**Setting Up Your Kubernetes Cluster**

To set up your Kubernetes cluster with one master node and two worker nodes, follow these steps:

**1. Prerequisites**

* **Ensure each node** has Ubuntu 22.04 installed.
* **Access with sudo privileges** to all nodes.
* **Set up SSH access** between nodes for convenience.

**2. Install Docker on All Nodes**

Install Docker on the master and worker nodes:

1. Update the package list:

bash

Copy code

sudo apt update

1. Install Docker:

bash

Copy code

sudo apt install docker.io -y

1. Enable and verify Docker:

bash

Copy code

sudo systemctl enable docker

sudo systemctl start docker

sudo systemctl status docker

**3. Install Kubernetes Tools on All Nodes**

1. Add the Kubernetes signing key:

bash

Copy code

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

1. Add the Kubernetes repository:

bash

Copy code

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update

1. Install kubeadm, kubelet, and kubectl:

bash

Copy code

sudo apt install kubeadm kubelet kubectl -y

1. Mark the packages to prevent updates:

bash

Copy code

sudo apt-mark hold kubeadm kubelet kubectl

1. Verify the installation:

bash

Copy code

kubeadm version

**4. Prepare the Nodes**

On **each node**:

1. Disable swap:

bash

Copy code

sudo swapoff -a

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

1. Load kernel modules:

bash

Copy code

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

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

1. Configure networking:

bash

Copy code

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

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

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

net.ipv4.ip\_forward = 1

EOF

sudo sysctl --system

**5. Configure Hostnames and Hosts File**

* Set unique hostnames:

bash

Copy code

sudo hostnamectl set-hostname master-node # On the master

sudo hostnamectl set-hostname worker01 # On worker 1

sudo hostnamectl set-hostname worker02 # On worker 2

* Update /etc/hosts on **each node** with all nodes' IPs and hostnames:

plaintext

Copy code

192.168.2.150 master-node

192.168.2.151 worker01

192.168.2.152 worker02

**6. Initialize the Master Node**

On the **master node**:

1. Initialize the cluster:

bash

Copy code

sudo kubeadm init --control-plane-endpoint=master-node --upload-certs

1. Save the kubeadm join command provided in the output.
2. Configure kubectl for the master:

bash

Copy code

mkdir -p $HOME/.kube

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

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

1. Deploy a network plugin (e.g., Flannel):

bash

Copy code

kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml

1. Untaint the master node:

bash

Copy code

kubectl taint nodes --all node-role.kubernetes.io/control-plane-

**7. Join Worker Nodes**

On **each worker node**:

1. Use the kubeadm join command saved earlier:

bash

Copy code

sudo kubeadm join 192.168.2.150:6443 --token <token> --discovery-token-ca-cert-hash sha256:<hash>

**8. Verify the Cluster**

On the **master node**, check the status of all nodes:

bash

Copy code

kubectl get nodes

You should see the master and both worker nodes in a Ready state.

**Conclusion**

Your Kubernetes cluster is now up and running with one master node and two worker nodes. You can deploy applications and manage workloads across the cluster using kubectl.