**Kubernetes(K8s)**

Importtent Link

1. <https://www.simplilearn.com/tutorials/kubernetes-tutorial/kubernetes-interview-questions>

**What is Kubernetes**: Kubernetes  is an open sorces container orchestation software which devloped by google & released on 21th July 2015. Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services—with a framework to run distributed systems resiliently. It takes care of your scaling requirements, failover, deployment patterns, scaling, load balancing, logging, and monitoring, much like PaaS offerings. However, it operates at the container level rather than at the hardware level.

Popular feature of Kubernetes

1. **Pods:** Pods can have one or more containers coupled together. They are basic unit of kubernetes. To increase High Avilability, we alsway prefer pods to be in replicas.
2. **Replication Container:**
3. **Rolling Update:**
4. **Storage Management:**
5. **Resources Management:**
6. **Health Checks:**
7. **Services Discovery:**
8. **Networking:**
9. **Secret Management:**

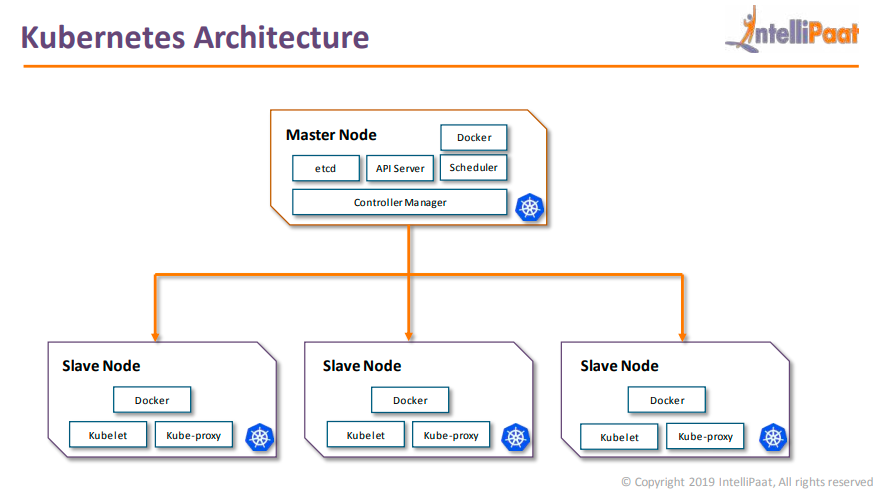
Major Topic Cover:

1. **Archetucture**
2. **Installtion**
3. **Deployement**
4. **Services**
5. **Ingress**
6. **DashBoard**

**Diff B/w Docker Swarm and Kubernetes**

1. Docker is easy to install & insitialize as k8s is complex procedure to install
2. Docker faster when compared to k8s
3. Docker is npt reliable and has less features as compare to k8s

**Kubernetes Archetucture:** K8s archeture components like below



1. **Master Node**
   1. **Etcd**: It is a highly avilable distributed key value store, which is used to store cluster wide secrets. It is only accessible by kubernerts API server as it has sensitive informtion.
   2. **API Server**: It exposes the kubernetes api. The k8s API is the fron-end for k8s Control Plane, and is used to deployed and execute all oprations in k8s
   3. **Scheduler**: The scheduler takes care of scheduling of all the process, Dynamic resources Management & manages present & future event on the cluster.
   4. **Controller Manager**: The controller manager, run all the controller on the k8s cluster. Although each controller is a seprate process but to reduce complexity, all the controller are compile into a single process. They follows: Node Controller, Replication Controller, Endpoints Controller Service Accounts & Token Controller.
2. **Slave Node**
   1. **Kubelet:** Kubelet takes the specification from the Api server, and ensures the application is running according to the specifition which were mentioned. Each node has it’s kubelet service.
   2. **Kube-Proxy**: This proxy run on each node and help in making services available to the external host. It help in connetion forwarding to the correct resources, it is also capable of doing primitive load blancing.

