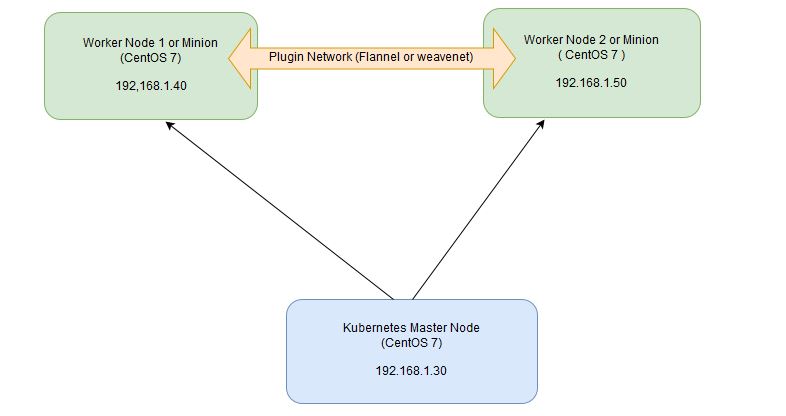
How to Install Kubernetes (k8s) 1.7 on CentOS 7 / RHEL 7



Min Virtual machine requirement

CPU 🡪 2

RAM 🡪 1.5 GB

Disk 🡪 10 GB

No of Node – 3 , 1 for Kubernetes Master and 2 for the Nodes.

#### **On the Master Node following components will be installed**

* **API Server**  – It provides kubernetes API using Jason / Yaml over http, states of API objects are stored in etcd
* **Scheduler**– It is a program on master node which performs the scheduling tasks like launching containers in worker nodes based on resource availability
* **Controller Manager** – Main Job of Controller manager is to monitor replication controllers and create pods to maintain desired state.
* **etcd** – It is a Key value pair data base. It stores configuration data of cluster and cluster state.
* **Kubectl utility** – It is a command line utility which connects to API Server on port 6443. It is used by administrators to create pods, services etc.

#### **On Worker Nodes following components will be installed**

* **Kubelet** – It is an agent which runs on every worker node, it connects to docker  and takes care of creating, starting, deleting containers.
* **Kube-Proxy** – It routes the traffic to appropriate containers based on ip address and port number of the incoming request. In other words we can say it is used for port translation.
* **Pod** – Pod can be defined as a multi-tier or group of containers that are deployed on a single worker node or docker host.

#### **Step 1: Disable SELinux & setup firewall rules**

**BELOW COMMANDS TO BE EXECUTED ON ALL THE 3 NODES (1 MASTER AND 2 CLIENT NODES)**

Login to your kubernetes master node and set the hostname and disable selinux using following commands

exec bash

setenforce 0

sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux

Set the following firewall rules.

firewall-cmd --permanent --add-port=6443/tcp

firewall-cmd --permanent --add-port=2379-2380/tcp

firewall-cmd --permanent --add-port=10250/tcp

firewall-cmd --permanent --add-port=10251/tcp

firewall-cmd --permanent --add-port=10252/tcp

firewall-cmd --permanent --add-port=10255/tcp

firewall-cmd --reload

modprobe br\_netfilter

[root@k8s-master ~]# echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables

**Note:** In case you don’t have your own dns server then update **/etc/hosts** file on master and worker nodes

192.168.1.101 k8s-master

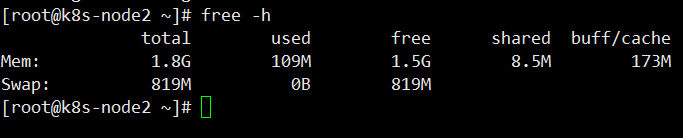
192.168.1.102 k8s-node1

192.168.1.103 k8s-node2

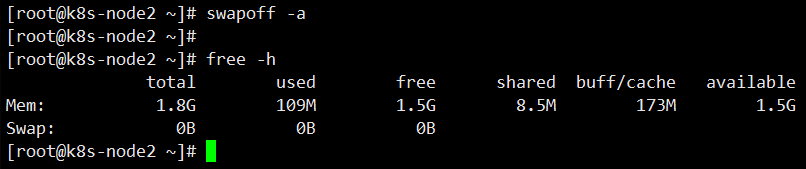
**Note: -- In this scenario, we have the private network as “192.168.1.0/24” for this lab setup.**

