**1.What is Kubernetes:**

* **K8s is an Orchestration Engine (it will manage containers)**
* **It is also called as k8s**
* **It is an Open-Source Platform**
* **It is used to manage Containerized apps**
* **Managing Means Scale up, down , Recreate , delete the container .**
* **K8s developed in Google Company using GO Language.**
* **Google donated K8s to CNCF in 2014**
* **K8s first version v1.0 released in the year of 2015**
* **K8s is replacement for Docker Swarm.**

A screenshot of a computer

Description automatically generated

**2.Difference between docker Swarm And Kubernetes**

|  |  |
| --- | --- |
| **Docker Swarm** | **Kubernetes** |
| **Docker Swarm is also used for Orchestration** | **Kubernetes is also used for Orchestration** |
| **When we go for Docker Swarm Auto Scaling is Not available In that Docker swarm we need to scale up and scale down** | **In Kubernetes Auto Scale capabilities is available Based on the Incoming traffic your containers has been scale up and scale down** |
| **Good Community Support is available for Docker Swarm** | **K8 Great active Community , Because Kubernetes is having lot of demands in the market** |
| **Setting up the docker swarm cluster is easy Because docker swarm is embedded in Docker engine** | **In k8's Setting up the cluster is difficult process.** |
| **Limited to the Docker API's Capabilities Only Docker engine is used to create the containers** | **Whereas K8 can be used with docker and along with other containerization can be used in k8** |
| **Docker Swarm in not recommended in Production Deployment** | **Whereas K8 is recommended for the Production Deployment** |

**Containerization and Orchestration both are different.**

1. Containerization is executing an application.
2. Orchestration means managing delete, recreate, scale up, scale down the containers.

**Kubernetes ,Docker Swarm are an Orchestration Platform**

* **Docker is a Containerization tool.**
* **Docker will package our application as image.**
* **Using image, we can create containers.**
* **Inside Containers our application will execute**
* **We can create docker containers in any machine using docker images.**
* **Instead of we are creating and managing containers we can used Kubernetes tool.**
* **K8s is Orchestration Tool**
* **Orchestration means Management.**

**3.What is Cluster?**

* **Kubernetes will follow Cluster architecture**
* **Cluster mean is always Group of servers; More than one server will be available.**
* **In the Cluster Always Master Node and Worker Nodes will be available for high availability.**
* **Devops Eng / Developer will give the task to K8s Master Node.**
* **Master Node will manage the Worker nodes**
* **Master Node will schedule the task to worker nodes**
* **Our containers will be created in worker nodes**

**Note: Docker Swarm and Kubernetes both are following the Cluster architecture.**

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer

Description automatically generated**Control Plan -> it is the heart of Kubernetes, which is play very important role, which is nothing but master node.**

**Etcd -> 1. it is internal database in the K8 cluster  
 2. ETCD is a IN memory in the k8 cluster, we can call as k8 database.**

**API server -> 1.It is used to receive the incoming request.  
 2.We will send the request to two ways either Kubectl or UI.  
 3.When we send the request to control node, API Server will receive the request , then immediately API server will store the request information in the etcd as a pending request.**

**Schedular -> 1. Schedular will go to etcd , will identify which request are pending in the etcd   
 2.Schedular is responsible for scheduling the pending task.  
 3. Then schedular will identify any pending request are available in the etcd Memory  
 4.Schedular will go to worker node and talk to worker node  
 5.How Schedular will talk to worker node ?  
 6.By using Kubelet , Kubelet will call as worker node agent**

**Kubelet -> 1. Kubelet will call as node agent/we can called as personal assistance for the node.  
 2.Kubelet will understand the worker node details.  
 Example: Like what is the health of worker node, how many applications are running in the worker node , how many containers are running in the worker node like all the details are maintained by Kubelet in the worker node.**

**3. By taking with all the worker node Agent (Kubelet), Schedular will understand which worker node is available to run our requesting application  
 4.So Schedular will communicate with Kubelet and schedular will understand the worker node status  
 5.Based on that schedular will schedule this application should be deploy in the first worker node, second worker node / container should be deployed in the first worker node, second worker node like that schedular will schedule our container creation on the worker node by getting the information from Kubelet.**

**Controller Manager -> 1. Once this information is decided by the schedular , then controller Manger is going to perform the operation which is schedule by the schedular  
 2. What controller manager is going to do , it is going to control our containers creations**

**Note: Docker we can create the containers directly. But in K8 we cannot be created directly. In the k8 there is a concept call POD. Every Container will be created inside the POD**

**POD -> 1.It is a Smallest Building Block in K8. Inside the pod container will be created . Inside the container application will be available.  
 2.In Work node multiple Pods will be available. In one Pods Multiple containers will be available.   
 3. As we run the container inside the pod, we need Docker engine to run the container**

**Docker Engine -> In Order to Create the Containers . In order to execute the containers . we need a docker engine, So in every worker node Docker engine will also available. Docker engine will used to create those containers and run those containers.**

