Create two server with name master and slave :

Step1: On master instance we have to run these commands:

First set the hostname so that while doing the command there is no confusion that you are running commands on which server master or slave.

$ sudo hostnamectl set-hostname master

$ bash

$ sudo su

$ sudo apt-get update && apt-get upgrade -y

```
ubuntu@ip-172-31-38-2:~$ sudo hostnamectl set-hostname master
ubuntu@ip-172-31-38-2:~$ bash
ubuntu@master:~$ sudo su
root@master:/home/ubuntu# sudo apt-get update && sudo apt-get upgrade -y
```

After this reboot the system by using the command

$ sudo reboot -f

```
Processing triggers for initramfs-tools (0.136ubuntu6.7) ...
update-initramfs: Generating /boot/initrd.img-5.15.0-1048-aws
root@master:/home/ubuntu# sudo reboot -f
Rebooting.
```

After reboot connect with the server again and run the following commands on Master server:

$ sudo su

$ apt-get update

```
Last login: Wed Feb  7 04:59:35 2024 from 112.196.33.226
ubuntu@master:~$ sudo su
root@master:/home/ubuntu# apt-get update
```

$ apt-get install docker.io

```
Reading package lists... Done
root@master:/home/ubuntu# apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

$ apt-get update && apt-get install -y apt-transport-https curl

```
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for dbus (1.12.16-2ubuntu2.3) ...
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
root@master:/home/ubuntu# apt-get update && apt-get install -y apt-transport-htt
ps curl
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
```

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

```
root@master:/home/ubuntu# curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
OK
root@master:/home/ubuntu#
```

$ cat <<EOF >/etc/apt/sources.list.d/kubernetes.list

deb https://apt.kubernetes.io/ kubernetes-xenial main

EOF

```
root@master:/home/ubuntu#  cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
> deb https://apt.kubernetes.io/ kubernetes-xenial main
> EOF
root@master:/home/ubuntu#
```

apt-get update

```
root@master:/home/ubuntu# apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [8993 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [69.9 kB]
Fetched 78.9 kB in 1s (65.2 kB/s)
Reading package lists... Done
root@master:/home/ubuntu#
```

$ apt-get install -y kubelet kubeadm kubectl

```
root@master:/home/ubuntu# apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
```

Step 2: Now On Slave we have to run theses commands:

In this we also do the same thing, first set the hostname so that while doing the command there is no confusion that you are running commands on which server master or slave.

$ sudo hostnamectl set-hostname slave

$ bash

$ sudo su

$ sudo apt-get update && apt-get upgrade -y

```
ubuntu@ip-172-31-40-29:~$ sudo hostnamectl set-hostname slave
ubuntu@ip-172-31-40-29:~$ bash
ubuntu@slave:~$ sudo su
root@slave:/home/ubuntu# sudo apt-get update && sudo apt-get upgrade-y
```

After this reboot the system by using the command

$ sudo reboot -f

```
root@slave:/home/ubuntu# sudo reboot -f
Rebooting.
```

After reboot connect with the server again and run the following commands on Slave server:

$ sudo su

$ apt-get update

```
Last login: Wed Feb  7 05:11:07 2024 from 112.196.33.226
ubuntu@slave:~$ sudo su
root@slave:/home/ubuntu# apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
root@slave:/home/ubuntu#
```

$ apt-get install docker.io

```
Reading package lists... Done
root@slave:/home/ubuntu# apt-get install docker.io -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
```

$ apt-get update && apt-get install -y apt-transport-https curl

```
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
root@slave:/home/ubuntu# apt-get update && apt-get install -y apt-transport-http
s curl
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
```

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

```
root@slave:/home/ubuntu# curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
OK
root@slave:/home/ubuntu#
```

$ cat <<EOF >/etc/apt/sources.list.d/kubernetes.list

deb https://apt.kubernetes.io/ kubernetes-xenial main

EOF

```
OK
root@slave:/home/ubuntu#  cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
> deb https://apt.kubernetes.io/ kubernetes-xenial main
> EOF
root@slave:/home/ubuntu#
```

$ apt-get update

```
root@slave:/home/ubuntu# apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [8993 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [69.9 kB]
Fetched 78.9 kB in 1s (72.0 kB/s)
Reading package lists... Done
root@slave:/home/ubuntu#
```

$ apt-get install -y kubelet kubeadm kubectl

```
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 P
Fetched 78.9 kB in 1s (72.0 kB/s)
Reading package lists... Done
root@slave:/home/ubuntu# apt-get install -y kubelet kubeadm kubectl
```

Step 3: Now on master we Initialize kubeadm using the following command:

$ kubeadm init --apiserver-advertise-address=<master private ip address> --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=all

```
root@master:/home/ubuntu# kubeadm init --apiserver-advertise-address=172.31.38.2 --pod-network-cidr=192.
168.0.0/16 --ignore-preflight-errors=all
```

Copy Kubeadm join command from master to Slave

Now on slave terminal we have to paste kubeadm join command .

We have to paste this token from master to slave so that we connect both master and slave

```
Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.38.2:6443 --token 64s509.e6173numyluflhkg \
        --discovery-token-ca-cert-hash sha256:36eeebbd8e124822b59e1ad4d25e244f5d6034c241048f3b378e107a94
66e335
root@master:/home/ubuntu#
```

Step 4 : Now on master we have to run the following commands:

Exit from root user for that you have to write **exit** on your master terminal

After that perform these commands :

$ mkdir -p $HOME/.kube

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

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

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

$ curl https://raw.githubusercontent.com/projectcalico/calico/v3.26.1/manifests/calico.yaml -O