**Please check the network in which you are working and accordingly, change the “4th Octect”.**

Also, switch off the swap on all the 3 virtual machines as a requirement for Kubernetes Cluster

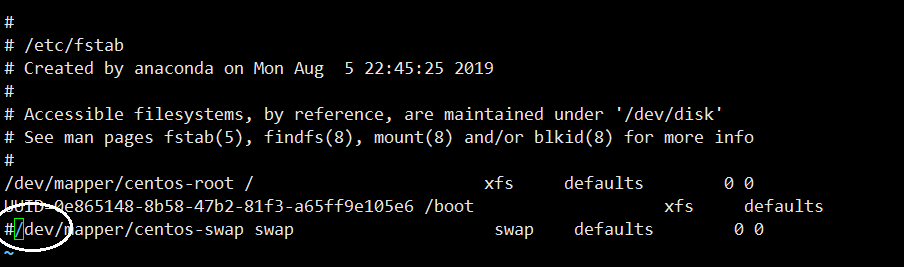


swapoff -a



Let’s make this config permanent, so that for the next reboot of the VM, the swap would be still off.

For that we would need to edit the **“/etc/fstab”** file



And either “delete” the line that say swap or comment the line with “#” in the beginning of the line.

And Save and quit 🡪 “:wq”

#### **Step 2: Configure Kubernetes Repository**

**BELOW COMMANDS TO BE EXECUTED ON ALL THE 3 NODES (1 MASTER AND 2 CLIENT NODES)**

Kubernetes packages are not available in the default CentOS 7 & RHEL 7 repositories, Use below command to configure its package repositories.

[root@k8s-master ~]# vi /etc/yum.repos.d/kubernetes.repo

[kubernetes]

name=Kubernetes

baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86\_64

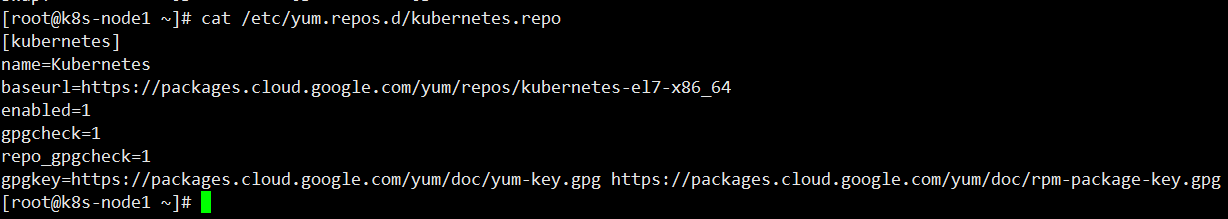
enabled=1

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg

[root@k8s-master ~]#



Make sure the alignment and space are exactly the same as the above screen shot, if not the **kubeadm** packages would not get installed, as the repo may not be recognized.

#### **Step 3: Install Kubeadm and Docker**

**BELOW COMMANDS TO BE EXECUTED ON ALL THE 3 NODES (1 MASTER AND 2 CLIENT NODES)**

Once the package repositories are configured, run the beneath command to install kubeadm and docker packages.

### Install Docker CE

Install the latest version of Docker-ce from the docker repository.

Install the package dependencies for docker-ce.

[root@k8s-master ~]# yum install -y yum-utils device-mapper-persistent-data lvm2

Add the docker repository to the system and install docker-ce using the yum command.

