Kubernetes Kubeadm Install

Step 1:- Disable selinux and firewalld

#getenforce
#setenforce 0
#vim /etc/selinux/config
SELINUX=disabled

#swapoff -a
#vi /etc/fstab --> Comment the swap fs
#systemctl stop firewalld && systemctl disable firewalld
#reboot

Step 2:- Update the system and install docker

#yum update -y && yum groupinstall "Compatibility libraries" -y
#yum -y install docker && systemctl start docker && systemctl enable docker

Step3:- Enable kubernetes repo and install kubernetes.

#cat <<'EOF' > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-$basearch
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

#cat /etc/yum.repos.d/kubernetes.repo
#yum -y install kubeadm kubelet kubectl && systemctl enable kubelet

Step 4:- Run below command on master node:-

#kubeadm init --pod-network-cidr=192.168.0.0/16

```
Your Kubernetes control-plane 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/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 10.128.15.221:6443 --token i0hraa.4odc9pejywz23uic \
    --discovery-token-ca-cert-hash sha256:646a8f422f187911af11cd6e7242076ea685c34d36ae16621443b64da44ff095
```

Note:- Copy token somewhere and use it while adding worker node.

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

#kubectl apply -f https://docs.projectcalico.org/v3.10/manifests/calico.yaml

```
[root@kubemaster ~]# kubectl apply -f https://docs.projectcalico.org/v3.10/manifests/calico.yaml
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
serviceaccount/calico-node created
deployment.apps/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
```

Step 5: Add above key token on all worker nodes. (below is example)

#kubeadm join 10.128.15.221:6443 --token uax4n0.c1hczbb7nc2th90s --discovery-token-ca-cert-hash sha256:0c6fb4174dc2b8ebe95c48331b3dfe2c3601ab638f309434d0e762d19f13ed62

```
[root@kubeworkernode1 ~]# kubeadm join 10.128.15.221:6443 --token i0hraa.4odc9pejywz23
uic --discovery-token-ca-cert-hash sha256:646a8f422f187911af11cd6e7242076ea685c34d36ae
16621443b64da44ff095
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm
kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.1
6" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/
kubeadm-flags.env"
[kubelet-start] Activating the kubelet service
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

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

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

[root@kubeworkernode1 ~]#
```

