

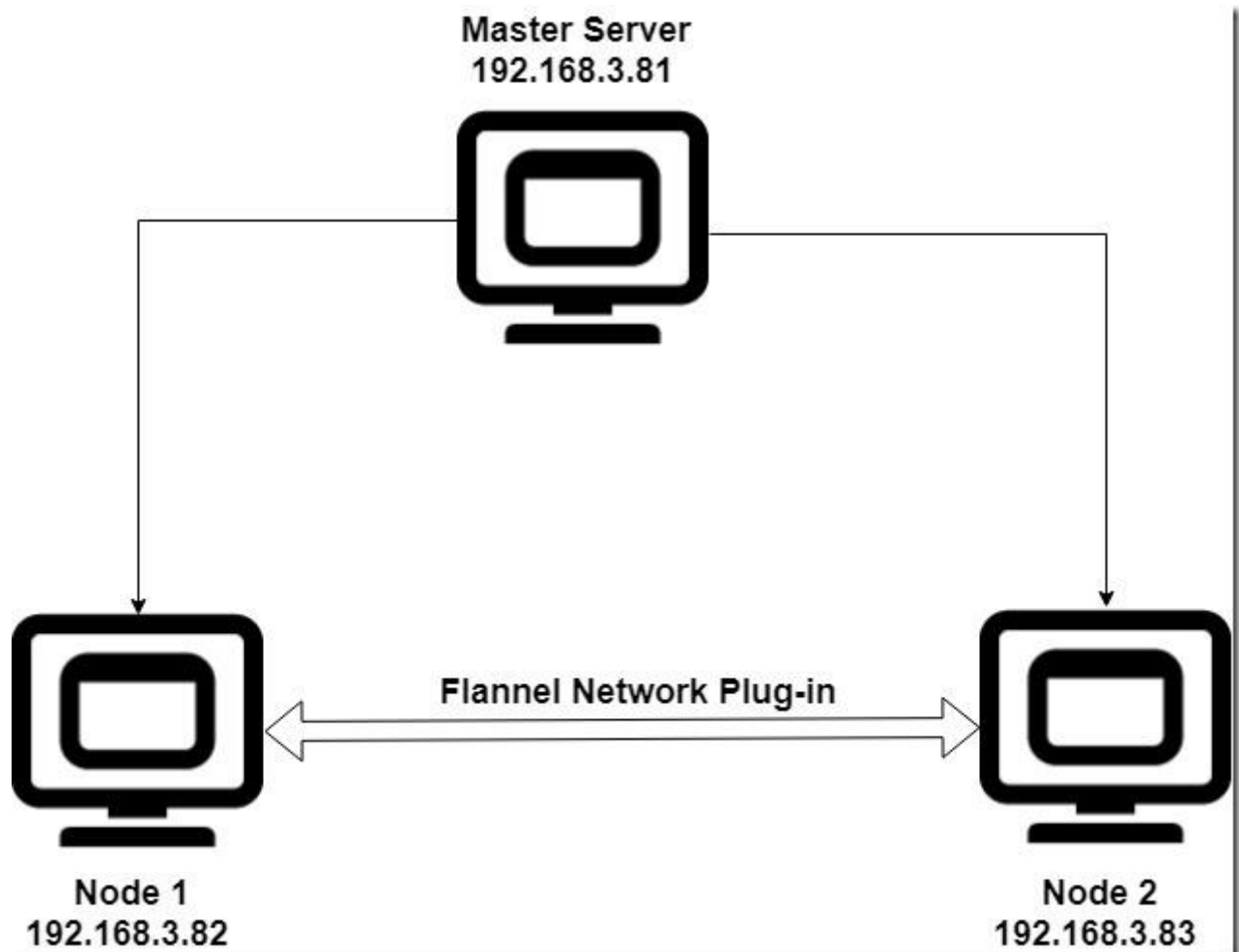
# Steps to install kubernetes cluster manually using CENTOS 7

In this blog, we will show you the Steps to install kubernetes cluster manually using CENTOS 7.

