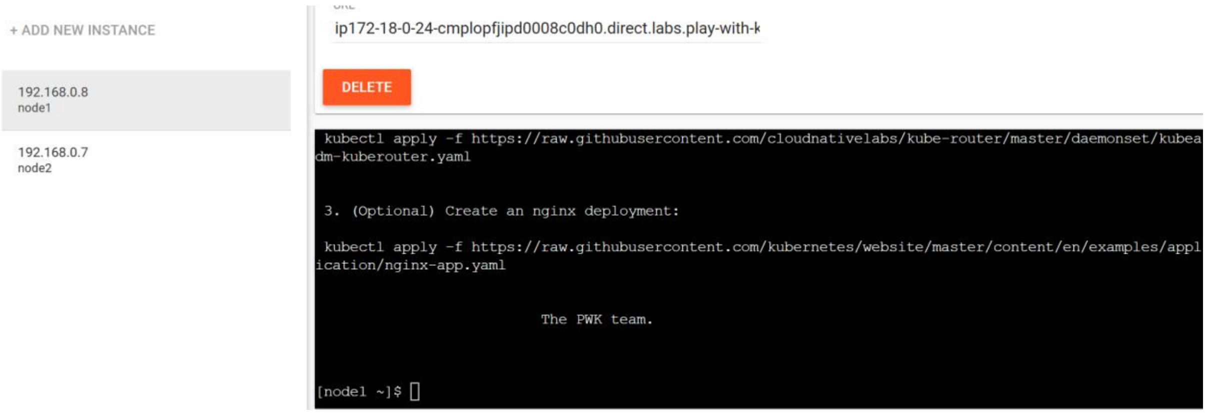
Cluster creation from a two node machine

- We are having a two machines, these has to be connected , declaring one as master node and the other as worker node



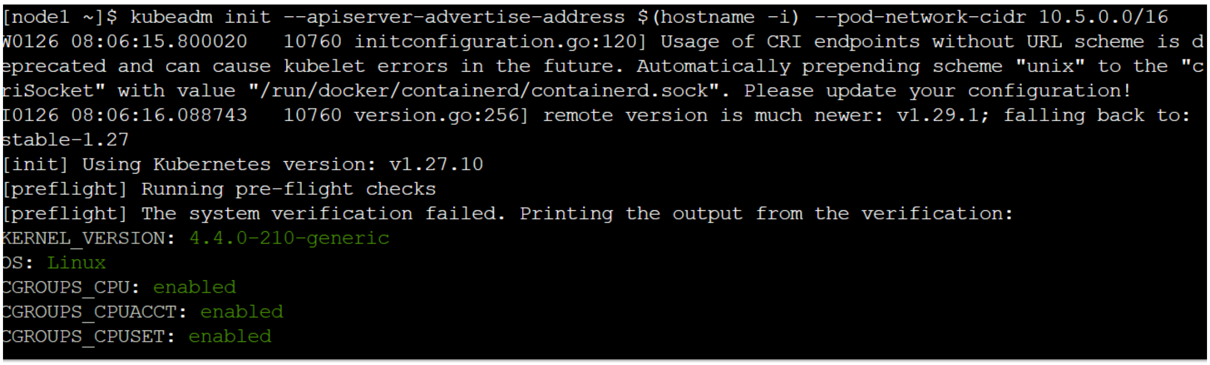
```
 kubectl apply -f https://raw.githubusercontent.com/cloudnativelabs/kube-router/master/daemonset/kubea
dm-kuberouter.yaml


 3. (Optional) Create an nginx deployment:

 kubectl apply -f https://raw.githubusercontent.com/kubernetes/website/master/content/en/examples/appl
ication/nginx-app.yaml


                          The PWK team.


[node1 ~]$ 
```

Step 1 - Installing packages in master node

- Since the node machine already comes with softwares like kubeadm and kubectl , no need to install those
- In the cli enter the following command

```
kubeadm init --apiserver-advertise-address $(hostname -i) --
pod-network-cidr 10.5.0.0/16
```

