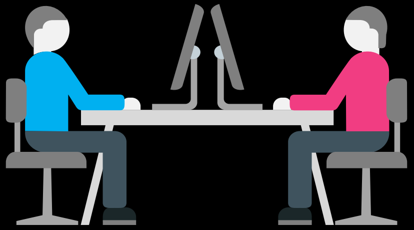
Installing Kubernetes Cluster using

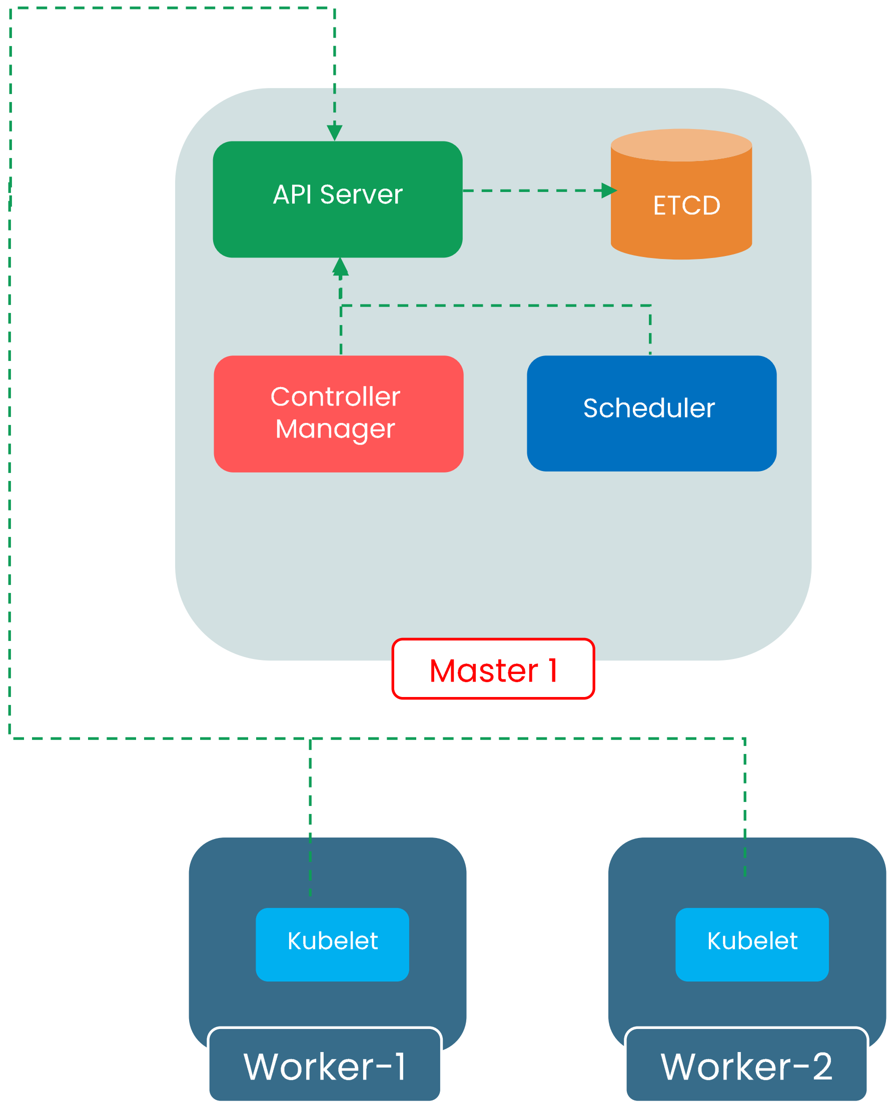
Kubeadm



kubeEd.com