**Q)What is Difference between Pod and Container?**

**1.One Pod will have contained Several Containers**

**2.Containers is nothing but like box , to execute the application.**

**Kube Proxy -> 1.It is used to establish network communication between Control plan and worker node  
 2.Control Plan will be available in one machine and worker plan available in another machine. Communication will be happened by using kube proxy.**

**4.Kubernetes Architecture**

* **Control Plan / Master Node / Manage Node**
* **Api Server**
* **Schedular**
* **Control Manager**
* **ETCD**
* **Worker Master Node**
* **Pods**
* **Containers**
* **Kubelet**
* **Kube Proxy**
* **Docker Runtime.**

**5.How to Communicate with K8s control Plan?**

* **Kubectl (CLI tool)**
* **Web UI Dashboard**

**Brief about K8 Architecture Components:**

**API Server: It is responsible to handle incoming request of control plane.**

**ETCD: It is an internal database k8 cluster, API server will store request/tasks info in ETCD**

**Schedular: It is responsible for schedule pending tasks available in ETCD. It will decide in which worker node our task should be executed . Schedular will decide that by communicating with Kubelet.**

**Kubelet : it is worker node agent . It will maintain all the information related to worker node**

**Controller-Manager: After scheduling completed , Controller-Manager will manage our task execution in Worker node which is decided by the schedular**

**Kube-Proxy: It will provide network for K8s cluster communication ( Master Node <-> Worker nodes)**

**Docker Engine: To run our container docker engine is required . Containers will be created in the worker node.**

**Container: It is run time instance of our application**

**POD: It is the smallest building block that will create in k8 to run our containers.**

**Kubernetes Cluster Setup:**

**Kubernetes Cluster we can setup in Multiple ways:**

**Self- Managed k8s Cluster:**

* **Kubeadm (Multi Node Cluster)**
* **Mini Kube ( Single Node Cluster) - for Practice Purpose**

**Managed K8s Cluster:**

* **AWS EKS**
* **Azure AKS**
* **GCP GKE**
* **IBM IKE**

**Self-Managed K8s : We need to setup the Cluster an our Own, Take the VM or Physical Machines and setup the cluster an our own. We need to manage that cluster**

**Managed K8s Cluster: Cloud Providers will provide Cluster for rent . we no need to manage the Cluster an our own when we go for manage k8 Cluster. Charges will be applicable hourly based.**

**What is POD?**

* **POD is a Smallest building block in K8s Cluster**
* **In k8s, every container will be created inside POD**
* **POD always run inside Node**
* **POD represent the running process.**
* **POD means group of containers running on the Node**
* **We can create Multiple pods on single node**
* **Every POD will have unique IP addresses**

**We can Create PODS in 2 ways**

1. **Interactive Approach**

**$ kubectl run arun --image=nginx**

1. **Declarative Approach (Manifest YAML)**

**K8s Manifest YML they are four section like below  
---  
apiVersion:**

**kind:**

**metadata:**

**spec:  
…**

**Example - Kubernetes POD Manifest YML**

**---  
apiVersion: v1**

**kind: Pod**

**metadata:  
 name: javaapppod  
 labels:  
 app: javaapp**

**spec:   
 containers:  
 - name: cont1  
 image: httpd  
 ports:  
 - containerPort: 80**

**…**

**Note: Labels is nothing but unique identifier to identify the pod**

**Display Existing Pods  
$ kubectl get pods**

**Create Pods using pod manifest yaml  
$ kubectl apply -f <pod.yaml>**

**Get Pods running on which worker nodes  
$ kubectl get pods -o wide**

**Describe Pods with events  
$ kubectl describe pod <pod id>**

**Get Pods Logs  
$ kubectl logs <pod name>**

**Note: PODS we can’t access outside.**

**We need to expose pod for outside access using Kubernetes service concept.**

**K8s service:**

**Kubernetes Service is used to expose the PODS outside Cluster.**

**We have three types of K8s services**

* **Cluster IP**
* **Node Port**
* **Load Balancer**

**we need to write Service manifest Yaml to expose PODS.**

**---  
apiVersion: v1**

**kind: Service**

**metadata:  
 name: arunsvc**

**spec:  
 type: NodePort  
 selector:   
 app: javaapp  
 ports:  
 - port: 80  
 targetPort: 80  
…**

**# Display existing services  
$ kubectl get svc**

**# Create k8s service  
$ kubectl apply -f <svc-manifest.yaml>**

**Note: We can see service information and Node Port number assigned by K8s**

**Note: Enable Node Port number in Security Group of worker node in which our POD is Running.**

**PODS are Ephemeral**

**Pods are short lived Objects**

**We are not using POD IP, because POD Ip will change when it re-created**

**Cluster IP:**

**When we create PODS , Every Pods will get unique IP addresses**

**We can access POD inside cluster using it IP addresses**

**Note: PODS are short lived object , When POD is recreated its IP will be changes so we can’t depend on POD IP to access.**

**To Expose POD within the Cluster we can use Cluster IP Services**

**Cluster IP will generate one static IP address to access our POD within the Cluster**

**Note: Cluster IP will not change even when POD are re-created**