## REQUIREMENTS

- 3 t2.medium CentOS 7 with an internet connection.
- Kubernetes components

## INFRASTRUCTURE OVERVIEW



- We are creating two node cluster for this demo.
- The Master IP will be 192.168.3.81.
- Node 1 IP will be 192.68.3.82 and Node 2 IP will be 192.168.3.83.
- We are using the **Flannel Network** for the POD communication in this demo.

*Note: The VM IP's may change based on your environment*

## MASTER SERVER CONFIGURATION

- Log in to the master server and we have already set the hostname as **k8s-master** during OS installation.

```
root@k8s-master:~
[root@k8s-master ~]# hostname
k8s-master
[root@k8s-master ~]#
```

- Disable the SELinux using the below commands.

**exec bash**

**setenforce 0**

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

```
root@k8s-master:~
[root@k8s-master ~]# exec bash
[root@k8s-master ~]# setenforce 0
[root@k8s-master ~]# sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g'
[root@k8s-master ~]#
```

- Open the **sysctl.conf** file.

**vi /etc/sysctl.conf**

```
root@k8s-master:~
[root@k8s-master ~]# vi /etc/sysctl.conf
```

- Add the below entries in the conf file to change the Linux host bridge values and save the changes.

**net.bridge.bridge-nf-call-ip6tables = 1**

**net.bridge.bridge-nf-call-iptables = 1**

```

root@k8s-master:~
# System default settings live in /usr/lib/sysctl.d/00-system.conf.
# To override those settings, enter new settings here, or in an /etc/sysctl.d/<name>.conf.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).

net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1

```

- open the fstab file.

**vi /etc/fstab**

```

root@k8s-master:~
[root@k8s-master ~]# vi /etc/fstab

```

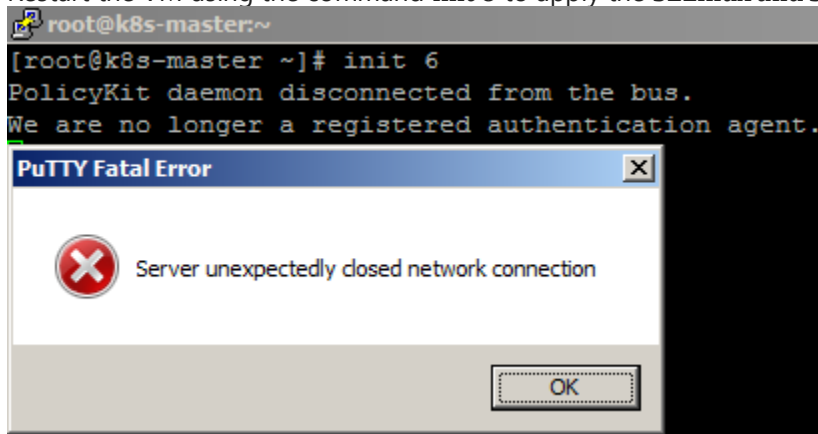
- Disable the SWAP by adding the # symbol at the beginning and save the changes.

```

root@k8s-master:~
#
# /etc/fstab
# Created by anaconda on Thu Mar  1 05:41:24 2018
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root / xfs defaults 0 0
UUID=0aad7414-bc94-4120-92c4-fd96022fe14d /boot xfs defaults
# /dev/mapper/centos-swap swap swap defaults 0 0

```

- Restart the VM using the command **init 6** to apply the **SELinux** and **SWAP** changes.



- Once the VM is back to online, open the host file.

**vi /etc/hosts**

```
root@k8s-master:~  
[root@k8s-master ~]# vi /etc/hosts
```

- Add the below entries in the host file and save the changes.

