***Setting up Kubernetes Cluster [1 Master 2 Node]***

1. Create a Virtual Machine with CentOS guest OS
2. We need to apply some configuration changes on the OS

**Set enforce to true**

setenforce 0

**Disable selinux**

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

**Make changes in the network config**

modprobe br\_netfilter

echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables

swapoff –a

**Edit this file**

**Comment out -> /dev/mapper/centos-swap swap**

vi /etc/fstab

/dev/mapper/centos-swap swap -> #/dev/mapper/centos-swap swap

**Install dependencies for Docker**

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

**Add repo for docker**

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

**Install Docker**

yum install -y docker-ce

**Add repo for Kubernetes**

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

**Install Kubernetes Components [These are required to run and Manage** **Kubernetes Cluster]**

yum install -y kubelet kubeadm kubectl

**Reboot The System so that all changes are applied**

sudo reboot

systemctl start docker && systemctl enable docker

systemctl start kubelet && systemctl enable kubelet

**Check Docker group info and then change if requires**

docker info | grep -i cgroup

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

**Restart the daemon and Kubelet**

systemctl daemon-reload

systemctl restart kubelet

**DO a “docker ps” to see if docker is running**

**\*\*\*\*\*\*\*\*Now Clone the systems**

**One will be Master and the other will be nodes**

**Check Ip for all the systems**

**Edit the hosts file**

vi /etc/hosts

**add in above:**

192.168.1.23 Centos-k8s-Master

198.168.1.24 node01

198.168.1.25 node-02

**Then run**

sudo reboot

systemctl start docker && systemctl enable docker

systemctl start kubelet && systemctl enable kubelet

systemctl daemon-reload

systemctl restart kubelet

**Check Firewall on All Systems**

**Disable if enabled or configure to access each other**

systemctl disable firewalld

systemctl stop firewalld

**\*\*\*\*\*\*\*\*\*\*\*\*\*only on Master**

kubeadm init --apiserver-advertise-address=<master ip> --pod-network-cidr=10.244.0.0/16

**master ip -> Replace this with the Master IP**

**Copy the kubadm join from the OUTPUT**

**Then run this**

**Set Kube Config**

mkdir -p $HOME/.kube

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

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

**Run Flannel**

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

**For RBAC Flannel:**

After setting up the cluster; in browser 6443 is inaccessible; unauthorised; that is resolved after implementing this

kubectl create -f kubectl create -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.yml

kubectl delete -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.yml

kubectl get nodes

kubectl get pods --all-namespaces

\*\*\*\*\*\*\*\*\*only on SLAVE

kubeadm join 192.168.43.128:6443 --token xb73dl.omxx6vpwhtu0h4ld --discovery-token-ca-cert-hash sha256:a19beea411690362ea3a1500fe232aeb1555d54805af39f427529073acfd777a

**\*\*\*\*\*\*\*\*\*\*Testing The Cluster**

kubectl create deployment nginx --image=nginx

kubectl create deployment nginx --image=nginx

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

kubectl get pods

kubectl get svc

curl k8s-Slave01:31437

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*In Browser:**

kubectl get svc

nginx NodePort 10.107.194.115 <none> 80:31437/TCP

5m

**Enter any of the IPs and port number from the service and you can see the Container**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DashBoard**

https://<master-ip>:<apiserver-port>/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/

https://192.168.43.128:6443/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/

*\*\*\*\*\*\*\*\*\*Error*

[root@k8s-Master kubernetes]# kubectl cluster-info

Kubernetes master is running at https://192.168.43.128:6443

KubeDNS is running at https://192.168.43.128:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

***If you open any of those in browser; then we get an error saying:***

{

"kind": "Status",

"apiVersion": "v1",

"metadata": {

},

"status": "Failure",

"message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",

"reason": "Forbidden",

"details": {

},

"code": 403

}

*\*\*\*\*\*\*\*\*\*Certificate in Browser* ***[https://]***

go to /etc/kubernetes on Master

grep 'client-certificate-data' ~/.kube/config | head -n 1 | awk '{print $2}' | base64 -d >> kubecfg.crt

grep 'client-key-data' ~/.kube/config | head -n 1 | awk '{print $2}' | base64 -d >> kubecfg.key

openssl pkcs12 -export -clcerts -inkey kubecfg.key -in kubecfg.crt -out kubecfg.p12 -name "kubernetes-client"

copy the .p12 file to the system in whose browser you are trying to access

Open Chrome

Settings-Advanced Settings-Import Certificate

Close and Open Browser

You should be able to access now

***To setup DashBoard:***

[*http://www.joseluisgomez.com/containers/kubernetes-dashboard/*](http://www.joseluisgomez.com/containers/kubernetes-dashboard/)

kubectl create -f https://raw.githubusercontent.com/kubernetes/dashboard/master/src/deploy/recommended/kubernetes-dashboard.yaml

***Create Service account***

apiVersion: v1

kind: ServiceAccount

metadata:

name: admin-user

namespace: kube-system

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

name: admin-user

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: cluster-admin

subjects:

- kind: ServiceAccount

name: admin-user

namespace: kube-system

***Get Token***

kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep admin-user | awk '{print $1}')

***Get KubeConfig***

Don’t Know

**Check Firewall on All Systems**