srinathchalla@outlook.com

SCALING KUBERNETES MASTER COMPONENTS



RAM: 4GB

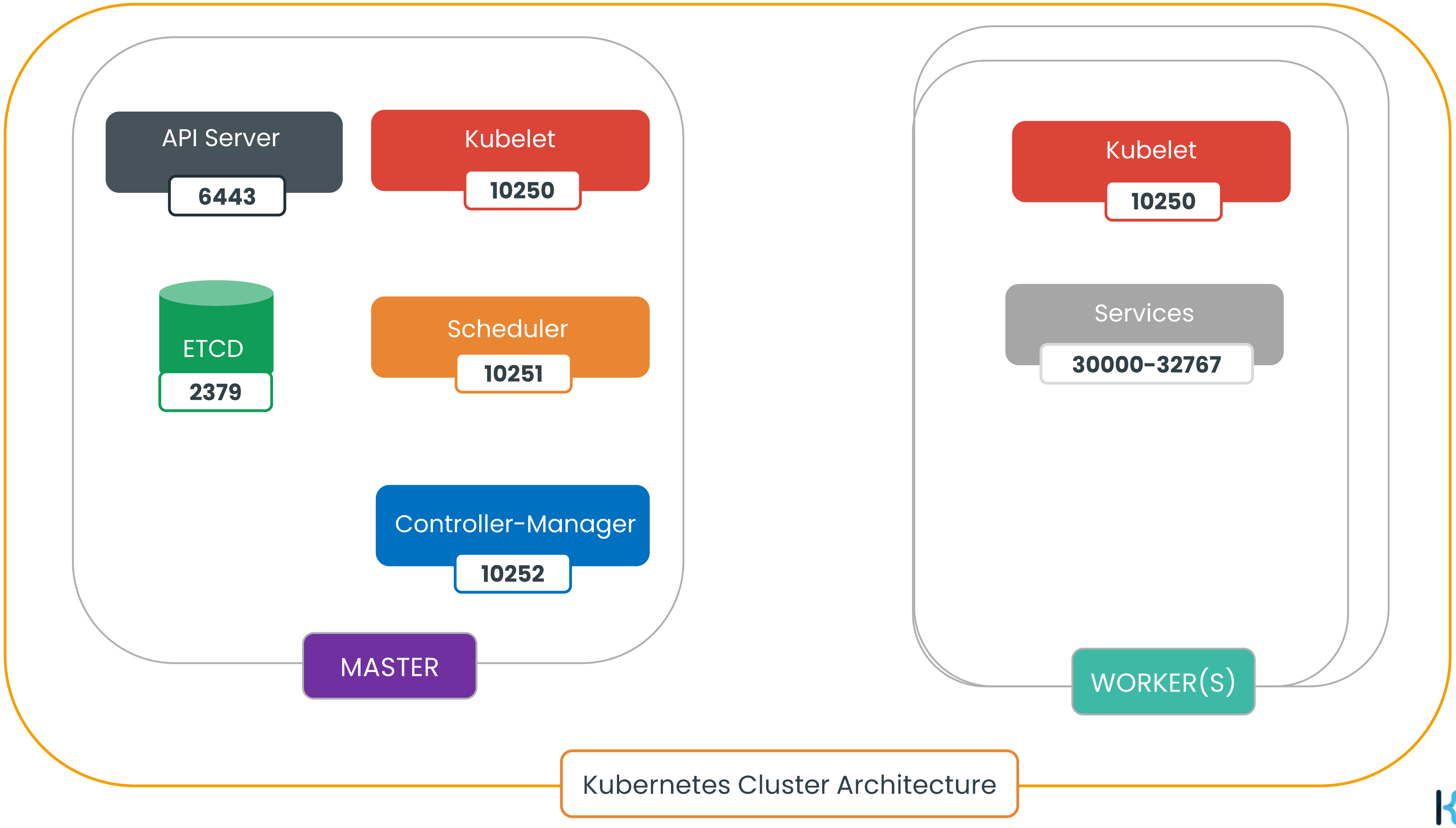
CPU: 2vCPUs

OS: Ubuntu 16.04.7

