**Docker setup**  
  
apt install docker.io -y

systemctl enable docker ; systemctl start docker

**Kubernetes setup**

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/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:/v1.30/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

apt update

apt install kubeadm kubelet kubectl

***Kubeadm*** *is a tool built to provide kubeadm init and kubeadm join as best-practice "fast paths" for creating Kubernetes clusters.*

***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.*

***Kubectl****. The command-line interface for interacting with clusters,*

*Kubernetes provides a command line tool for communicating with a Kubernetes cluster's control plane, using the Kubernetes API.*

*This tool is named kubectl.*

*For configuration, kubectl looks for a file named config in the $HOME/.kube directory. You can specify other* [*kubeconfig*](https://kubernetes.io/docs/concepts/configuration/organize-cluster-access-kubeconfig/) *files by setting the KUBECONFIG environment variable or by setting the* [*--kubeconfig*](https://kubernetes.io/docs/concepts/configuration/organize-cluster-access-kubeconfig/) *flag.*

apt-mark hold kubeadm kubelet kubectl

*To avoid auto update*

kubeadm version

swapoff -a

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

vim /etc/modules-load.d/containerd.conf

overlay

br\_netfilter

modprobe overlay

modprobe br\_netfilter

vim /etc/sysctl.d/kubernetes.conf

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

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

net.ipv4.ip\_forward = 1

sysctl –system

hostnamectl set-hostname master01

hostnamectl set-hostname worker01

vim /etc/hosts

192.168.1.25 k8s-master

192.168.1.26 k8s-worker-01

vim /etc/default/kubelet

KUBELET\_EXTRA\_ARGS="--cgroup-driver=cgroupfs"

systemctl daemon-reload && sudo systemctl restart kubelet

vim /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"],

"log-driver": "json-file",

"log-opts": {

"max-size": "100m"

},

"storage-driver": "overlay2"

}

systemctl daemon-reload && sudo systemctl restart docker

vim /etc/systemd/system/kubelet.service.d/10-kubeadm.conf

systemctl daemon-reload && sudo systemctl restart kubelet

vim /etc/systemd/system/kubelet.service.d/10-kubeadm.conf

Environment="KUBELET\_EXTRA\_ARGS=--fail-swap-on=false"

kubeadm init --apiserver-advertise-address=192.168.1.25 --control-plane-endpoint=k8s-master --upload-certs

troubleshooting step:  
rm /etc/containerd/config.toml

systemctl restart containerd

rm /etc/kubernetes/manifests/\*

kubeadm reset

rm -rf .kube/

rm -rf /etc/kubernetes/

rm -rf /var/lib/kubelet/

rm -rf /var/lib/etcd

/etc/docker/daemon.json

"exec-opts": ["native.cgroupdriver=systemd"],

/etc/containerd/config.toml

#disabled\_plugins = ["cri"]

mkdir -p $HOME/.kube

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

*Your Kubernetes control-plane has initialized successfully!*

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

*mkdir -p $HOME/.kube*

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

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

*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/*](https://kubernetes.io/docs/concepts/cluster-administration/addons/)

*You can now join any number of the control-plane node running the following comm and on each as root:*

*kubeadm join kmaster:6443 --token 6596nq.e47rdkpikydgl6ex \*

*--discovery-token-ca-cert-hash sha256:f749c0ffdb76cc8c272b1a3f97d5ceb63e fd676660fa9e8e8e4cf8a66fd5544b \*

*--control-plane --certificate-key 1ec7ac065f0e4b19b70698759a5b0fc15f67b4 8588293e63785ee831c8e1de23*

*Please note that the certificate-key gives access to cluster sensitive data, kee p it secret!*

*As a safeguard, uploaded-certs will be deleted in two hours; If necessary, you can use*

*"kubeadm init phase upload-certs --upload-certs" to reload certs afterward.*

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

*kubeadm join kmaster:6443 --token 6596nq.e47rdkpikydgl6ex \*

*--discovery-token-ca-cert-hash sha256:f749c0ffdb76cc8c272b1a3f97d5ceb63e fd676660fa9e8e8e4cf8a66fd5544b*

kubectl label node kslave node-role.kubernetes.io/worker=worker node/kslave labeled