```
ubuntu@master:~$ curl https://raw.githubusercontent.com/projectcalico/calico/v3.26.1/manifests/calico.ya
ml -O
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  238k  100  238k    0     0   756k      0 --:--:-- --:--:-- --:--:--  753k
ubuntu@master:~$
```

$ kubectl apply -f calico.yaml

```
ubuntu@master:~$ kubectl apply -f calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
serviceaccount/calico-cni-plugin created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
```

$ kubectl get nodes

```
ubuntu@master:~$ kubectl get nodes
NAME     STATUS   ROLES           AGE     VERSION
master   Ready    control-plane   8m48s   v1.28.2
slave    Ready    <none>          8m22s   v1.28.2
ubuntu@master:~$
```

Step 5: Now create deployment file on master server

$ nano deployment.yml

Copy this script and paste it in your deployment file :

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      app: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

$ kubectl create -f deployment.yml

$ kubectl get po

```
ubuntu@master:~$ nano deployment.yml
ubuntu@master:~$ kubectl create -f deployment.yml
deployment.apps/nginx created
ubuntu@master:~$ kubectl get po
NAME                     READY   STATUS    RESTARTS   AGE
nginx-7c5ddbdf54-6jbqv   1/1     Running   0          41s
nginx-7c5ddbdf54-q89bk   1/1     Running   0          41s
nginx-7c5ddbdf54-r56rk   1/1     Running   0          41s
ubuntu@master:~$
```

Step 6: Now we will create service on Master server:

$ nano service.yml

Copy this script and paste it your service file

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  externalTrafficPolicy: Local
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
  type: NodePort
```

$ kubectl create -f service.yml

```
ubuntu@master:~$ nano service.yml
ubuntu@master:~$ kubectl create -r service.yml
error: unknown shorthand flag: 'r' in -r
See 'kubectl create --help' for usage.
ubuntu@master:~$ kubectl create -f service.yml
service/nginx created
ubuntu@master:~$
```

$ kubectl get service |grep nginx

```
ubuntu@master:~$ kubectl get service | grep nginx
nginx        NodePort    10.109.16.41   <none>        80:30950/TCP   117s
ubuntu@master:~$
```

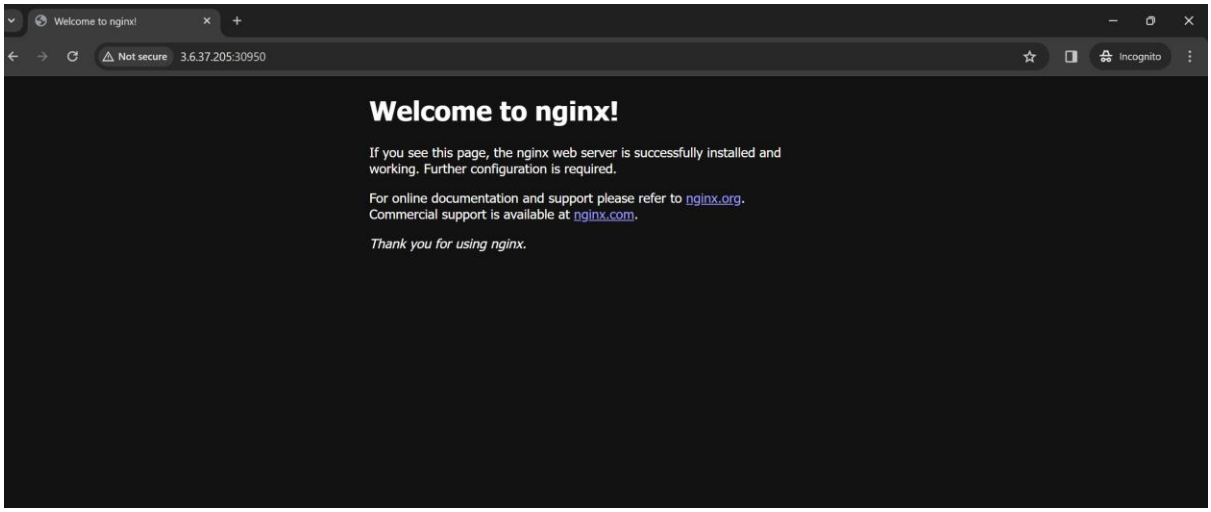
$ kubectl get service | grep nginx

```
ubuntu@master:~$ kubectl get service | grep nginx
nginx        NodePort    10.109.16.41   <none>        80:30950/TCP   117s
ubuntu@master:~$ kubectl describe service nginx
Name:                     nginx
Namespace:                default
Labels:                   app=nginx
Annotations:              <none>
Selector:                 app=nginx
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.109.16.41
IPs:                      10.109.16.41
Port:                     http  80/TCP
TargetPort:               80/TCP
NodePort:                 http  30950/TCP
Endpoints:                192.168.25.4:80,192.168.25.5:80,192.168.25.6:80
Session Affinity:         None
External Traffic Policy:  Local
Events:                   <none>
ubuntu@master:~$
```

In the above screenshot, it can be seen that the Service is available on Port 30950. This can be different for you as the port is randomly assigned from the available range.

Now go on the browser and put public ip address of your master or slave whatever you want and using colon put port number . for ex : Ipslave:30950

Here it is: **3.6.37.205:30950**


Welcome to nginx!
Not secure
3.6.37.205:30950
Incognito
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.