**[root@k8s-master ~]# *yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo***

[root@k8s-master ~]# yum install kubeadm docker-ce -y

Start and enable docker service

[root@k8s-master ~]# systemctl restart docker && systemctl enable docker

#### **Step 4: Initialize Kubernetes Master with ‘kubeadm init’**

**BELOW COMMANDS TO BE EXECUTED ONLY ON MASTER NODE**

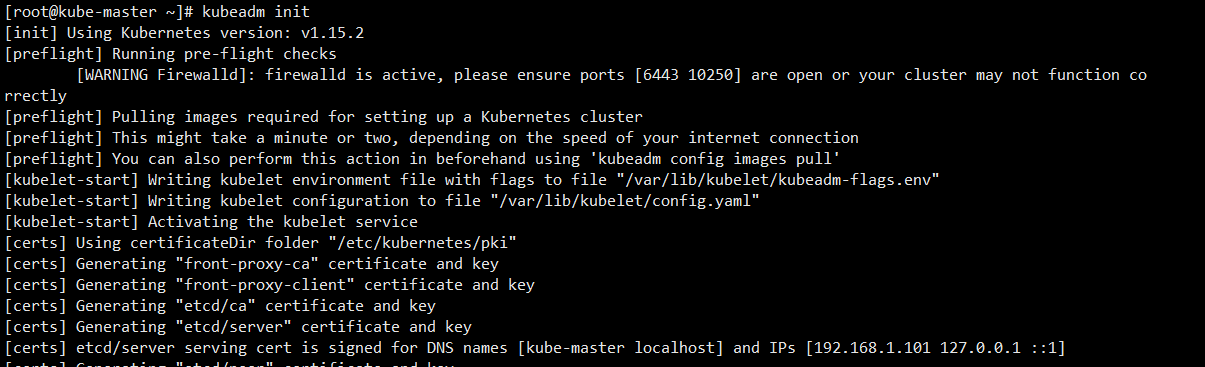
Start and enable kubectl service

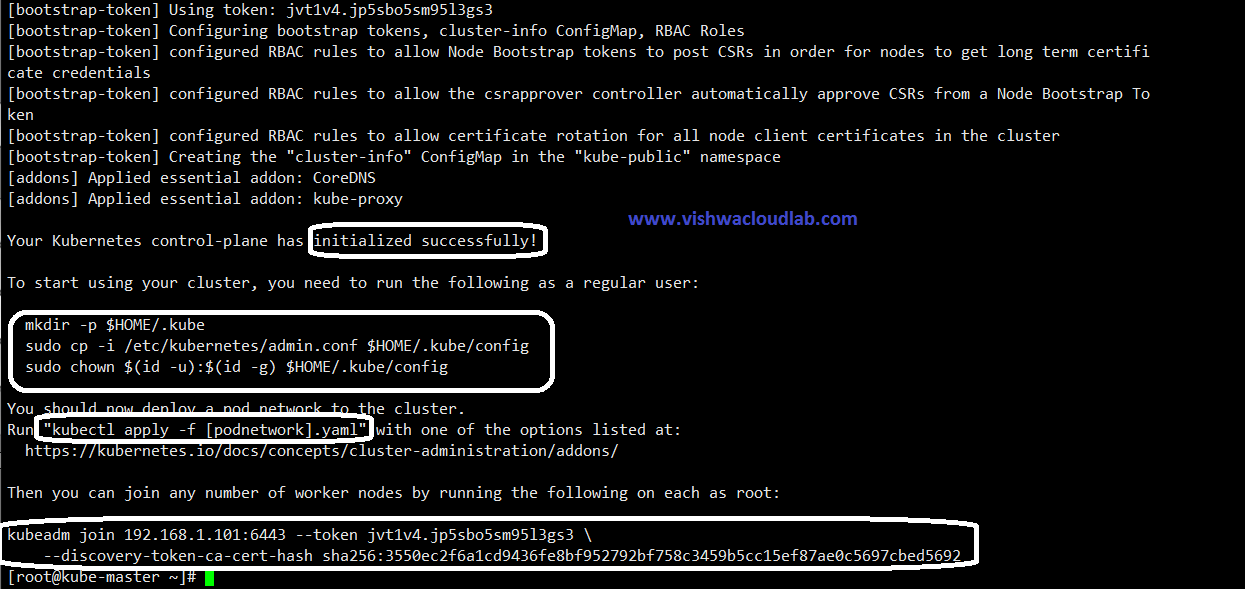
[root@k8s-master ~]# systemctl  restart kubelet && systemctl enable kubelet

Run the beneath command to  initialize and setup kubernetes master.

[root@kube-master ~]# kubeadm init

Output of above command would be something like below





As we can see in the output that kubernetes master has been initialized successfully. Execute the beneath commands to use the cluster as root user.