**Kubernetes Installtion**

There is numbers of way to install the k8s like below

1. **Kubeadm**: Bare Metal Installtion
2. **Minikube**: Virtualized Envirroment for k8s
3. **Kops**: K8s on AWS
4. **Kubernetes** on GCP: K8s on GCP

We will learn k8s through Kubeadm:---

1. **1st We need to lunch three(3 AWS Instance or VM, Name like: K8Master, K8Slave1, K8Slave2) aws instance(generally know as 3 cluster node)**

1.1 => 1-Master => Try to keep t2.mudum (Depand on requirement but try to keep it atleast t2.midum)

1.2 => 2 Slub/Node machine => Try to keep t2.micro

2**. Connect All 3 instance and run belo command**

sudo apt-get update

3.1 **Install The Kudeadm on all 3 instance and this 1st create one file through nano with .sh extention like and pest all below command inside the file same it**

sudo nano MyInstalltionFile.sh

3.2 **Copy & pest below command in newaly creaded file**

sudo apt update -y

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update -y

sudo apt install kubeadm=1.20.0-00 kubectl=1.20.0-00 kubelet=1.20.0-00 -y

3.**3 Run the below command for run & install all packages/command which are inside the MyInstalltionFile.sh file**

bash MyInstalltionFile.sh

4. **Master Node : Run only in master**

4.1 **Chnage the HostName**

sudo su

hostnamectl set-hostname master

exec bash

**5 kubeadm: run below command in only master**

5.1 **kubeadm init through super user (this will generate the join token command which needs to be executed on the worker/slave node)**

kubeadm init

5.**2 run as normal user not super**

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

5.**3 Deploy CNI in master**

kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml

5.4 **Go To your worker/slave machine login as super user and pest the complete tocken which generate on master/main machine**

5.4.1 sudu su

5.4.2 cd

5.4.3 kubeadm join x.x.x.x --token dqrofv.t1y8jr6odiez5das --discovery-token-ca-cert-hash sha256:150....b65

6.**1 run this command in master for Watch connected slave/worker machine**

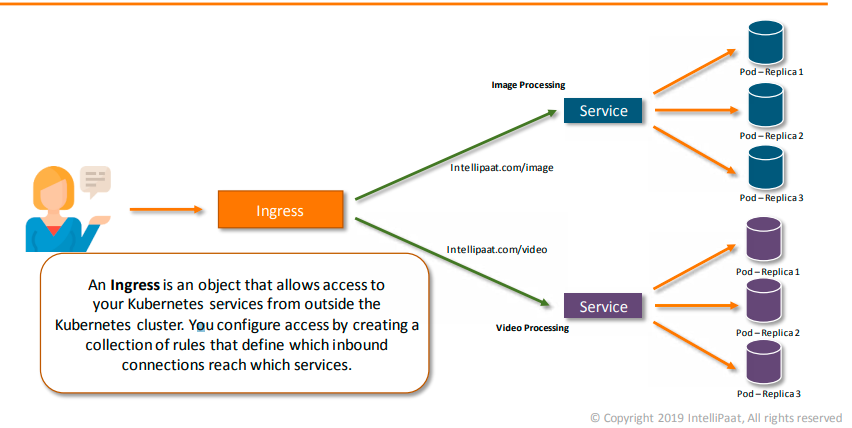
kubectl get nodes -w

**Kubernetes Deployement** :

**Kubernetes Services** : Servcices are used to load balance the traffic among pods. It folows round robin ditribution among the healthy pods. A service is basically a round-robin load blancer for all the pods, which match with it’s name or selector. It constantly monitor the pods, in case a pod gets unhealthy, the service will start deploying the traffic to the others healthy pods. Type of Services

1. **ClusterIP**: Exposes the service on cluster-internal IP
2. **NodePort**: Expose the service on each node’s IP at a static IP
3. **LoadBalancer**: Expose the service externally using cloud provider’s load blancer.
4. **ExternalName**: Mpas the services to the contents of the externalName

**Kubernetes Ingress**: An Ingress is an object that allowed access to your services from outside the k8s cluster. Your configure access by creating a collection of rule that define which inbound connection reach which service. Means with the help of single ingress configuration we can handle multiple services by identified the incoming request/url/port.



**Kubernetes  DashBoard**: It is web based dashboard k8s user interface. You can use dashboard to deployed containerized application to a k8s cluster, troubleshoot your containerised application and manage the cluster.

kubectl get nodes -o wide

kubectl describe pod