**How to Install Kubernetes Cluster on Ubuntu 22.04**

🡪create 1 VPC.

🡪create 3 subnets.

🡪Create 3 ec2 instances.

Master node

Worker node1

Worker node 2

Open 3 instances in mob

In 3 instances

Sudo su

apt-get update.

Apt-get upgrade.

K8 Install ubuntu 22.04.

<https://www.linuxtechi.com/install-kubernetes-on-ubuntu-22-04/>

A Kubernetes cluster consists of worker nodes on which application workload is deployed and set up master nodes that are used to manage worker nodes and pods in the cluster.

sudo hostnamectl set-hostname master

exec bash

exit

exec bash

sudo swapoff -a

cat /etc/fstab(to check)

top

sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF(from top to all one command)

sudo modprobe overlay

sudo modprobe br\_netfilter

sudo tee /etc/sysctl.d/kubernetes.conf <<EOT

net.bridge.bridge-nf-call-ip6tables = 1

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

net.ipv4.ip\_forward = 1

EOT(from sudo to here)

sudo sysctl --system

sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/docker.gpg

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

sudo apt update

sudo apt install -y containerd.io

containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update

sudo apt install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

sudo kubeadm init

To start using your cluster, you need to run the following as a regular user:

This 3 commands we should give in master only

mkdir -p $HOME/.kube

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

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

ls -a

echo $HOME

cd .kube/

ls(config)

vi config config

cd ..

kubectl get nodes

now keep joint command in worker1 and worker2

Sudo kubeadm join 10.0.10.242:6443 --token 0wk5rk.lt6tr6ju0ahkaiea \

--discovery-token-ca-cert-hash sha256:896744cd00e268e101051d368fb71a040a7d8efa8cdfc5e561fc917013b1ab22

And go to master

Kubectl get nodes

Kubectl get all -A

kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml

kubectl get pods -n kube-system

kubectl get nodes

kubectl create deployment nginx-app --image=nginx --replicas=2

kubectl get pods

kubectl get pods -w

kubectl get pods -A

kubectl get nodes -o wide

kubectl get pods -o wide

kubectl get pods -A -o wide

**Alternatively,** if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:

https://kubernetes.io/docs/concepts/cluster-administration/addons/

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

Sudo kubeadm join 10.0.10.242:6443 --token 0wk5rk.lt6tr6ju0ahkaiea \

--discovery-token-ca-cert-hash sha256:896744cd00e268e101051d368fb71a040a7d8efa8cdfc5e561fc917013b1ab22

A computer screen shot of text

Description automatically generated

Master

1 sudo su

2 exec bash

3 sudo swapoff -a

4 sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

5 sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

6 sudo modprobe overlay

7 sudo modprobe br\_netfilter

8 sudo tee /etc/sysctl.d/kubernetes.conf <<EOT

net.bridge.bridge-nf-call-ip6tables = 1

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

net.ipv4.ip\_forward = 1

EOT

9 sudo sysctl --system

10 sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

11 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

12 sudo apt update

13 sudo apt install -y containerd.io

14 containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

15 sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

16 sudo systemctl restart containerd

17 sudo systemctl enable containerd

18 sudo systemctl status containerd

19 curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

20 echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

21 ls

22 sudo apt update

23 sudo apt install -y kubelet kubeadm kubectl

24 sudo apt-mark hold kubelet kubeadm kubectl

25 sudo kubeadm init

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

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

28 ls -a

29 eco $HOME

30 echo $HOME

31 cd .kube/

32 ls

33 vi config config

34 cd ..

