -n kube-system

IPAM - IP Address Management -> Pod IP

Service

CIDR - 10.96.0.0/12

Defined -> sudo cat /etc/kubernetes/manifests/kube-apiserver.yaml | grep service-cluster-ip-range

kube-proxy -> IPAM and Config(svc)

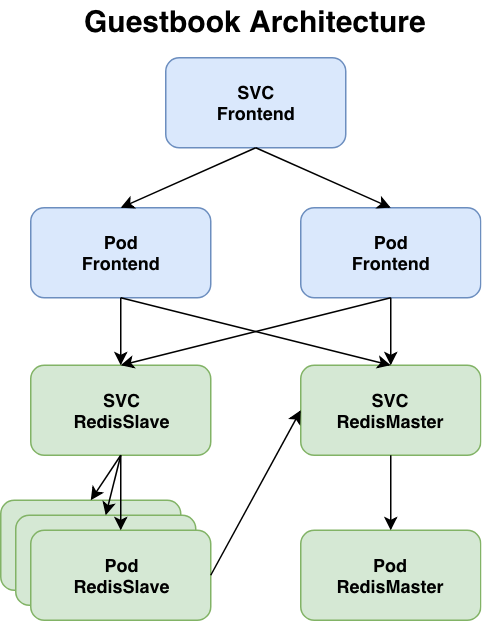
k get ds kube-proxy -n kube-system

**Cloud**:

User -> LB -> AutoScaling [EC2-X, EC2-Y, EC2-Z]

**Kubernetes**:

User -> Service -> Deployment [pod-x, pod-y, pod-z] + HPA



**Service Types**:

Cluster-IP (default) - Internal LB - Internal Traffic(within Cluster)

example: Redis, DB, kafka, MessageQueues

NodePort - External LB - Organisation

employee -> Intranet(frontend)

example: JIRA(30022), SCM/git, Jenkins

Range: [30000 - 32767] ~ 2,767

LoadBalancer - External Traffic (Internet) [AKS]

example: Amazon

frontend - frontend-58cf9cd597-4klw4 - 172.16.232.201

web - nginx-deployment-86dcfdf4c6-2qckc - 172.16.232.199

Ingress Controllers

Nginx Ing Ctrl  
 Traefik

Istio (Service mesh)

Route:

localhost:30493/cart -> Cart svc

/accounts -> accounts svc

/orders -> orders svc

/redis -> ? <404 Not Found>

# Role Based Access Control (RBAC)

Permissions (Labs):

Start

Stop

Reset

Extend

Role = Resource + Permissions

Learner = Lab + {Start, Stop}

Trainer = Lab + {Start, Stop, Reset}

LSM = Lab + {\*}

Support = {\*.\*}

RoleBinding = Role + User/Group/ServiceAccount(SA)

Swami = Trainer + G(ACT-2024)

Support + G(simplilearn-support)

LMS

Login -> Authentication

Access -> Authorization (RBAC)

**Kubernetes**

Permissions: Verb [get, list, watch, create, update, patch, delete]

Create

Read

Update

Patch

Delete

Role = Resource + Permissions + Namespace

Developer/QA = [Pods, Deployments + get, list, update, watch] + act

DevOps = [\* + get, list, watch, update, patch, delete] + act

Support = [\*.\*] + act

RoleBinding = Role + User/Group/ServiceAccount(SA) + Namespace

Developer + u(Swami) + act

DevOps + g(devops) + act

Support + g(cka\_devops) + act

Sundar -> devops

Sundar -> cka\_devops

[\* + get, list, watch, update, patch, delete] + [\*.\*] = [\*.\*]

Application Onboarding into Kubernetes Cluster:

1. Create a Namespace
2. RBAC - Roles & RoleBinding -> NS
3. Resource Quota -> NS
4. Create workloads in NS

Cluster Role & Cluster RoleBinding

Cluster Role = Resource + Permissions

Cluster RoleBinding = Role + User/Group/ServiceAccount(SA)

Cluster Admin = \*(resources).\*(permissions) + \*(ns)

# Storage

Steps for PV

1. Create Volume - PV - Storage Admin
2. Claim Volume - PVC - Namespace
3. Use Volume in Pod - Pod - Namespace

Medium - 30 CM

XL - 40 CM

My request - 35

# Resource Quota

Metaverse: Prod Cluster [10 workerNodes - 100Gb Mem]

facebook - 25gb Mem

Instagram -

Whatsapp - 10

Meta-pay - 5

Kube-system - 5

# Certification - CKA and CKAD

Black Monday Sale - December

Actual cost - $395 [Sale: 50- 60% - $150]

12th Dec 2024 - Registration $395

14th May 2025 - 1st Attempt

Outcome = Pass - CKA/CKAD OR not clear = 2nd Attempt

14th Sep 2025 - 2nd Attempt

11th Dec 2025 - Deadline [1 year]

Validity - 2 years

!!! Certification revamped in Jan 2025

Exam:

Remote Desktop

17 Questions

2 Hours

!!! 2 Simulation - killer.sh

**Certified Kubernetes Administrator:** [**https://www.cncf.io/certification/cka/**](https://www.cncf.io/certification/cka/)

**Certified Kubernetes Application Developer:** [**https://www.cncf.io/training/certification/ckaD/**](https://www.cncf.io/training/certification/ckaD/)

**Exam Curriculum (Topics):** [**https://github.com/cncf/curriculum**](https://github.com/cncf/curriculum)

**Candidate Handbook:** [**https://www.cncf.io/certification/candidate-handbook**](https://www.cncf.io/certification/candidate-handbook)

**Exam Tips:** [**http://training.linuxfoundation.org/go//Important-Tips-CKA-CKAD**](http://training.linuxfoundation.org/go//Important-Tips-CKA-CKAD)

**Kubectl Man page** - <https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#-strong-getting-started-strong->

**Kubectl Cheatsheet** -

https://kubernetes.io/docs/reference/kubectl/quick-reference/