Google Cloud Platform

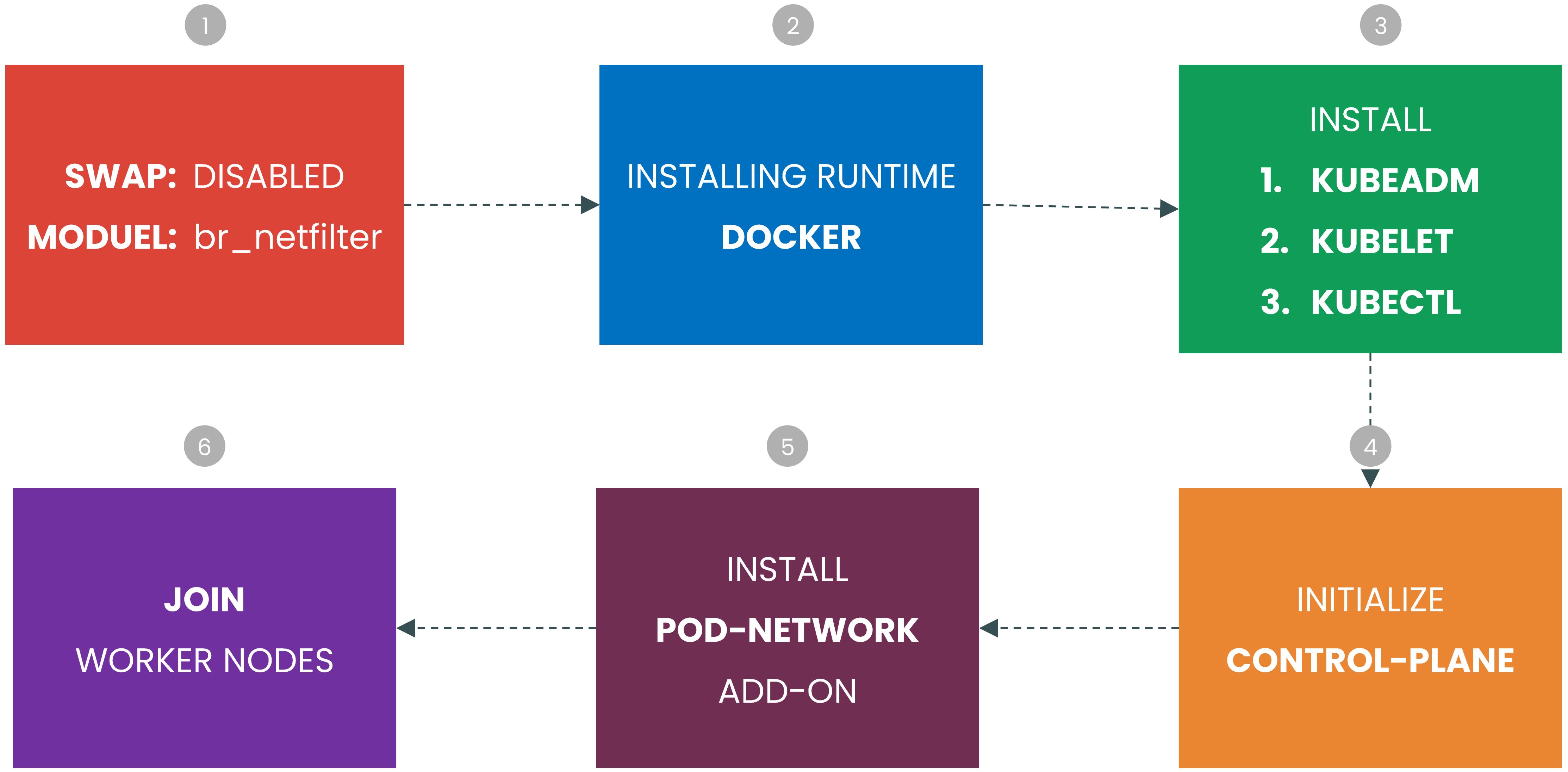
KubeEd.com

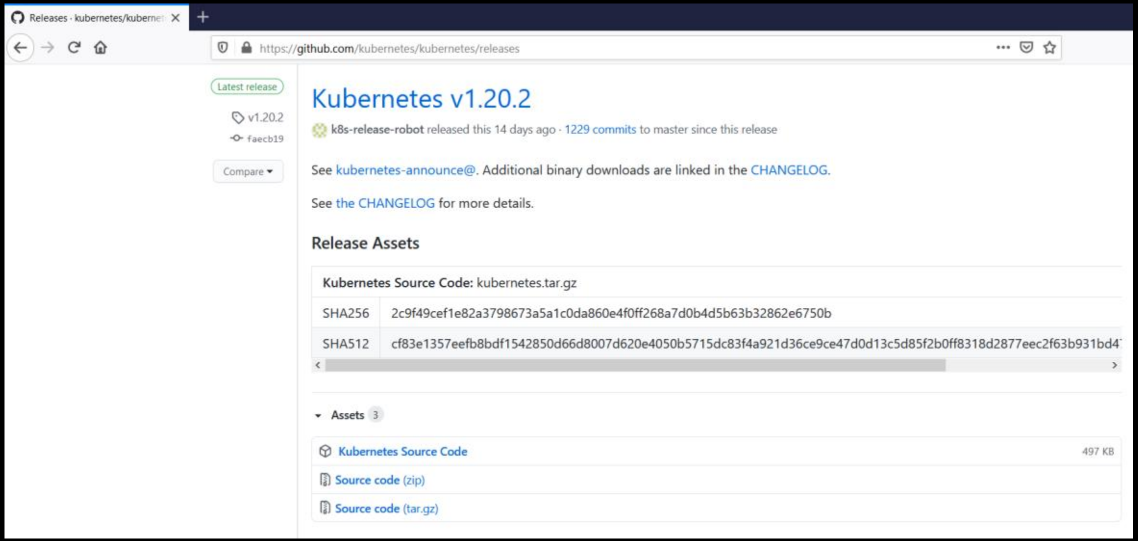
PORTS: TCP – INBOUND (INGRESS)



