Lab Manual – Kubernetes Multinode Cluster on Ubuntu

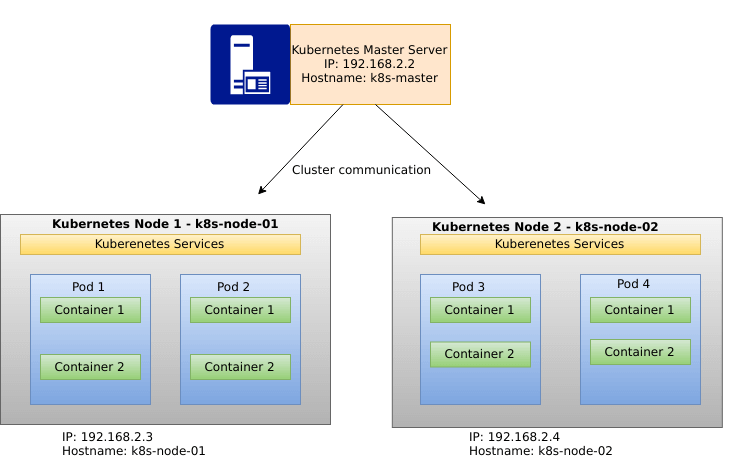
**Steps:**

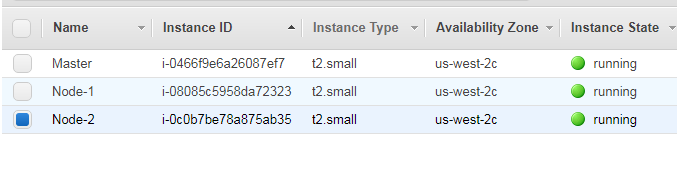
1. Diagram
2. Configure system hostnames and the /etc/hosts file
3. Prerequisites (Run on all nodes)
4. Install Docker Engine ( Run on all Nodes)
5. Install and Configure Kubernetes Master

# **How to setup 3 node Kubernetes Cluster on Ubuntu 18.04**

**Step1: Diagram**

This setup is based on the following diagram:





1. **Step2: Configure system hostnames and the /etc/hosts file**

**On Master Node:**

Set hostname like below:

$ sudo hostnamectl set-hostname k8s-master

**On Worker Node 01:**

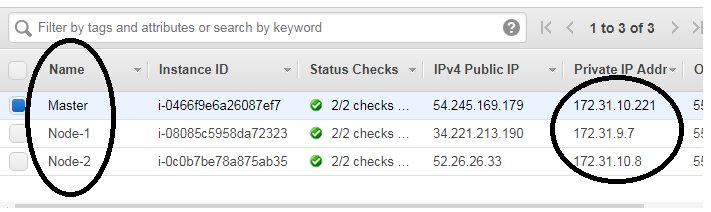
Set the hostname using hostamectl command line tool.

$ sudo hostnamectl set-hostname k8s-node-01

**On Worker Node 02:**

Also set hostname for Kubernetes worker node 02.

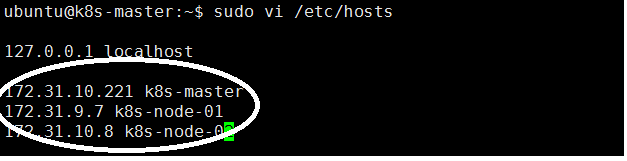
$ sudo hostnamectl set-hostname k8s-node-02



Once correct hostname has been configured on each host, update the /etc/hosts file with the same as below.

$ sudo vi /etc/hosts

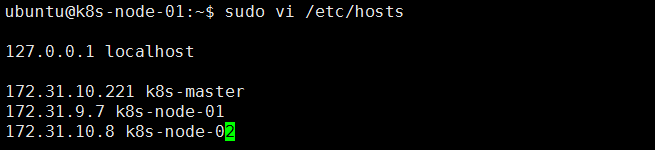
**Master Node**



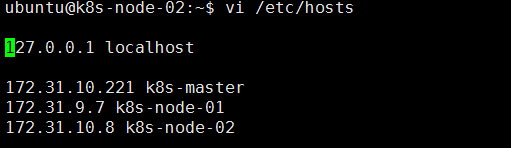
Make sure you use the Private IP’s of the EC2 instance

And NOT the Public IP.

**Node-01**



**Node-02**



**Step3: Prerequisites (Run on all nodes)**

Update system packages to the latest release on all nodes:

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install linux-image-extra-virtual

sudo reboot

Add user to manage Kubernetes cluster:

sudo useradd -s /bin/bash -m k8s-admin

sudo passwd k8s-admin

sudo usermod -aG sudo k8s-admin

echo "k8s-admin ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/k8s-admin

If you prefer entering sudo password when running sudo commands as k8s-admin user, then you can ignore the last line. You can test if no password prompt for sudo:

$ su - k8s-admin

k8s-admin@k8s-master:~$ sudo su -

root@k8s-master:~#

All looks good, let’s proceed to install Docker engine.