On master node:-

#kubectl get nodes -o wide

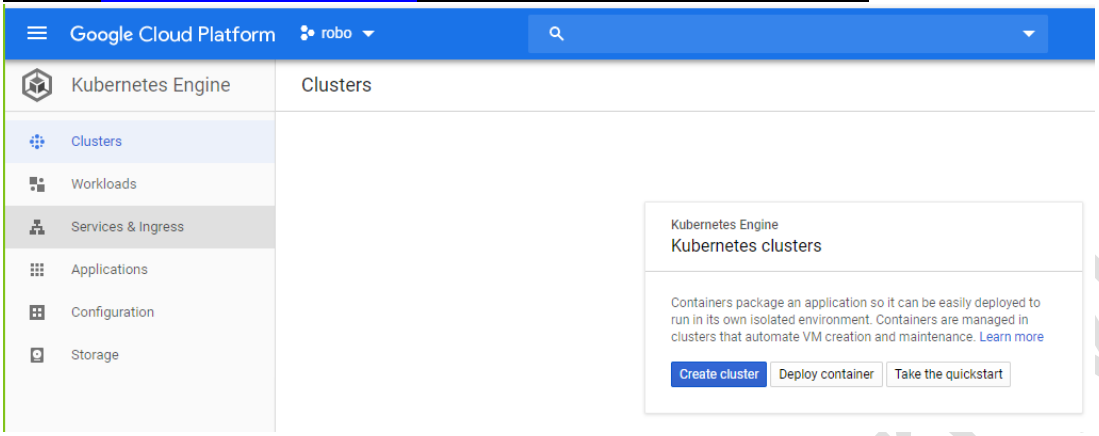
#kubectl get nodes

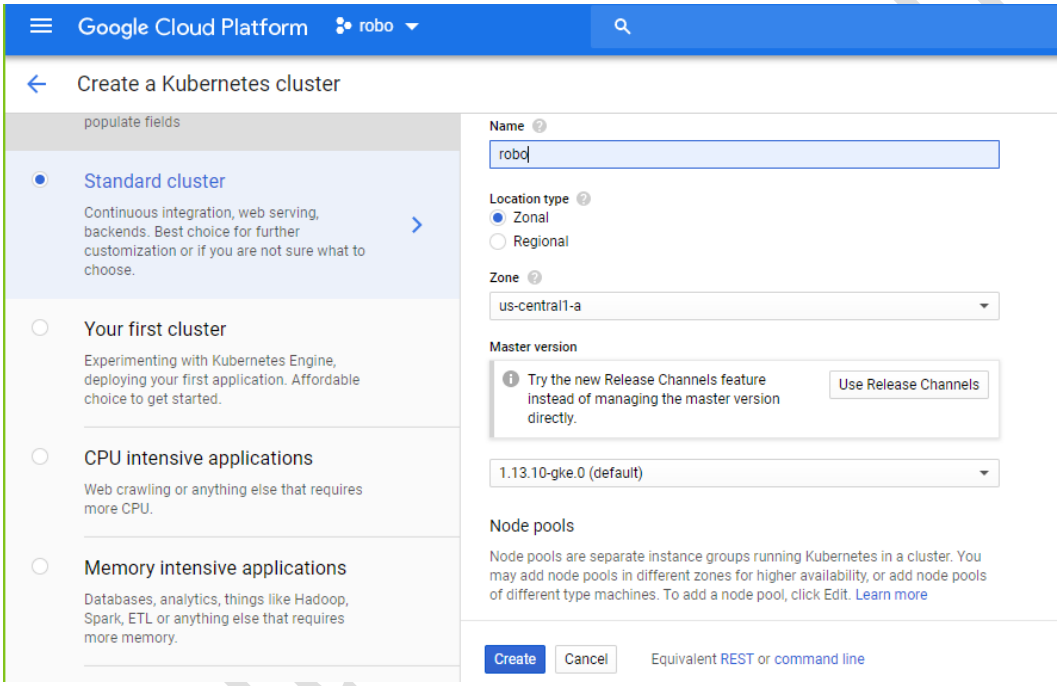
```
[root@kubemaster ~]# kubectl get nodes
NAME              STATUS   ROLES    AGE     VERSION
kubemaster        Ready    master   16m     v1.16.2
kubeworkernode1   Ready    <none>   8m29s   v1.16.2
kubeworkernode2   Ready    <none>   7m57s   v1.16.2
[root@kubemaster ~]#
```

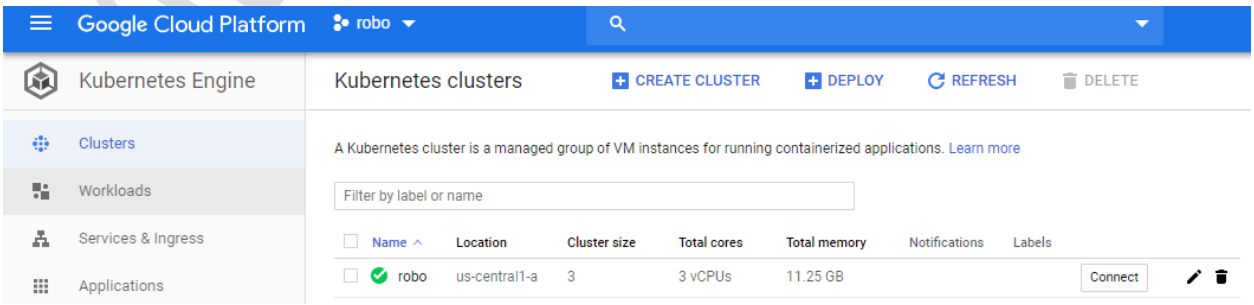
Note:- I have taken machines from google cloud platform and done the Kubernetes cluster setup.

Kubernetes Engine Cluster On GCP

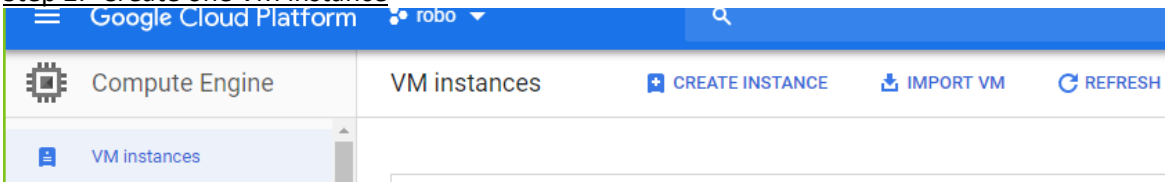
Step 1:- https://cloud.google.com/ and create Kubernetes engine cluster