[root@kube-master ~]# mkdir -p $HOME/.kube

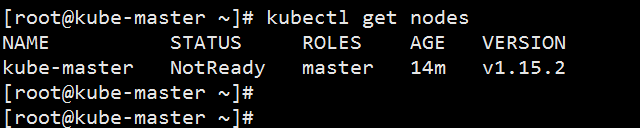
[root@kube-master ~]# cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

[root@kube-master ~]# chown $(id -u):$(id -g) $HOME/.kube/config

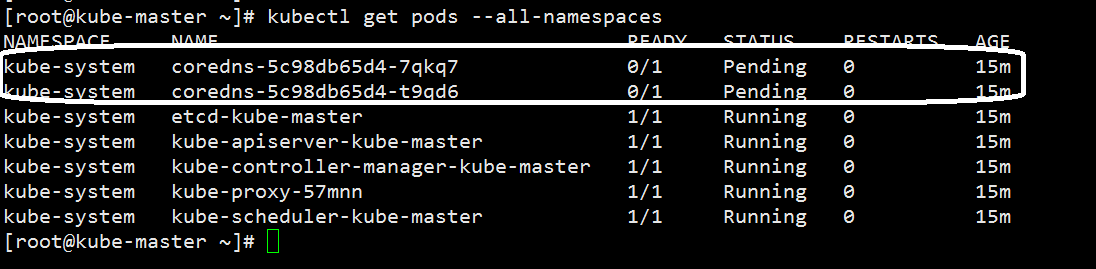
#### **Step 5: Deploy pod network to the cluster**

Lets try to run below commands to get status of cluster and pods.

[root@kube-master ~]# kubectl get nodes



[root@kube-master ~]# kubectl get pods --all-namespaces



To make the cluster status ready and kube-dns status running, deploy the pod network so that containers of different host communicated each other.

**POD** network is the overlay network between the worker nodes.

[root@kube-master ~]# export kubever=$(kubectl version | base64 | tr -d '\n')

[root@kube-master ~]# kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"

serviceaccount "weave-net" created

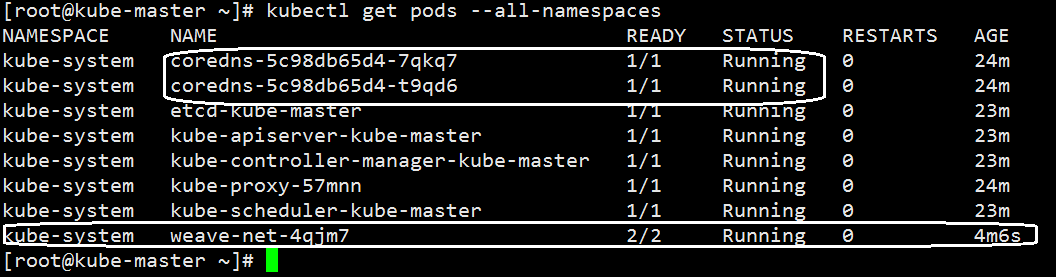
clusterrole "weave-net" created

clusterrolebinding "weave-net" created

daemonset "weave-net" created

[root@kube-master ~]#

Now, lets chk the DNS pods again,



**Note: -- This take approx. 2 -5 min sometimes. Please be patient…**

Now, that the Kube Master is ready, lets configure the 2 worker nodes for this cluster.

### **Step 6: Perform the following steps on each worker node**

#### **Step A: Disable SELinux & configure firewall rules on both the nodes**

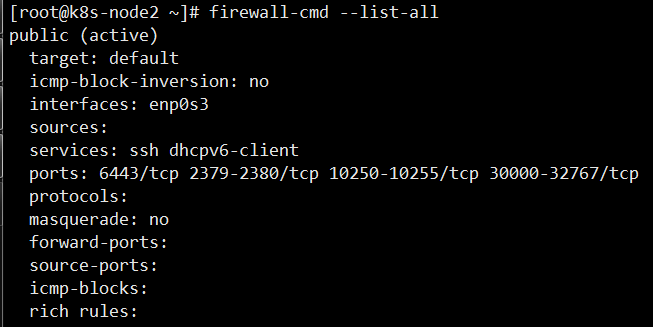
Add these firewall ports which is for the worker nodes only on the both nodes as ‘k8s-node1’ and ‘k8s-node2’ respectively

