**Prerequisites**

* 2 CentOS 7 Servers
  + *192.168.11.143      k8-master*
  + *192.168.11.144      k8-worker*
* Root privileges

**What we will do?**

1. Kubernetes Installation
2. Kubernetes Cluster Initialization
3. Adding k8-worker and node02 to the Cluster
4. Testing - Create First Pod

**Step 1 - Kubernetes Installation**

In this first step, we will prepare those 3 servers for Kubernetes installation, so run all commands on the master and node servers.

We will prepare all servers for Kubernetes installation by changing the existing configuration on servers, and also installating some packages, including docker-ce and kubernetes itself.

**- Configure Hosts**

Edit hosts file on all server using the vim editor.

vim /etc/hosts

Paste the host's list below.

192.168.11.143      k8-master  
192.168.11.144      k8-worker

Save and exit.

**- Disable SELinux**

In this tutorial, we will not cover about SELinux configuration for Docker, so we will disable it.

Run the command below to disable SELinux.

setenforce 0  
sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux

**- Enable br\_netfilter Kernel Module**

The br\_netfilter module is required for kubernetes installation. Enable this kernel module so that the packets traversing the bridge are processed by iptables for filtering and for port forwarding, and the kubernetes pods across the cluster can communicate with each other.

Run the command below to enable the br\_netfilter kernel module.

modprobe br\_netfilter  
echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables

**- Disable SWAP**

Disable SWAP for kubernetes installation by running the following commands.

swapoff -a

And then edit the '/etc/fstab' file.

vim /etc/fstab

Comment the swap line UUID as below.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/1b.png)

**- Install Docker CE**

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

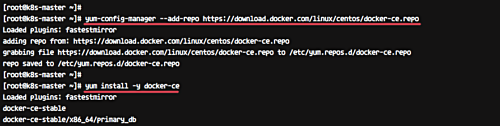
Install the package dependencies for docker-ce.

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

Wait for the docker-ce installation.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/2.png)

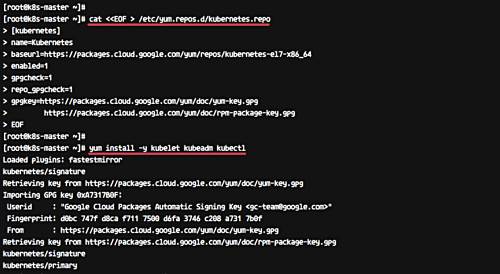
**- Install Kubernetes**

Add the kubernetes repository to the centos 7 system by running the following command.

cat <<EOF > /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  
EOF

Now install the kubernetes packages kubeadm, kubelet, and kubectl using the yum command below.

yum install -y kubelet kubeadm kubectl

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/3.png)

After the installation is complete, restart all those servers.

sudo reboot

Log in again to the server and start the services, docker and kubelet.

systemctl start docker && systemctl enable docker  
systemctl start kubelet && systemctl enable kubelet

**- Change the cgroup-driver**

We need to make sure the docker-ce and kubernetes are using same 'cgroup'.

Check docker cgroup using the docker info command.

docker info | grep -i cgroup

And you see the docker is using '**cgroupfs**' as a cgroup-driver.

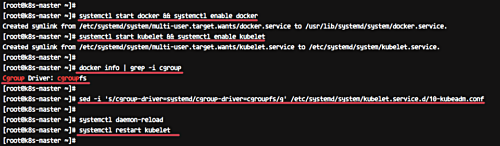
Now run the command below to change the kuberetes cgroup-driver to 'cgroupfs'.

sed -i 's/cgroup-driver=systemd/cgroup-driver=cgroupfs/g' /etc/systemd/system/kubelet.service.d/10-kubeadm.conf

Reload the systemd system and restart the kubelet service.

systemctl daemon-reload  
systemctl restart kubelet

Now we're ready to configure the Kubernetes Cluster.