192.168.3.81 k8s-master

192.168.3.82 k8s-node1

192.168.3.83 k8s-node2

```
root@k8s-master:~  
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4  
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6  
192.168.3.81 k8s-master  
192.168.3.82 k8s-node1  
192.168.3.83 k8s-node2
```

*Note: The IP's may change based on your environment*

- Create a new file named **kubernetes.repo** under **yum.repos.d** folder using the below command.

vi /etc/yum.repos.d/kubernetes.repo

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

- Add the below entries and save the changes.

[kubernetes]

name=Kubernetes

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

```

## INSTALLING DOCKER AND KUBEADM

- Now, Install the **docker** and **kubeadm** using the below command.

**yum install -y kubeadm docker**

```

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

```

- It will take few minutes to complete the installation.

```

Updated:
  dracut.x86_64 0:033-502.el7_4.1                selinux-policy-targeted.noarch 0:3.13.1

Dependency Updated:
  audit.x86_64 0:2.7.6-3.el7                      audit-libs.x86_64 0:2.7.6-3.el7
  dracut-network.x86_64 0:033-502.el7_4.1          libgudev1.x86_64 0:219-42.el7_4.7
  libselinux-utils.x86_64 0:2.5-11.el7              libsemanage.x86_64 0:2.5-8.el7
  polycoreutils.x86_64 0:2.5-17.1.el7              selinux-policy.noarch 0:3.13.1-166.e
  systemd-sysv.x86_64 0:219-42.el7_4.7

Complete!
[root@k8s-master ~]#

```

- Enable and start the docker and kubelet services using commands.

**systemctl enable docker && systemctl enable kubelet**

**systemctl start docker && systemctl start kubelet**

```
root@k8s-master:~  
[root@k8s-master ~]# systemctl enable docker && systemctl enable kubelet  
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr  
Created symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /et  
[root@k8s-master ~]#  
[root@k8s-master ~]#  
[root@k8s-master ~]# systemctl start docker && systemctl start kubelet  
[root@k8s-master ~]#
```

## NODE(S) CONFIGURATION

- We have already set the hostname for **node1** and **node2** during OS installation.

```
root@k8s-node1:~  
[root@k8s-node1 ~]# hostname  
k8s-node1  
[root@k8s-node1 ~]#  
  
root@k8s-node2:~  
[root@k8s-node2 ~]# hostname  
k8s-node2  
[root@k8s-node2 ~]#
```

- Please follow the above **master server** configuration steps for the kubernetes node preparation. We have already installed the docker and kubeadm on both the nodes.

root@k8s-node1:~

```
Verifying : systemd-219-19.el7.x86_64
Verifying : selinux-policy-3.13.1-60.el7.noarch
Verifying : libselinux-utils-2.2.2-6.el7.x86_64
Verifying : libsepol-2.1.9-3.el7.x86_64
Verifying : libselinux-2.2.2-6.el7.x86_64
Verifying : dracut-network-033-359.el7.x86_64
Verifying : systemd-libs-219-19.el7.x86_64
Verifying : audit-libs-2.4.1-5.el7.x86_64
Verifying : policycoreutils-2.2.5-20.el7.x86_64
```

Installed:

docker.x86\_64 2:1.12.6-71.git3e8e77d.el7.centos.1

kubeadm.x86\_64 1:1.12.6-71.git3e8e77d.el7.centos.1

Dependency Installed:

```
audit-libs-python.x86_64 0:2.7.6-3.el7
container-selinux.noarch 2:2.36-1.gitff95335.el7
docker-client.x86_64 2:1.12.6-71.git3e8e77d.el7.centos.1
ebtables.x86_64 0:2.0.10-15.el7
kubenet.x86_64 0:1.9.3-0
libcgrouper.x86_64 0:0.41-13.el7
libselinux-python.x86_64 0:2.5-11.el7
oci-register-machine.x86_64 1:0-3.14.gitcd1e331.el7
oci-umount.x86_64 2:2.3.1-2.gitbf16163.el7
python-IPy.noarch 0:0.75-6.el7
skopeo-containers.x86_64 1:0.1.26-2.dev.git2e8377a.el7.centos
yajl.x86_64 0:2.0.4-4.el7
```