~]# firewall-cmd --permanent --add-port=30000-32767/tcp

~]#

~]# firewall-cmd  --reload

**Output: To chk the firewall config**



#### **Step B: Configure Kubernetes Repositories on both worker nodes**

~]# vi /etc/yum.repos.d/kubernetes.repo

[kubernetes]

name=Kubernetes

baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86\_64

enabled=1

gpgcheck=1

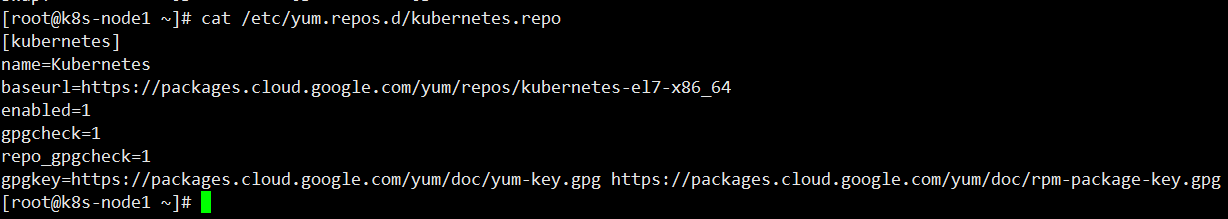
repo\_gpgcheck=1

gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg

Press -- > “esc”

And Then 🡪 “:wq”

TO save the files.



Make sure the alignment and space are exactly the same as the above screen shot, if not the **kubeadm** packages would not get installed, as the repo may not be recognized.

#### **Step C: Install kubeadm and docker package on both worker nodes**

### Install Docker CE

Install the latest version of Docker-ce from the docker repository.

Install the package dependencies for docker-ce.

*yum install -y yum-utils device-mapper-persistent-data lvm2*

Add the docker repository to the system and install docker-ce using the yum command.

*yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo  
yum install -y docker-ce*

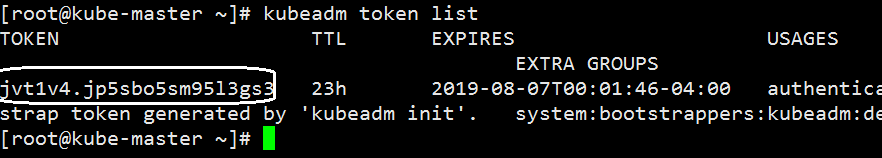
[root@k8s-node1 ~]# yum  install kubeadm -y

[root@k8s-node1 ~]# systemctl restart docker && systemctl enable docker

#### **Step D: Now Join worker nodes to master node**

To join worker nodes to Master node, a token is required. Whenever kubernetes master initialized , then in the output we get command and token.  Copy that command and run on both nodes.

To get the token on the master

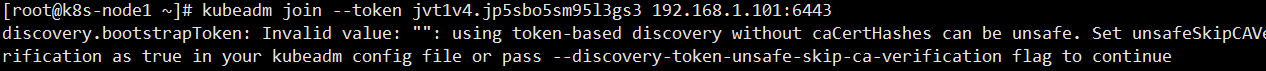


Now, with the join command

[root@k8s-node1 ~]# kubeadm join --token jvt1v4.jp5sbo5sm95l3gs3 192.168.1.101:6443