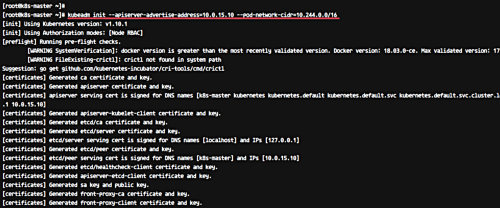
[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/4.png)

**Step 2 - Kubernetes Cluster Initialization**

In this step, we will initialize the kubernetes master cluster configuration.

Move the shell to the master server 'k8-master' and run the command below to set up the kubernetes master.

kubeadm init --apiserver-advertise-address=192.168.11.143 --pod-network-cidr=192.168.0.0/16

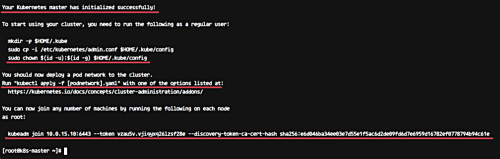
**[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/6.png)**

**Note:**

*--apiserver-advertise-address* = determines which IP address Kubernetes should advertise its API server on.

*--pod-network-cidr* = specify the range of IP addresses for the pod network. We're using the 'flannel' virtual network. If you want to use another pod network such as weave-net or calico, change the range IP address.

When the Kubernetes initialization is complete, you will get the result as below.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/7.png)

**Note:**

Copy the '**kubeadm join ... ... ...**' command to your text editor. The command will be used to register new nodes to the kubernetes cluster.

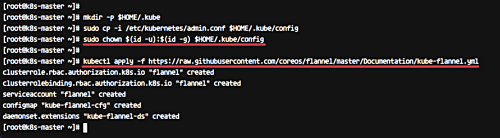
Now in order to use Kubernetes, we need to run some commands as on the result.

Create new '.kube' configuration directory and copy the configuration 'admin.conf'.

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

Next, deploy the flannel network to the kubernetes cluster using the kubectl command.

kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/8.png)

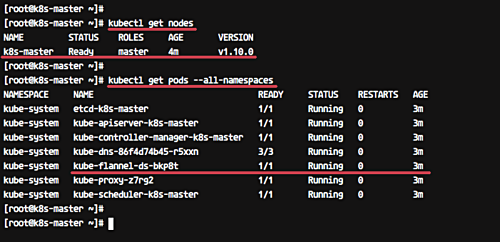
The flannel network has been deployed to the Kubernetes cluster.

Wait for a minute and then check kubernetes node and pods using commands below.

kubectl get nodes  
kubectl get pods --all-namespaces

And you will get the 'k8-master' node is running as a 'master' cluster with status 'ready', and you will get all pods that are needed for the cluster, including the 'kube-flannel-ds' for network pod configuration.

Make sure all kube-system pods status is 'running'.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/9.png)

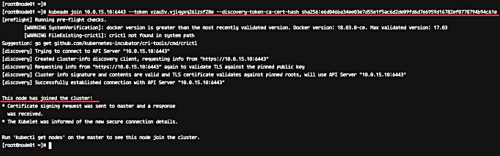
Kubernetes cluster master initialization and configuration has been completed.

**Step 3 - Adding k8-worker and node02 to the Cluster**

In this step, we will add k8-worker and node02 to join the 'k8s' cluster.

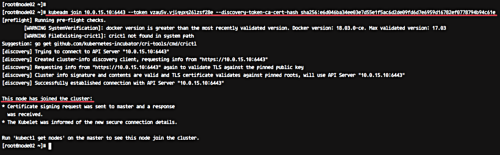
Connect to the k8-worker server and run the kubeadm join command as we copied on the top.

kubeadm join 192.168.11.143:6443 --token vzau5v.vjiqyxq26lzsf28e --discovery-token-ca-cert-hash sha256:e6d046ba34ee03e7d55e1f5ac6d2de09fd6d7e6959d16782ef0778794b94c61e

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/10.png)

Connect to the node02 server and run the kubeadm join command as we copied on the top.