```
checkpolicy.x86_64 0:2.2.2-6.el7
container-storage-engine.x86_64 1:1.26-1.gitff95335.el7
docker-common.x86_64 2:1.12.6-71.git3e8e77d.el7.centos.1
kubect.x86_64 0:1.12.6-71.git3e8e77d.el7.centos.1
kubernetes-cni.x86_64 0:1.12.6-71.git3e8e77d.el7.centos.1
libseccomp.x86_64 0:2.5-11.el7
libsemanage-python.x86_64 0:2.5-11.el7
oci-systemd-hook.x86_64 1:0-3.14.gitcd1e331.el7
policycoreutils-python.x86_64 0:2.2.5-20.el7
setools-libs.x86_64 0:3.13.1-166.el7
socat.x86_64 0:1.7.3-2.el7
```

Updated:

dracut.x86\_64 0:033-502.el7\_4.1

selinux-policy-targeted.noarch 0:3.13.1-166.el7\_4.7

Dependency Updated:

```
audit.x86_64 0:2.7.6-3.el7
dracut-network.x86_64 0:033-502.el7_4.1
libselinux-utils.x86_64 0:2.2.2-6.el7
policycoreutils.x86_64 0:2.2.5-20.el7
systemd-sysv.x86_64 0:219-42.el7_4.7
```

```
audit-libs.x86_64 0:2.7.6-3.el7
libgudev1.x86_64 0:219-42.el7_4.7
libsemanage.x86_64 0:2.5-8.el7
selinux-policy.noarch 0:3.13.1-166.el7_4.7
```

Complete!

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

Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service

Created symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /etc/systemd/system/kubelet.service

[root@k8s-node1 ~]# systemctl start docker && systemctl start kubelet

[root@k8s-node1 ~]#



```

root@k8s-node2:~
Verifying : systemd-219-19.el7.x86_64
Verifying : selinux-policy-3.13.1-60.el7.noarch
Verifying : libselinux-utils-2.2.2-6.el7.x86_64
Verifying : libsepol-2.1.9-3.el7.x86_64
Verifying : libselinux-2.2.2-6.el7.x86_64
Verifying : dracut-network-033-359.el7.x86_64
Verifying : systemd-libs-219-19.el7.x86_64
Verifying : audit-libs-2.4.1-5.el7.x86_64
Verifying : policycoreutils-2.2.5-20.el7.x86_64

Installed:
  docker.x86_64 2:1.12.6-71.git3e8e77d.el7.centos.1
  kubeadm.x86_64 0:1.12.6-71.git3e8e77d.el7.centos.1

Dependency Installed:
  audit-libs-python.x86_64 0:2.7.6-3.el7
  container-selinux.noarch 2:2.36-1.gitff95335.el7
  docker-client.x86_64 2:1.12.6-71.git3e8e77d.el7.centos.1
  ebttables.x86_64 0:2.0.10-15.el7
  kubelet.x86_64 0:1.9.3-0
  libcgroupp.x86_64 0:0.41-13.el7
  libselinux-python.x86_64 0:2.5-11.el7
  oci-register-machine.x86_64 1:0-3.14.gitcd1e331.el7
  oci-umount.x86_64 2:2.3.1-2.gitbf16163.el7
  python-IPy.noarch 0:0.75-6.el7
  skopeo-containers.x86_64 1:0.1.26-2.dev.git2e8377a.el7.centos
  yajl.x86_64 0:2.0.4-4.el7
  checkpolicy.x86_64 0:2.2.5-20.el7.x86_64
  container-storage-setup.x86_64 0:1.26-1.gitff95335.el7
  docker-common.x86_64 2:1.12.6-71.git3e8e77d.el7.centos.1
  kubect1.x86_64 0:1.9.3-0
  kubernetes-cni.x86_64 0:1.9.3-0
  libseccomp.x86_64 0:2.11.2-1.el7.x86_64
  libsemanage-python.x86_64 0:1.4.5-4.el7.x86_64
  oci-systemd-hook.x86_64 1:0-3.14.gitcd1e331.el7
  policycoreutils-python.x86_64 0:2.2.5-20.el7.x86_64
  setools-libs.x86_64 0:3.1.7-1.el7.x86_64
  socat.x86_64 0:1.7.3-2.el7.x86_64

