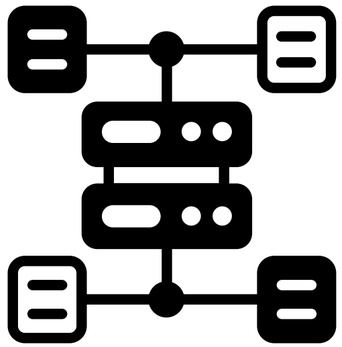
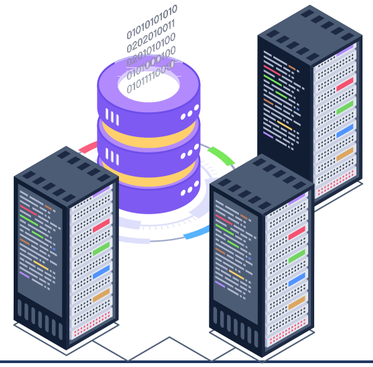
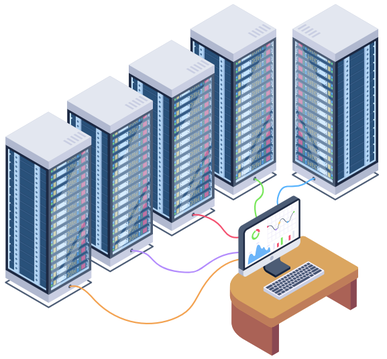
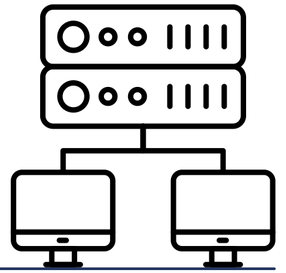
Kubernetes Cluster Set up









By Sandip Das

Basic Components

kubelet

The kubelet is the primary "node agent" that runs on each node. It can register the node with the apiserver using one of: the hostname; a flag to override the hostname; or specific logic for a cloud provider.

kubeadm

Kubeadm automates the installation and configuration of Kubernetes components such as the API server, Controller Manager, and Kube DNS
It's is a tool built to provide kubeadm init and kubeadm join as best-practice "fast paths" for creating Kubernetes clusters

kubectl

The Kubernetes command-line tool, it allows us to run commands against Kubernetes clusters. We can use kubectl to deploy applications, inspect and manage cluster resources, and view logs

kubernetes-cni

It's a an interface between network provider and is used by container runtimes

Let's set up Master Node

01 Install curl and apt-transport-https

sudo apt install apt-transport-https curl

02 Add Kubernetes signing key

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add

03 Add Kubernetes repository

echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" >> ~/kubernetes.list

sudo mv ~/kubernetes.list /etc/apt/sources.list.d

04 Update the servers

sudo apt update

05 Install kubeadm, kubelet, kubectl, and kubernetes-cni

sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni

Verify installation:

kubectl version --client && kubeadm version

Let's set up Master Node

06 Disable Swap Memory

sudo swapoff -a

sudo nano /etc/fstab # comment out **swapfile** line (if any)

07 Setup unique hostname

sudo hostnamectl set-hostname kube-master

run "hostname" command to confirm

08 Enable Bridge Traffic in IP Tables

sudo modprobe br_netfilter

sudo sysctl net.bridge.bridge-nf-call-iptables=1

09 Install Docker Runtime and run docker with systemd

Refer to conainer_runtime_set_up.sh

10 Initialize Kubernetes Master Node

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

Let's set up Master Node

11 Disable Swap Memory

sudo swapoff -a

sudo nano /etc/fstab # comment out **swapfile** line (if any)

12 Setup unique hostname

sudo hostnamectl set-hostname kube-master

run "hostname" command to confirm

13 Enable Bridge Traffic in IP Tables

sudo modprobe br_netfilter

sudo sysctl net.bridge.bridge-nf-call-iptables=1

14 Install Docker Runtime and run docker with systemd

Refer to conainer_runtime_set_up.sh (Click Here)

15 Initialize Kubernetes Master Node

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

After running this command, we will get the join command, which we can use to make worker nodes join master node, in order to create join token later on, run: sudo kubeadm token create --print-join-command

Let's set up Master Node

16 Create Kubernetes Config as advised

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

17 Set-up Pod Network

A pod network facilitates communication between servers and it’s necessary for the proper functioning of the Kubernetes cluster. We will be using the Flannel pod network for this tutorial. Flannel is a simple overlay network that satisfies the Kubernetes requirements.

Allow firewall rule to create exceptions for port 6443 (default port for Kubernetes)

```
sudo ufw allow 6443
sudo ufw allow 6443/tcp
```

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.yml
```

Check Status

```
kubectl get pods --all-namespaces
```

check component's health/status

```
kubectl get componentstatus (or kubectl get cs)
```

Let's set up Worker Node

01 Set-up basic steps

Follow master node set-up set-up step from 1 - 14 (till container runtime setup) then follow below

02 Join

In previous master setup we got the joining command, run it in worker node

sudo kubeadm join MASTER_IP_HERE:6443 --token 1kogba.5e1g1lcgbffw7aqs --discovery-token-ca-cert-hash sha256:e6c4b0477d4ccc07395f9d2c373a90ed73066197986e23f7664b4ec78a0afb58

Let's run a simple nginx app

01 Create Nginx Deployment

kubectl create deployment nginx --image=nginx

02 Make Nginx Accessable

kubectl create service nodeport nginx --tcp=80:80

03 Get accessable link

kubectl get svc

04 Verify working or not

curl localhost:host_port

Thanks for watching!

See you all soon in my next session

Connect

contact@sandipdas.in