**Kubernetes Set Up Using Kubeadm**

**On Master and Worker Nodes :**

# Add the docker repository GPG key

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

# Add the Docker repository

1. sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu/dists/artful \
   1. $(lsb\_release -cs) \
   2. stable"

# Reload the apt sources list

1. sudo apt-get update

# Install Docker

1. sudo apt-get install -y docker-ce=18.06.1~ce~3-0~ubuntu

# Prevent auto updates for docker package

1. sudo apt-mark hold docker-ce

# Check Docker version and successful installation

1. sudo docker version

# Add the Kubernetes repository GPG key

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

# Add the Kubernetes repository

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

# Reload the apt sources list

1. sudo apt-get update

# Install Packages

1. sudo apt-get install -y kubelet=1.12.2-00 kubeadm=1.12.2-00 kubectl=1.12.2-00

# Prevent auto updates for kube package

1. sudo apt-mark hold kubelet kubeadm kubectl

# Check Docker version and successful installation

1. kubeadm version

**On Master Node :**

**#** Initialize the cluster on Kube master server

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

# Set up Kubeconfig for the local user on kube master server . This will allow to use kubectl when logged in to the master

1. mkdir -p $HOME/.kube

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

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

# Check kubectl version

1. kubectl version

You should get Server Version as well as Client Version. It should look something like this:

Client Version: version.Info{Major:"1", Minor:"12", GitVersion:"v1.12.2", GitCommit:"17c77c7898218073f14c8d573582e8d2313dc740", GitTreeState:"clean", BuildDate:"2018-10-24T06:54:59Z", GoVersion:"go1.10.4", Compiler:"gc", Platform:"linux/amd64"}

Server Version: version.Info{Major:"1", Minor:"12", GitVersion:"v1.12.2", GitCommit:"17c77c7898218073f14c8d573582e8d2313dc740", GitTreeState:"clean", BuildDate:"2018-10-24T06:43:59Z", GoVersion:"go1.10.4", Compiler:"gc", Platform:"linux/amd64"}

# The kubeinit command should have provided a kubeadm join command. Copy that command from master node and run it on the **worker nodes :**

1. sudo kubeadm join $some\_ip:6443 --token $some\_token --discovery-token-ca-cert-hash $some\_hash

**On Master Node :**

# Get a list of all the nodes. Check the cluster is successfully set up. The nodes will be in “Not Ready” state

1. kubectl get nodes

You should see all three of your nodes listed. It should look something like this:

NAME STATUS ROLES AGE VERSION

wboyd1c.mylabserver.com NotReady master 5m17s v1.12.2

wboyd2c.mylabserver.com NotReady <none> 53s v1.12.2

wboyd3c.mylabserver.com NotReady <none> 31s v1.12.2

**On Master and Worker Nodes :**

# For networking to work run :

1. echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf

sudo sysctl –p

**On Master Node :**

# Install Flannel using a YAML template

1. kubectl apply -f <https://raw.githubusercontent.com/coreos/flannel/bc79dd1505b0c8681ece4de4c0d86c5cd2643275/Documentation/kube-flannel.yml>

# Verify that all the nodes now have a STATUS of Ready:

1. kubectl get nodes

You should see all three of your servers listed, and all should have a STATUS of Ready. It should look something like this:

NAME STATUS ROLES AGE VERSION

wboyd1c.mylabserver.com Ready master 5m17s v1.12.2

wboyd2c.mylabserver.com Ready <none> 53s v1.12.2

wboyd3c.mylabserver.com Ready <none> 31s v1.12.2

# Get a list of system pods:

1. kubectl get pods -n kube-system
2. kubeadm token generate