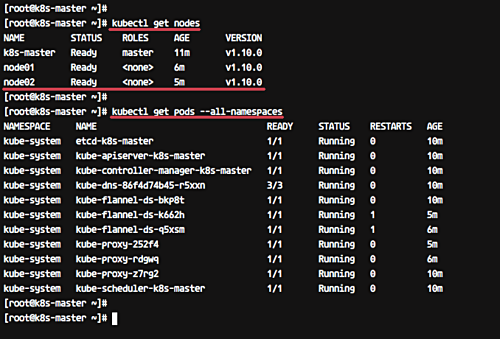
kubeadm join 192.168.11.143:6443 --token vzau5v.vjiqyxq26lzsf28e --discovery-token-ca-cert-hash sha256:e6d046ba34ee03e7d55e1f5ac6d2de09fd6d7e6959d16782ef0778794b94c61e

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/11.png)

Wait for some minutes and back to the 'k8-master' master cluster server check the nodes and pods using the following command.

kubectl get nodes  
kubectl get pods --all-namespaces

Now you will get k8-worker and node02 has been added to the cluster with status 'ready'.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/12.png)

k8-worker and node02 have been added to the kubernetes cluster.

**Step 4 - Testing Create First Pod**

In this step, we will do a test by deploying the Nginx pod to the kubernetes cluster. A pod is a group of one or more containers with shared storage and network that runs under Kubernetes. A Pod contains one or more containers, such as Docker container.

Login to the 'k8-master' server and create new deployment named 'nginx' using the kubectl command.

kubectl create deployment nginx --image=nginx

To see details of the 'nginx' deployment sepcification, run the following command.

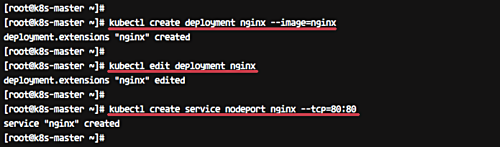
kubectl describe deployment nginx

And you will get the nginx pod deployment specification.

Next, we will expose the nginx pod accessible via the internet. And we need to create new service NodePort for this.

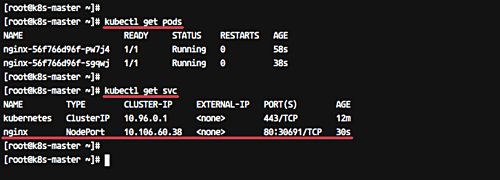
Run the kubectl command below.

kubectl create service nodeport nginx --tcp=80:80

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/13.png)

Make sure there is no error. Now check the nginx service nodeport and IP using the kubectl command below.

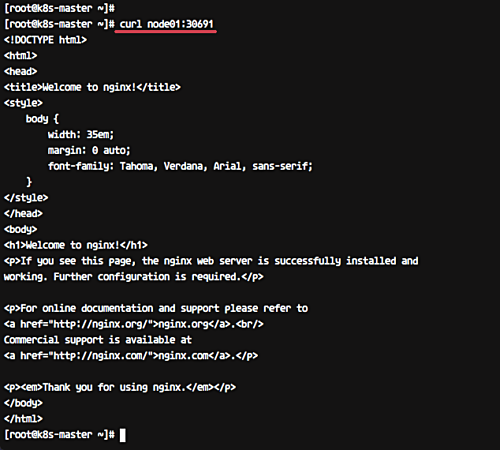
kubectl get pods  
kubectl get svc

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/14.png)

Now you will get the nginx pod is now running under cluster IP address '10.160.60.38' port 80, and the node main IP address '10.0.15.x' on port '30691'.

From the 'k8-master' server run the [curl command](https://www.howtoforge.com/community/threads/how-to-install-curl.20374/) below.

curl k8-worker:30691

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/15.png)

The Nginx Pod has now been deployed under the Kubernetes cluster and it's accessible via the internet.

Now access from the web browser.

***http://192.168.11.143:30691/***

And you will get the Nginx default page.

[](https://www.howtoforge.com/images/how_to_set_up_kubernetes_cluster_on_centos_7/big/17.png)

The Kubernetes cluster Installation and configuration on CentOS 7 has been completed successfully.

Check UP:-

Find the pods cluster IP: kubectl get pod -o wide

Finding the service IP: kubectl get service --all-namespaces

Finding and Entering Pod Network Namespaces: docker ps

docker inspect --format '{{ .State.Pid }}' container-id-or-name