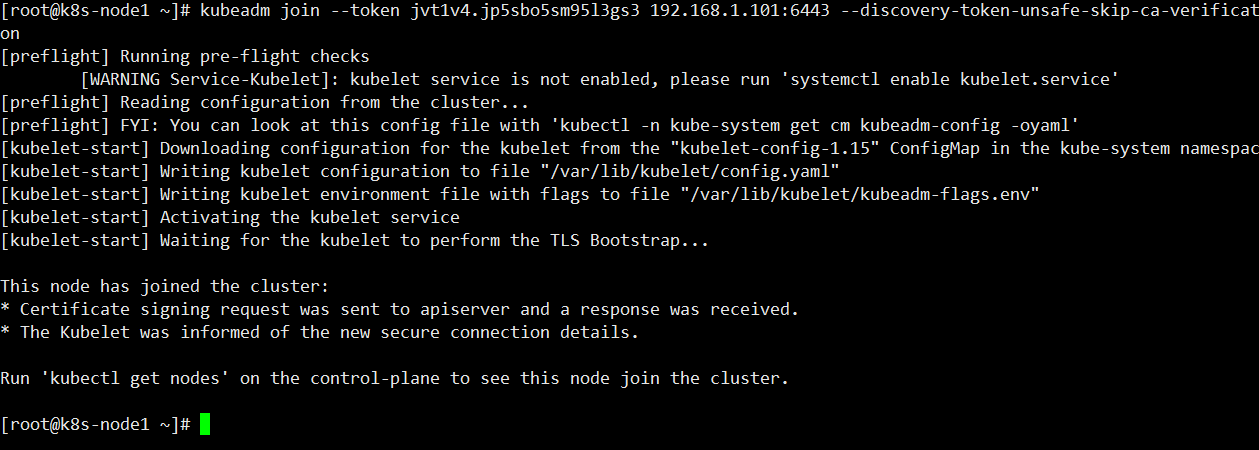
Here, the Ip address of the master Kube.

**Note—you make get an warning**



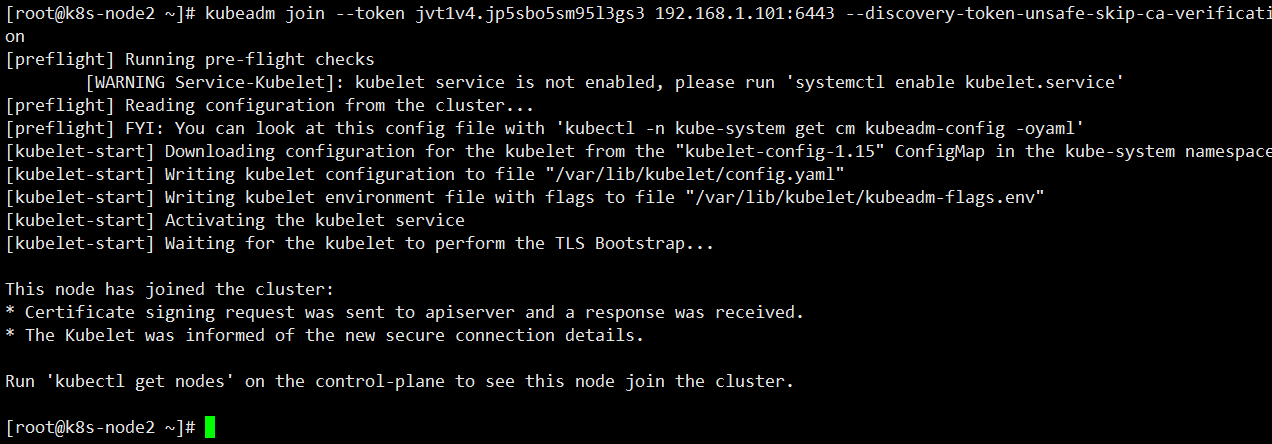
Saying that token is without the certificates and it might be unsafe.

[root@k8s-node1 ~]# kubeadm join --token jvt1v4.jp5sbo5sm95l3gs3 192.168.1.101:6443 -–discovery-token-unsafe-skip-ca-verification



This shows that the node is successfully joined the cluster.

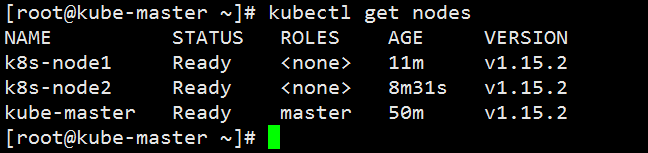
Similarly for Node2.



Now verify Nodes status from master node using kubectl command

[root@kube-master ~]# kubectl get nodes

Note: -- It takes around 5 - 9 min for all the nodes to show as “**Ready**” .



As we can see master and worker nodes are in ready status.

This concludes that kubernetes 1.7 has been installed successfully and also we have successfully joined two worker nodes.

Now we can create pods and services on this Kube Cluster.