35 kubectl get nodes

36 kubectl get all -A

37 kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml

38 kubectl get nodes

39 kubectl create deployment nginx-app --image=nginx --replicas=2

40 kubectl get nodes

41 kubectl get pods

42 kubectl get pods -w

43 kubectl get pods -A

44 kubectl get nodes -o wide

45 kubectl get pods -o wide

46 kubectl get pods -A -o wide

47 kubectl get pods -A -o wide

48 history

**Worker1**

ubuntu@worker1:~$ history

1 sudo su

2 exec bash

3 sudo swapoff -a

4 sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

5 sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

6 sudo modprobe overlay

7 sudo modprobe br\_netfilter

8 sudo tee /etc/sysctl.d/kubernetes.conf <<EOT

net.bridge.bridge-nf-call-ip6tables = 1

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

net.ipv4.ip\_forward = 1

EOT

9 sudo sysctl --system

10 sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

11 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

12 sudo apt update

13 sudo apt install -y containerd.io

14 containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

15 sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

16 sudo systemctl restart containerd

17 sudo systemctl enable containerd

18 sudo systemctl status containerd

19 curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

20 echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

21 sudo apt update

22 sudo apt install -y kubelet kubeadm kubectl

23 sudo apt-mark hold kubelet kubeadm kubectl

24 sudo kubeadm join 10.0.10.242:6443 --token 0wk5rk.lt6tr6ju0ahkaiea --discovery-token-ca-cert-hash sha256:896744cd00e268e101051d368fb71a040a7d8efa8cdfc5e561fc917013b1ab22

25 history

**Worker2**

ubuntu@worker2:~$ history

1 sudo su

2 exec bash

3 sudo swapoff -a

4 sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

5 cat /etc/fstab

6 sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

7 sudo modprobe overlay

8 sudo modprobe br\_netfilter

9 sudo tee /etc/sysctl.d/kubernetes.conf <<EOT

net.bridge.bridge-nf-call-ip6tables = 1

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

net.ipv4.ip\_forward = 1

EOT

10 sudo sysctl --system

11 sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

12 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

13 sudo apt update

14 sudo apt install -y containerd.io

15 containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

16 sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

17 sudo systemctl restart containerd

18 sudo systemctl enable containerd

19 sudo systemctl status containerd

20 curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

21 echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

22 sudo apt update

23 sudo apt install -y kubelet kubeadm kubectl

24 sudo apt-mark hold kubelet kubeadm kubectl

25 Sudo kubeadm join 10.0.10.242:6443 --token 0wk5rk.lt6tr6ju0ahkaiea --discovery-token-ca-cert-hash sha256:896744cd00e268e101051d368fb71a040a7d8efa8cdfc5e561fc917013b1ab22

26 sudo kubeadm join 10.0.10.242:6443 --token 0wk5rk.lt6tr6ju0ahkaiea --discovery-token-ca-cert-hash sha256:896744cd00e268e101051d368fb71a040a7d8efa8cdfc5e561fc917013b1ab22

27 history

ubuntu@worker2:~$

**Metric server:**

Metrics Server is a scalable, efficient source of container resource metrics for Kubernetes built-in autoscaling pipelines.

**🡪**To continue above installation

🡪go to master node.

🡪ls

🡪metric server k8

🡪 <https://github.com/kubernetes-sigs/metrics-server>

🡪create directory.

🡪 mkdir Metris server

🡪cd Metris/

🡪go to the document installation

🡪 wget <https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml>

🡪ls(components.yaml)

🡪vi components.yaml

🡪 - --kubelet-insecure-tls=true

🡪kubectl apply -f components.yaml

🡪 kubectl get all -n kube-system (metris should run)

🡪 kubectl get pod -n kube-system -w (Metris should run)

🡪To get all Kubernetes API resources commands

🡪 kubectl api-resources

🡪 kubectl get apiservice(when you get error)

🡪 kubectl get all -n kube-system

🡪kubectl edit(deployment-Metris-server should keep this)-n kube-system

🡪go to that file down dns

🡪kubectl top nodes (Metris not available)

🡪 kubectl get all -n kube-system (pod Metris error)

🡪kubectl logs pod (pina error pod copy and paste) -n kube-system

🡪vi components.yaml

🡪change container port 10255 two times change.

🡪and again secvice account name down write

🡪 hostNetwork: true

🡪

🡪 kubectl top nodes

🡪 kubectl top pods -A

🡪kubectl get hpa -n nginx -w(unkown50%)

🡪 kubectl top pods -A(to check all )

🡪kubectl top pod (top lo nginx copy)(individual to get)

🡪cd ..

🡪ls(total 5 need to get)

🡪cd nginx/

🡪ls(deployment.yaml)

🡪kubectl delete -f deployment.yaml -n nginx(deleted)

🡪