**Kubernetes Installation Guide on Ubuntu Server (Step-by-Step with Explanation)**

**1. System Preparation(control plane setup):**

**1.1 Disable Swap**

Kubernetes requires swap to be disabled to ensure predictable memory management.

sudo swapoff -a

**1.2 Enable Kernel Modules**

These modules are necessary for Kubernetes networking.

cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf

br\_netfilter

EOF

sudo modprobe br\_netfilter

**1.3 Configure sysctl**

These settings allow Kubernetes networking to work properly.

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf

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

net.ipv4.ip\_forward = 1

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

EOF

sudo sysctl --system

**2. Install and Configure containerd (Container Runtime)**

**2.1 Install containerd**

sudo apt update

sudo apt install -y containerd

**2.2 Generate default configuration file**

This creates a default config to edit.

sudo mkdir -p /etc/containerd

containerd config default | sudo tee /etc/containerd/config.toml

**2.3 Enable Systemd cgroup driver**

Kubelet expects SystemdCgroup = true to match its default configuration.

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

**2.4 Restart containerd**

sudo systemctl restart containerd

sudo systemctl enable containerd

**3. Install Kubernetes Tools**

**3.1 Add Kubernetes apt repository**

sudo apt install -y apt-transport-https curl

sudo curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | \

sudo tee /etc/apt/keyrings/kubernetes-apt-keyring.asc > /dev/null

echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.asc] \

https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /" | \

sudo tee /etc/apt/sources.list.d/kubernetes.list

**3.2 Install kubelet, kubeadm, and kubectl**

sudo apt update

sudo apt install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

Holding the versions prevents them from being upgraded automatically, which may break compatibility.

**4. Initialize Kubernetes Cluster**

Use a **pod network CIDR** (IP range for pods). Flannel uses 10.244.0.0/16.

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

**5. Configure kubectl for Your User**

After kubeadm initializes the cluster, kubectl needs the kubeconfig file.

mkdir -p $HOME/.kube

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

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

You can now run kubectl commands:

$kubectl get nodes

**6. Install a Pod Network Add-on CNI (Flannel)**

Without a pod network, your pods cannot communicate. Flannel is a simple and popular choice.

$kubectl apply -f <https://raw.githubusercontent.com/flannel-io/flannel/master/Documentation/kube-flannel.yml>

$kubectl get pods -n kube-system

**System Preparation(workernode-setup):**

Repeat the above steps up to step 3 before init process.

**7. Joining the worker node to the control plane**

**Type the below command inside the worker node to connect the control plane:**

$ sudo kubeadm join --token <token> <control-plane-host>:<control-plane-port> --discovery-token-ca-cert-hash sha256:<hash>

**Generate the token inside the control plane:**

$ sudo kubeadm token list

$ sudo kubeadm token create

If you don't have the value of --discovery-token-ca-cert-hash, you can get it by running the following commands on the control plane node:

$ sudo cat /etc/kubernetes/pki/ca.crt | openssl x509 -pubkey | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.\* //'

**NOTE:**

To remove the role:

$ kubectl label node <container-name> kubernetes.io/role-

To add the role :

$ kubectl label node <container-name> node-role.kubernetes.io/worker or control-plane=worker or control-plane