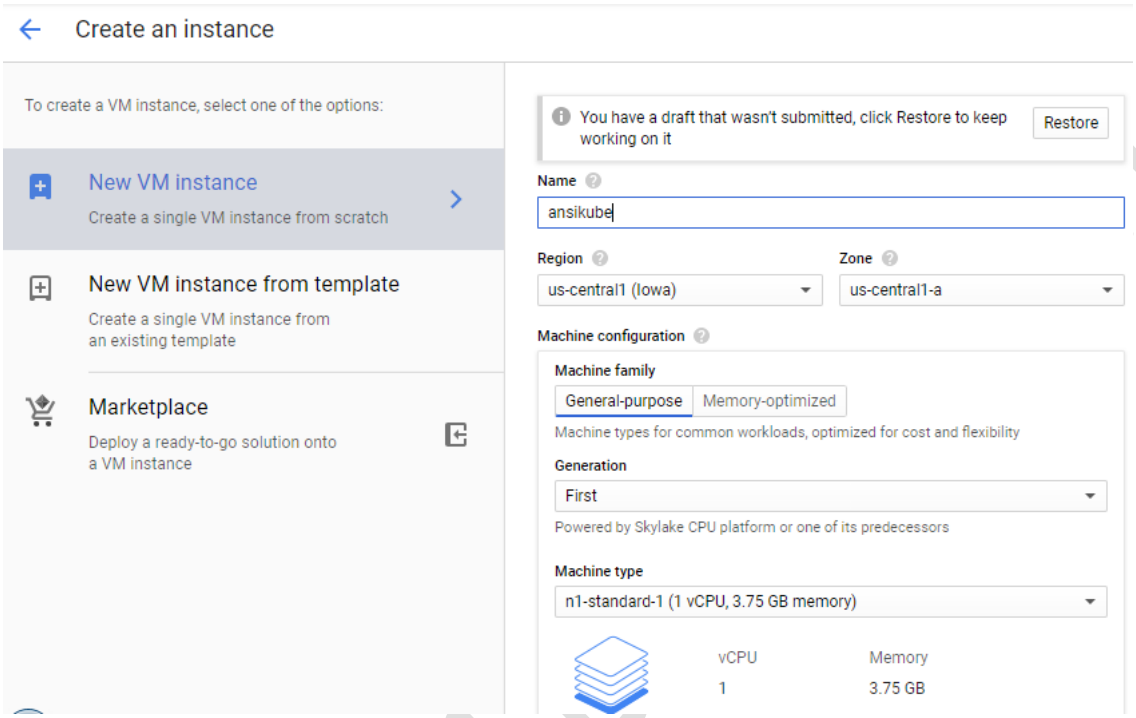


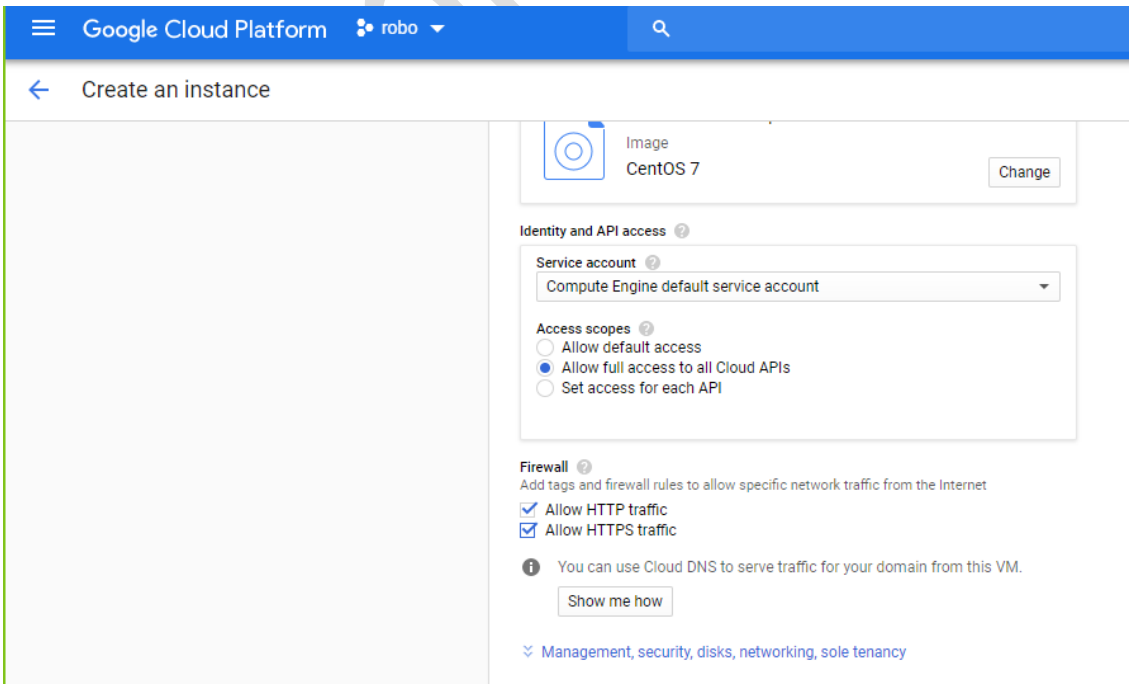




Step 2:- Create one VM instance





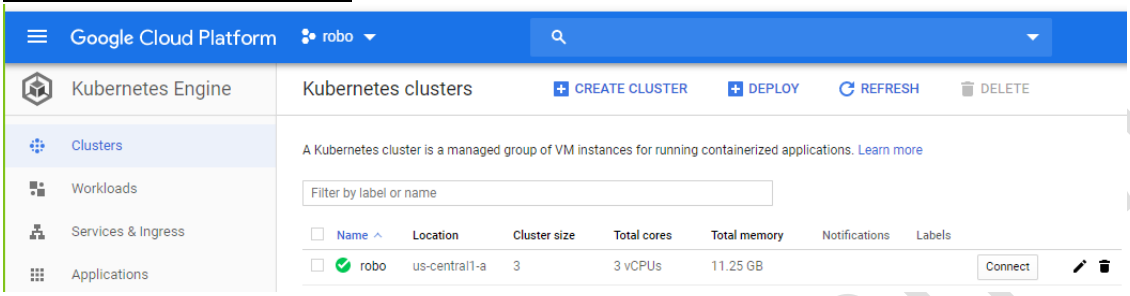


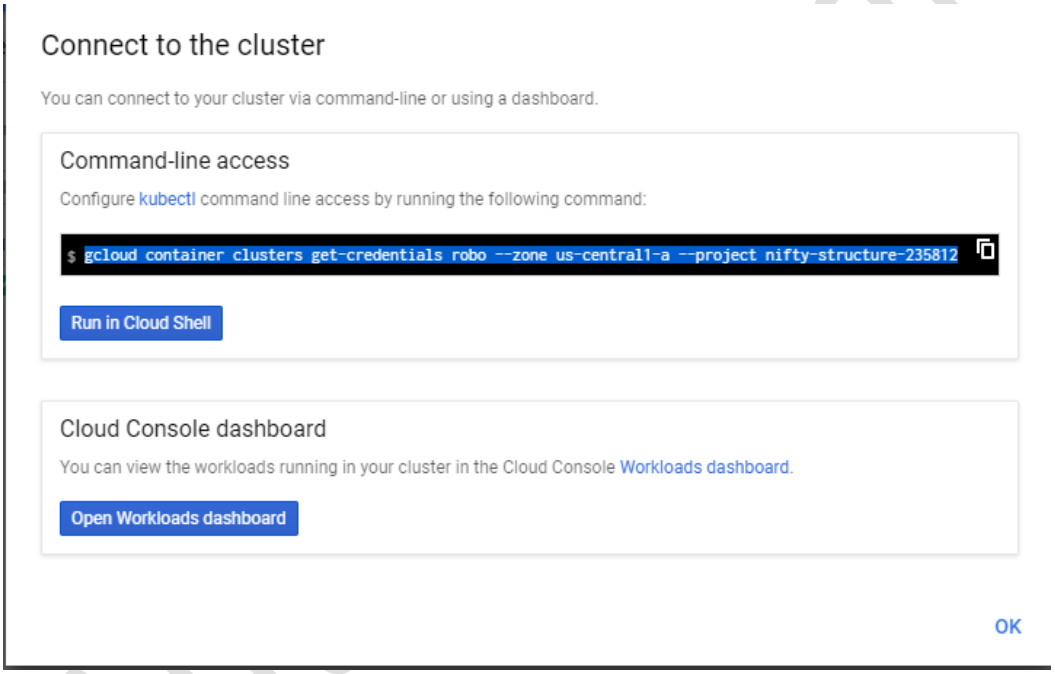
@@Take External ip and login to the server via putty with ssh key.

yum update -y && yum groupinstall "Compatibility libraries" -y && yum install kubectl -y

```
[root@ansikube ~]# kubectl get nodes
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[root@ansikube ~]#
```

@@open the Kubernetes engine cluster and click connect → select command-line access → copy and run in on standlone machine.






```
[root@ansikube ~]# gcloud container clusters get-credentials robo --zone us-central1-a --project nifty-structure-235812
Fetching cluster endpoint and auth data.
kubeconfig entry generated for robo.
[root@ansikube ~]#
```

#kubectl get nodes

```
[root@ansikube ~]# kubectl get nodes
NAME                                   STATUS   ROLES    AGE   VERSION
gke-robo-default-pool-9930fdca-1nbr    Ready    <none>   16m   v1.13.10-gke.0
gke-robo-default-pool-9930fdca-3q4m    Ready    <none>   16m   v1.13.10-gke.0
gke-robo-default-pool-9930fdca-zg7p    Ready    <none>   16m   v1.13.10-gke.0
[root@ansikube ~]#
```