## Step4: Install Docker Engine

 Ensure any old version of Docker engine is uninstalled on your system:

sudo apt-get remove docker docker-engine docker.i

Install dependencies:

$ sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

software-properties-common

Import Docker repository GPG key:

$ 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"

**Install docker:**

sudo apt-get update

sudo apt-get install docker-ce -y

sudo usermod -aG docker k8s-admin

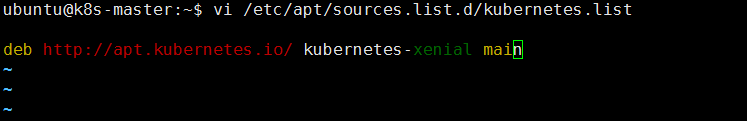
When docker has been installed, you can continue to configure the Kubernetes master node.

## Step5: Install and Configure Kubernetes Master

we will add a repository for Ubuntu 16.04

$ sudo vi /etc/apt/sources.list.d/kubernetes.list

deb <http://apt.kubernetes.io/> kubernetes-xenial main



Then import GPG key:

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

Update apt package index:

sudo apt update

#### **Install Kubernetes Master Components**

Install kubectl, kubelet, kubernetes-cni and kubeadmKubernetes master components:

sudo apt install kubectl kubelet kubeadm kubernetes-cni

Confirm that all package binaries are present on the file system.

$ which kubelet

/usr/bin/kubelet

$ which kubeadm

/usr/bin/kubeadm

If swap is on, turn it off. ( Run this on all the nodes)

sudo swapoff -a

#### **Initialize Kubernetes Cluster**

When all Kubernetes packages have been installed, you’re ready to initialize the cluster using kubeadm command line tool.

Export required variables (**Optional)**

export API\_ADDR=`ifconfig eth0 | grep 'inet'| cut -d':' -f2 | awk '{print $2}'`

export DNS\_DOMAIN="k8s.local"

export POD\_NET="10.4.0.0/16"

export SRV\_NET="10.5.0.0/16"

Then initialize the Kubernetes cluster using variables defined above:

kubeadm init --pod-network-cidr ${POD\_NET} --service-cidr ${SRV\_NET} \

--service-dns-domain "${DNS\_DOMAIN}" --apiserver-advertise-address ${API\_ADDR}

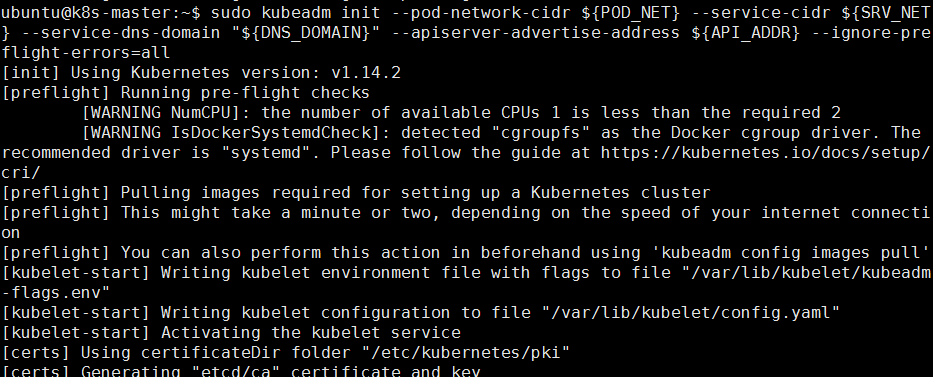
Note: -- If you have selected a VM’s less than 2 CPU, you might get an warning.

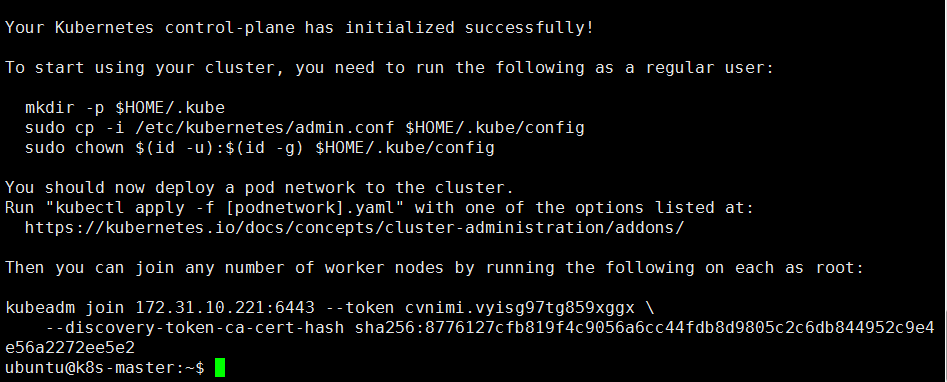
Then run the below

kubeadm init --pod-network-cidr ${POD\_NET} --service-cidr ${SRV\_NET} --service-dns-domain "${DNS\_DOMAIN}" --apiserver-advertise-address ${API\_ADDR} --ignore-preflight-errors=all

If all goes well , here is the output

OUTPUT:





In the above the kubeadm join is visible, which shows the Cluster is enabled successfully on Master node.

#### **Configure Access for k8s-admin user on the Master server**

Switch to k8s-adminand copy Kubernetes configuration file with cluster information.

su - k8s-admin

mkdir -p $HOME/.k8s

sudo cp -i /etc/kubernetes/admin.conf $HOME/.k8s/config

sudo chown $(id -u):$(id -g) $HOME/.k8s/config

export KUBECONFIG=$HOME/.k8s/config

echo "export KUBECONFIG=$HOME/.k8s/config" | tee -a ~/.bashrc