Updated:
  dracut.x86_64 0:033-502.el7_4.1
  selinux-policy-targeted.noarch 0:3.13.1-166.el7_4.7

Dependency Updated:
  audit.x86_64 0:2.7.6-3.el7
  audit-libs.x86_64 0:2.7.6-3.el7
  dracut-network.x86_64 0:033-502.el7_4.1
  libgudev1.x86_64 0:219-42.el7_4.7
  libselinux-utils.x86_64 0:2.2.2-6.el7.x86_64
  libsemanage.x86_64 0:2.5-8.el7
  policycoreutils.x86_64 0:2.2.5-20.el7.x86_64
  selinux-policy.noarch 0:3.13.1-166.el7_4.7
  systemd-sysv.x86_64 0:219-42.el7_4.7

Complete!
[root@k8s-node2 ~]# systemctl enable docker && systemctl enable kubelet
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service
Created symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /etc/systemd/system/kubelet.service
[root@k8s-node2 ~]# systemctl start docker && systemctl start kubelet
[root@k8s-node2 ~]#

```

## CREATING CLUSTER

- We are using the **Flannel network** for this demo.
- To make the flannel to working properly, we need to specify the network CIDR while configuring the cluster. Use the below command to create a cluster.

**kubeadm init --pod-network-cidr=10.244.0.0/16**

```

root@k8s-master:~
[root@k8s-master ~]# kubeadm init --pod-network-cidr=10.244.0.0/16

```

- It will take few minutes to complete the configuration process.



```

root@k8s-master:~#
[root@k8s-master ~]# kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.9.3
[init] Using Authorization modes: [Node RBAC]
[preflight] Running pre-flight checks.
      [WARNING FileExisting-crictl]: crictl not found in system path
[certificates] Generated ca certificate and key.
[certificates] Generated apiserver certificate and key.
[certificates] apiserver serving cert is signed for DNS names [k8s-master kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.3.81]
[certificates] Generated apiserver-kubelet-client certificate and key.
[certificates] Generated sa key and public key.
[certificates] Generated front-proxy-ca certificate and key.
[certificates] Generated front-proxy-client certificate and key.
[certificates] Valid certificates and keys now exist in "/etc/kubernetes/pki"
[kubeconfig] Wrote KubeConfig file to disk: "admin.conf"
[kubeconfig] Wrote KubeConfig file to disk: "kubelet.conf"
[kubeconfig] Wrote KubeConfig file to disk: "controller-manager.conf"
[kubeconfig] Wrote KubeConfig file to disk: "scheduler.conf"
[controlplane] Wrote Static Pod manifest for component kube-apiserver to "/etc/kubernetes/manifests/kube-apiserver.yaml"
[controlplane] Wrote Static Pod manifest for component kube-controller-manager to "/etc/kubernetes/manifests/kube-controller-manager.yaml"
[controlplane] Wrote Static Pod manifest for component kube-scheduler to "/etc/kubernetes/manifests/kube-scheduler.yaml"
[etcd] Wrote Static Pod manifest for a local etcd instance to "/etc/kubernetes/manifests/etcd.yaml"
[init] Waiting for the kubelet to boot up the control plane as Static Pods from directory "/etc/kubernetes/manifests".
[init] This might take a minute or longer if the control plane images have to be pulled.

```

- Kubernetes cluster has configured successfully.

