How to install Kubernetes on CentOS 7

Step 1. Install Docker on all CentOS 7 VMs

Update the package database

sudo yum check-update

Install the dependencies

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

Add and enable official Docker Repository to CentOS 7

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

Install the latest Docker version on CentOS 7

sudo yum install docker-ce

A successful installation output will be concluded with a Complete!

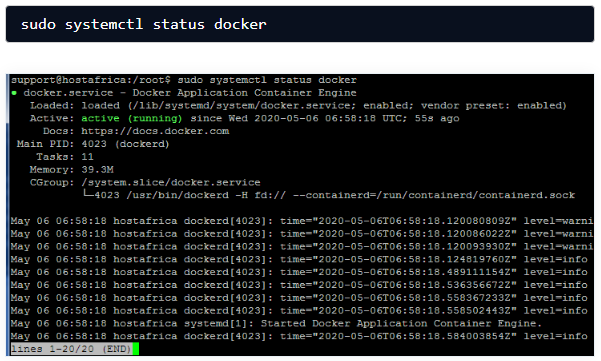
Manage Docker Service

Now Docker is installed, but the service is not yet running. Start and enable Docker using the commands

sudo systemctl start docker

sudo systemctl enable docker

To confirm that Docker is active and running use



Step 2. Set up the Kubernetes Repository

Since the Kubernetes packages aren’t present in the official CentOS 7 repositories, we will need to add a new repository file. Use the following command to create the file and open it for editing:

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

Once the file is open, press I key to enter insert mode, and paste the following contents:

[kubernetes]

name=Kubernetes

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

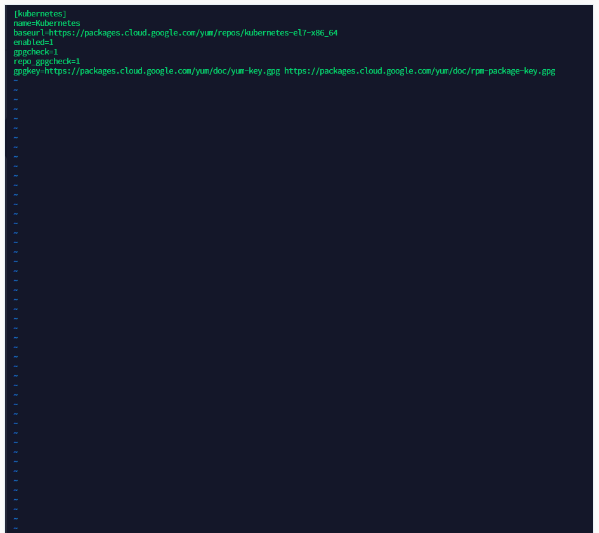
enabled=1

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

Once pasted, press escape to exit insert mode. Then enter :x to save the file and exit.

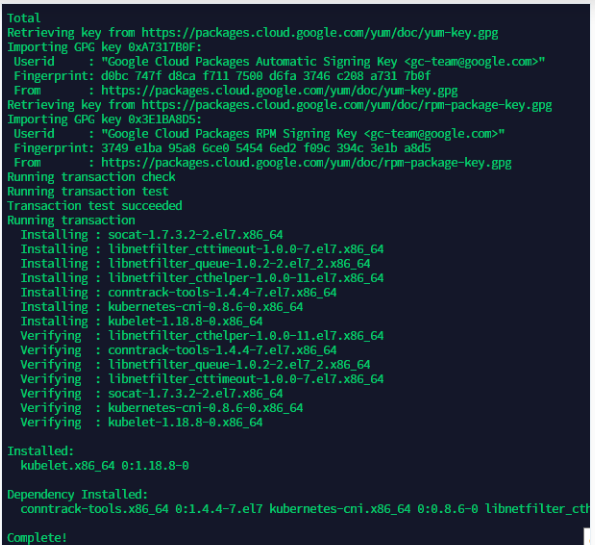


Step 3. Install Kubelet on CentOS 7

The first core module that we need to install on every node is Kubelet. Use the following command to do so:

sudo yum install -y kubelet

Once you enter the command, you should see a lot of logs being printed. A successful installation will be indicated by the Complete! keyword at the end. See below:



Step 4. Install kubeadm and kubectl on CentOS 7

kubeadm, the next core module, will also have to be installed on every machine. Use the following command:

sudo yum install -y kubeadm

Successful installation should result in the following output:  
(Note that kubeadm automatically installs kubectl as a dependency)



Step 5. Set hostnames

On your Master node, update your hostname using the following command:

sudo hostnamectl set-hostname master-node

sudo exec bash

And

sudo hostnamectl set-hostname W-node1

sudo exec bash

Now open the */etc/hosts* file and edit the hostnames for your worker nodes:

sudo cat <<EOF>> /etc/hosts

10.168.10.207 master-node

10.168.10.208 node1 W-node1

10.168.10.209 node2 W-node2

EOF

Step 6. Disable SElinux

To allow containers to be able to access the file system, we need to enable the “permissive” mode of SElinux. Use the following commands:  
(Note: For these commands to take effect, you will have to reboot)

sudo setenforce 0

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

reboot

### Step 9. Disable **swap**

For Kubelet to work, we also need to disable **swap** on all of our VMs:

sudo sed -i '/swap/d' /etc/fstab

sudo swapoff -a

This concludes our installation and configuration of Kubernetes on CentOS 7. We will now share the steps for deploying a k8s cluster.