KubeEd.com

INSTALLING **KUBERNETES** USING **KUBEADM**





Releases · kubernetes/kubernetes

https://github.com/kubernetes/kubernetes/releases

Latest release

v1.20.2

faecb19

Compare ▾

Kubernetes v1.20.2

k8s-release-robot released this 14 days ago · 1229 commits to master since this release

See kubernetes-announce@. Additional binary downloads are linked in the CHANGELOG.

See the CHANGELOG for more details.

Release Assets

Kubernetes Source Code: kubernetes.tar.gz	
SHA256	2c9f49cef1e82a3798673a5a1c0da860e4f0ff268a7d0b4d5b63b32862e6750b
SHA512	cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d2877eec2f63b931bd4

▾ Assets 3

Kubernetes Source Code	497 KB
Source code (zip)	
Source code (tar.gz)	

v1.20.2

Jan 2021

1

SWAP: DISABLED

MODUEL: br_netfilter

Disabling SWAP

```
swapoff -a
```

```
sed -i.bak -r 's/(.+ swap .+)/#\1/' /etc/fstab
```

Bridged traffic

```
lsmod | grep br_netfilter
```

```
sudo modprobe br_netfilter
```

```
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
```

```
sudo sysctl --system
```

KubeEd.com

2. INSTALLING RUNTIME DOCKER

```
sudo apt-get update

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

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

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

KubeEd.com

2. INSTALLING RUNTIME

DOCKER

```
sudo apt-get update && sudo apt-get install -y \
  containerd.io=1.2.13-2 \
  docker-ce=5:19.03.11~3-0~ubuntu-$(lsb_release -cs) \
  docker-ce-cli=5:19.03.11~3-0~ubuntu-$(lsb_release -cs)
```

```
# Set up the Docker daemon
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF
```

```
sudo mkdir -p /etc/systemd/system/docker.service.d
```

```
sudo systemctl daemon-reload
sudo systemctl enable docker
sudo systemctl restart docker
sudo systemctl status docker
```

3. Installing

KUBEADM - KUBELET - KUBECTL

```
sudo apt-get update && sudo apt-get install -y apt-transport-https curl

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

cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```

```
sudo systemctl daemon-reload
sudo systemctl enable kubelet
sudo systemctl restart kubelet
sudo systemctl status kubelet
```

KubeEd.com

4. Initializing **CONTROL-PLANE**

```
kubeadm init
```

```
. . .
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

You should now deploy a Pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  /docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join <control-plane-host>:<control-plane-port> --token <token> --discovery-token-ca-cert-hash sha256:<hash>
To make kubectl work for your non-root user, run these commands, which are also part of the

kubeadm join 10.128.0.10:6443 --token pf0lkm.f239fukrkuzhk951 \
    --discovery-token-ca-cert-hash sha256:aac8491d0502d6f420ac8ae50a5e181647893f9fcdb8f74c42538ffcb3e65d79
```

5. Installing **POD-NETWORK** add-on

```
# for kubectl
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

```
kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"
```

```
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
```

KubeEd.com

6. **JOINING** WORKER NODES

```
kubeadm join 10.128.0.10:6443 --token pf0lkm.f239fukrkuzhk951 \
    --discovery-token-ca-cert-hash sha256:aac8491d0502d6f420ac8ae50a5e181647893f9fcdb8f74c42538ffcb3e65d79
```

```
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -
oyaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

```
kubeadm token create --print-join-command
```