```

[bootstraptoken] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get CSRs in order for nodes to get CSRs in order for nodes to get CSRs
[bootstraptoken] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from CSRs in order for nodes to get CSRs
[bootstraptoken] Configured RBAC rules to allow certificate rotation for all node client certificates in the CSRs in order for nodes to get CSRs
[bootstraptoken] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[addons] Applied essential addon: kube-dns
[addons] Applied essential addon: kube-proxy

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join --token a69705.533d90e77d47e2b2 192.168.3.81:6443 --discovery-token-ca-cert-hash sha256:6a23
  415c3338f1

[root@k8s-master ~]#

```

- Execute the below commands below start using the cluster.

```
mkdir -p $HOME/.kube
```

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

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

```
[root@k8s-master ~]# mkdir -p $HOME/.kube
[root@k8s-master ~]# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
[root@k8s-master ~]# sudo chown $(id -u):$(id -g) $HOME/.kube/config
[root@k8s-master ~]#
```

- Also, make a note of the kubeadm join command to add the nodes into the cluster.

```
kubeadm join --token a69705.533d90e77d47e2b2 192.168.3.81:6443 --discovery-token-ca-cert-hash sha256:6a23415c3338f1
```

- We can list the available system PODS using the below command.

**kubect1 get pods --all-namespaces**

```
root@k8s-master:~
[root@k8s-master ~]# kubect1 get pods --all-namespaces
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	etcd-k8s-master	1/1	Running	2	6m
kube-system	kube-apiserver-k8s-master	1/1	Running	5	6m
kube-system	kube-controller-manager-k8s-master	1/1	Running	3	6m
kube-system	kube-dns-6f4fd4bdf-9j7zp	0/3	Pending	0	27m
kube-system	kube-proxy-p2247	1/1	Running	2	27m
kube-system	kube-scheduler-k8s-master	1/1	Running	2	6m

```
[root@k8s-master ~]#
```

- Kube-DNS will be in pending state until we install the POD network for the cluster.

## INSTALLING NETWORK

- Use the below command to install the **Flannel** network.

**kubect1 apply -**

**f** <https://raw.githubusercontent.com/coreos/flannel/v0.9.1/Documentation/kube-flannel.yml>

```
root@k8s-master:~
[root@k8s-master ~]# kubect1 apply -f https://raw.githubusercontent.com/coreos/flannel/
```

- It will take few minutes to complete the installation.

```

root@k8s-master:~
[root@k8s-master ~]# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/
clusterrole "flannel" created
clusterrolebinding "flannel" created
serviceaccount "flannel" created
configmap "kube-flannel-cfg" created
daemonset "kube-flannel-ds" created
[root@k8s-master ~]#

```

- The flannel pod will start initiating and it will take few minutes to complete the configuration process.

```

root@k8s-master:~
[root@k8s-master ~]# kubectl get pods --all-namespaces

```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	etcd-k8s-master	0/1	Pending	0	4s
kube-system	kube-apiserver-k8s-master	0/1	Pending	0	3s
kube-system	kube-controller-manager-k8s-master	0/1	Pending	0	3s
kube-system	kube-dns-6f4fd4bdf-9j7zp	0/3	Pending	0	32m
kube-system	kube-flannel-ds-92bd7	0/1	Init:0/1	0	25s
kube-system	kube-proxy-p2247	1/1	NodeLost	2	32m
kube-system	kube-scheduler-k8s-master	0/1	Pending	0	3s

```

[root@k8s-master ~]#

```

- We can get the detailed information about a POD using **describe** option.