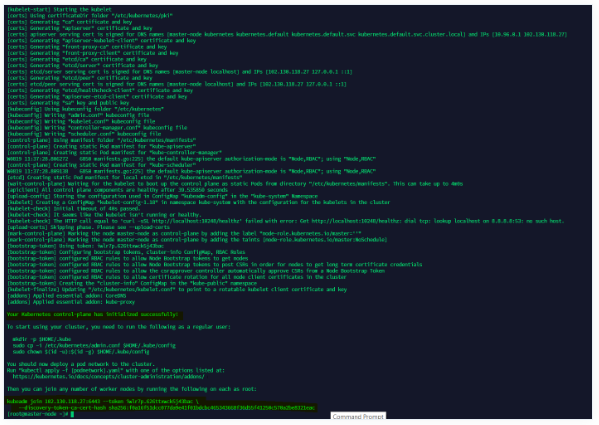
Deploying a Kubernetes Cluster on CentOS 7

Step 1. kubeadm initialization

To launch a new Kubernetes cluster instance, you need to initialize kubeadm. Use the following command:

sudo kubeadm init

This command may take several minutes to execute. Upon success, you should get logs similar to those in this screenshot:



You will also get an auto-generated command at the end of the output. Copy the text following the line Then you can join any number of worker nodes by running the following on each as root: as highlighted in the above screenshot and save it somewhere safe. We will use this to add worker nodes to our cluster.

Note: If you forgot to copy the command, or have misplaced it, don’t worry. You can retrieve it again by entering the following command:

sudo kubeadm token create --print-join-command

Step 2. Create required directories and start managing Kubernetes cluster

In order to start managing your cluster, you need to create a directory and assume ownership. Run the following commands as a regular user:

mkdir -p $HOME/.kube

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

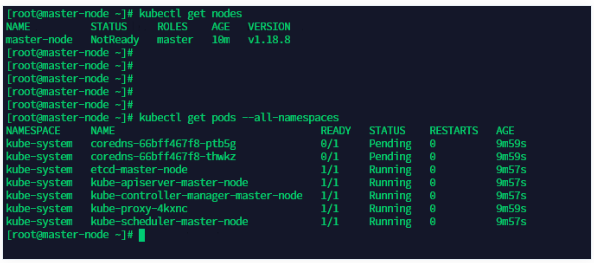
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Step 3. Set up Pod network for the Cluster

Pods within a cluster are connected via the pod network. At this point, it’s not working. This can be verified by entering the following two commands:

sudo kubectl get nodes

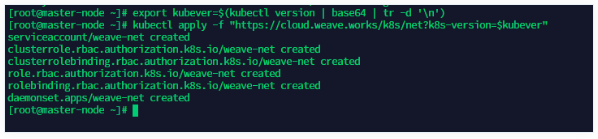
sudo kubectl get pods --all-namespaces

  
As you can see, the status of master–node is NotReady. The CoreDNS service is also not running. To fix this, run the following commands:

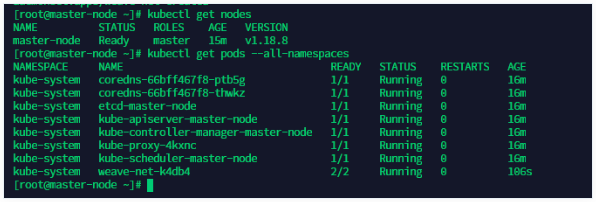
sudo export kubever=$(kubectl version | base64 | tr -d '\n')

sudo kubectl apply -f https://cloud.weave.works/k8s/net?k8s-version=$kubever

You should get the following output:



And now if you verify the statuses of your node and CoreDNS service, you should get Ready and Running like seen below:

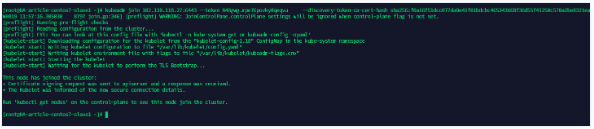


Step 4. Add nodes to your cluster

As a final step, you need to add worker nodes to your cluster. We will use the kubeadm join auto-generated token in [Step 1.](https://www.hostafrica.com/blog/new-technologies/install-kubernetes-delpoy-cluster-centos-7/#auto-gen) here. Run your own version of the following command on all of the worker node VMs:

sudo kubeadm join 102.130.118.27:6443 --token 848gwg.mpe76povky8qeqvu --discovery-token-ca-cert-hash sha256:f0a16f51dcc077da9e41f01bdcbc465343668f36d55f41250c570a2be8321eac

On successful addition, you should get the following output:



Running the following command on the master-node should show your newly added node.

sudo kubectl get nodes