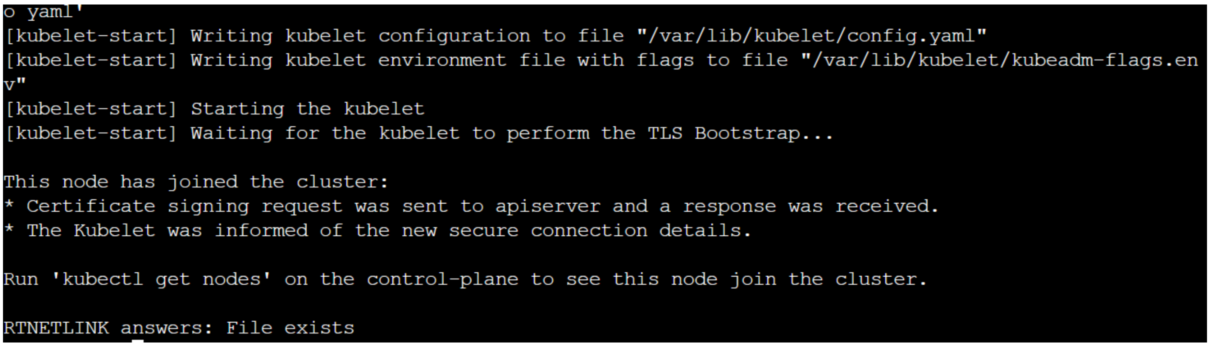
- Kubeadm init - will setup a Kubernetes control plane in machine
- --apiserver-advertise-address $(hostname -i) - This will get advertise the api layer of Kubernetes to the host machine
- Pod-network - This is range of pod network



```
[node1 ~]$ kubeadm init --apiserver-advertise-address $(hostname -i) --pod-network-cidr 10.5.0.0/16
W0126 08:06:15.800020   10760 initconfiguration.go:120] Usage of CRI endpoints without URL scheme is d
eprecated and can cause kubelet errors in the future. Automatically prepending scheme "unix" to the "c
riSocket" with value "/run/docker/containerd/containerd.sock". Please update your configuration!
I0126 08:06:16.088743   10760 version.go:256] remote version is much newer: v1.29.1; falling back to:
stable-1.27
[init] Using Kubernetes version: v1.27.10
[preflight] Running pre-flight checks
[preflight] The system verification failed. Printing the output from the verification:
KERNEL_VERSION: 4.4.0-210-generic
OS: Linux
CGROUPS_CPU: enabled
CGROUPS_CPUACCT: enabled
CGROUPS_CPUSET: enabled
```

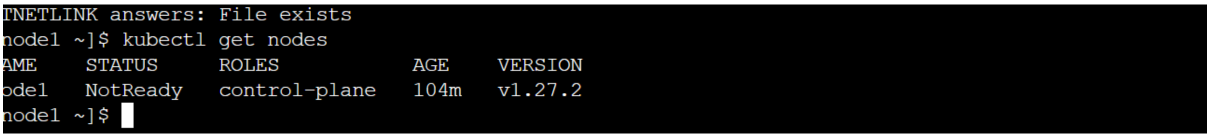
- After this there will be a config file created in the location /etc/kubernetes/admin.conf
- At the end of command execution , there will be number of commands that has to be executed

```
mkdir -p $HOME/.kube

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

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

export KUBECONFIG=/etc/kubernetes/admin.conf

kubeadm join 192.168.0.8:6443 --token c3u6xu.evrwojz3mvcw8ulv --discovery-token-ca-cert-
hash sha256:22acffd2bc6c76faa4a8d0c3ba6e22ca2556b127f86641fb552e575d0c05f582
```



```
o yaml'
[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.en
v"
[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.

RTNETLINK answers: File exists
```

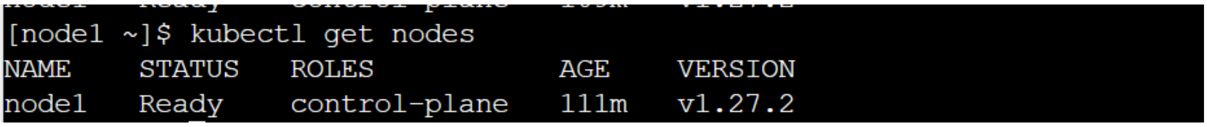
- Kubeadm join – used to join the nodes
- 192.168.0.8:6443 – Ip address of the control plane



```
TNETLINK answers: File exists
node1 ~]$ kubectl get nodes
AME     STATUS     ROLES           AGE    VERSION
ode1    NotReady   control-plane   104m   v1.27.2
node1 ~]$
```

- Node is the not ready state, This is because we have not initialized networking for the cluster. Run the following command to initialize the networking

```
kubectl apply -f
https://raw.githubusercontent.com/cloudnativelabs/kube-
router/master/daemonset/kubeadm-kuberouter.yaml
```



```
[node1 ~]$ kubectl get nodes
NAME     STATUS    ROLES           AGE     VERSION
node1    Ready     control-plane   111m    v1.27.2
```