**kubectl describe pod <pod name> --namespace=kube-system**

```

[root@k8s-master ~]# kubectl describe pod kube-flannel-ds-92bd7 --namespace=kube-system

```

Type	Reason	Age	From	Message
Normal	SuccessfulMountVolume	7m	kubelet, k8s-master	MountVolume.SetUp succeeded
Normal	SuccessfulMountVolume	7m	kubelet, k8s-master	MountVolume.SetUp succeeded
Normal	SuccessfulMountVolume	7m	kubelet, k8s-master	MountVolume.SetUp succeeded
Normal	SuccessfulMountVolume	7m	kubelet, k8s-master	MountVolume.SetUp succeeded
Normal	Pulling	7m	kubelet, k8s-master	pulling image "quay.io/core"
Normal	Pulled	6m	kubelet, k8s-master	Successfully pulled image "
Normal	Created	6m	kubelet, k8s-master	Created container
Normal	Started	6m	kubelet, k8s-master	Started container
Normal	Pulled	6m	kubelet, k8s-master	Container image "quay.io/co"
Normal	Created	6m	kubelet, k8s-master	Created container
Normal	Started	6m	kubelet, k8s-master	Started container

```

[root@k8s-master ~]#

```

- After some time, all the system pods are in running state.

```

root@k8s-master:~
[root@k8s-master ~]# kubectl get pods --all-namespaces
NAMESPACE      NAME                                     READY   STATUS    RESTARTS   AGE
kube-system    etcd-k8s-master                       1/1     Running   2          7m
kube-system    kube-apiserver-k8s-master             1/1     Running   5          7m
kube-system    kube-controller-manager-k8s-master    1/1     Running   3          7m
kube-system    kube-dns-6f4fd4bdf-9j7zp             3/3     Running   0          40m
kube-system    kube-flannel-ds-92bd7                1/1     Running   0          8m
kube-system    kube-proxy-p2247                     1/1     Running   2          40m
kube-system    kube-scheduler-k8s-master             1/1     Running   2          7m
[root@k8s-master ~]#
[root@k8s-master ~]#
[root@k8s-master ~]#

```

## ADDING NODES TO THE CLUSTER

- From the node1 use the **kubeadm join** command to add the nodes into the cluster.

```

kubeadm join --token 878914.4587d610b5478141 192.168.3.81:6443 --discovery-token-ca-
cert-hash
sha256:6a23a6a7be997625f53e564a8b510627d035b000f9c288f7487fc9415c3338f1

```

*Note: Token ID,IP and sha256 details will vary based on your environment*

```

root@k8s-node1:~
[root@k8s-node1 ~]# kubeadm join --token 878914.4587d610b5478141 192.168.3.81:6443 --discovery-token-ca-cert-
b000f9c288f7487fc9415c3338f1

```

- Node has been joined into the cluster successfully.

```

root@k8s-node1:~
[root@k8s-node1 ~]# kubeadm join --token 878914.4587d610b5478141 192.168.3.81:6443 --discovery-token-ca-cert-
b000f9c288f7487fc9415c3338f1
[preflight] Running pre-flight checks.
[WARNING FileExisting-crictl]: crictl not found in system path
[discovery] Trying to connect to API Server "192.168.3.81:6443"
[discovery] Created cluster-info discovery client, requesting info from "https://192.168.3.81:6443"
[discovery] Requesting info from "https://192.168.3.81:6443" again to validate TLS against the pinned public
[discovery] Cluster info signature and contents are valid and TLS certificate validates against pinned roots.
[discovery] Successfully established connection with API Server "192.168.3.81:6443"

This node has joined the cluster:
* Certificate signing request was sent to master and a response
  was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the master to see this node join the cluster.
[root@k8s-node1 ~]#

```

## VERIFICATION

- From the master server, type the below command to list the available nodes in the cluster.

```

kubectl get nodes

```

```
root@k8s-master:~  
[root@k8s-master ~]# kubectl get nodes  
NAME           STATUS    ROLES    AGE   VERSION  
k8s-master     Ready     master   1h    v1.9.3  
k8s-node1      Ready     <none>   6m    v1.9.3  
[root@k8s-master ~]#
```

- We can use the same kubeadm join command to add more nodes in future.