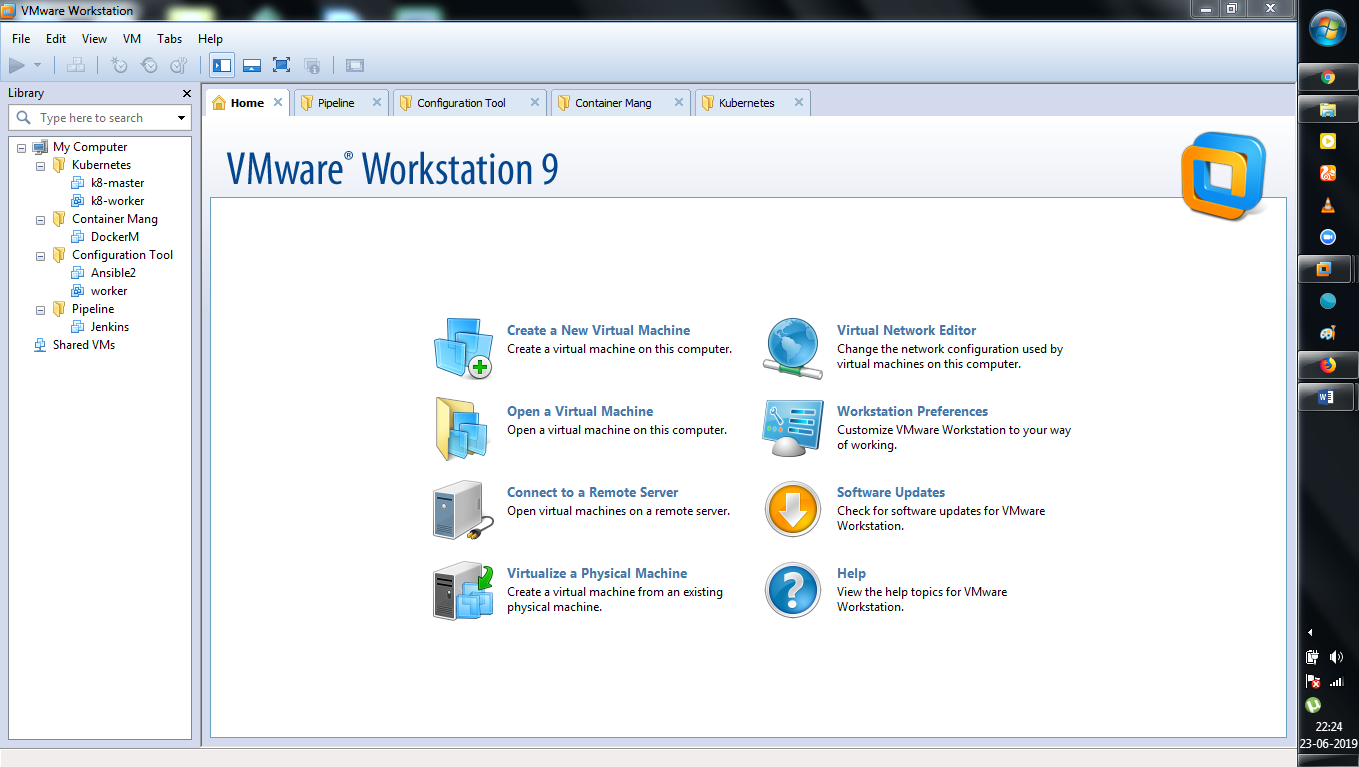
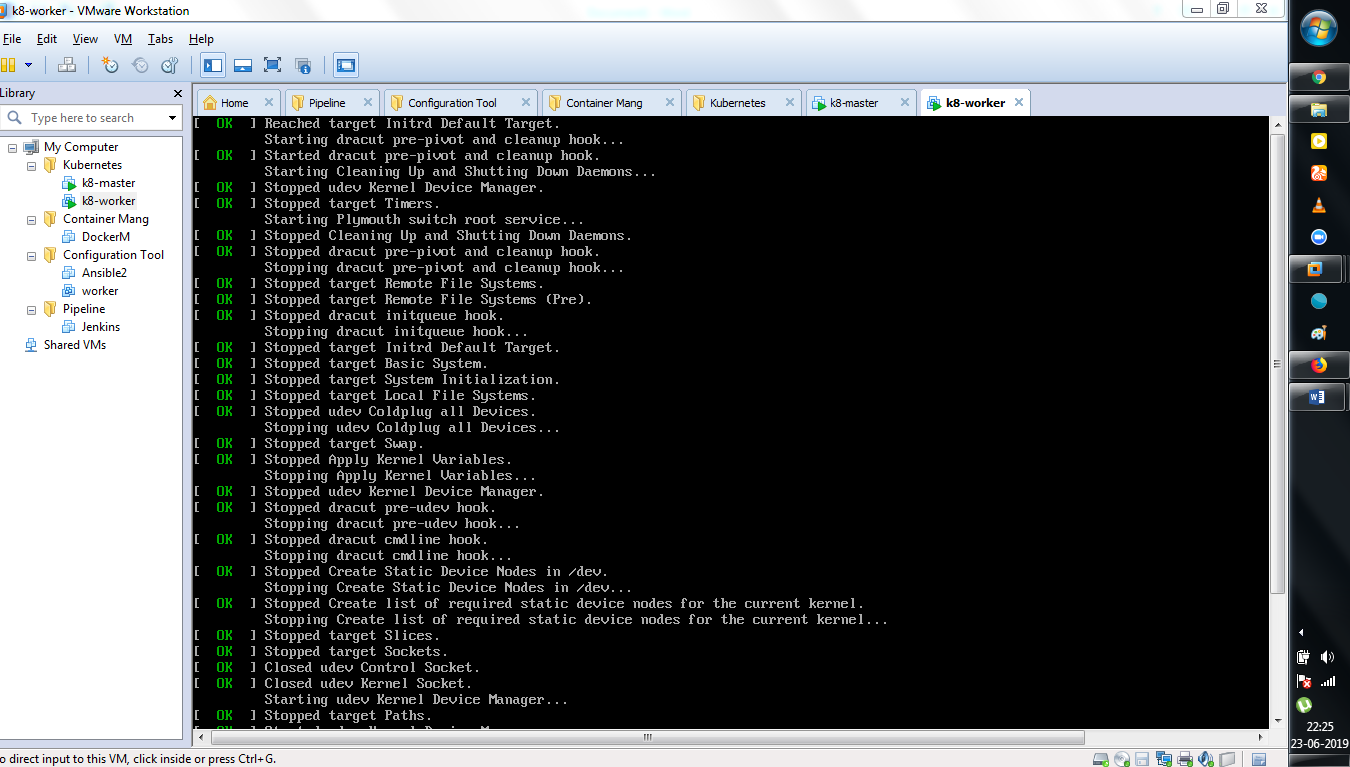
Take home test for DevOps/SysOps role – Level#1

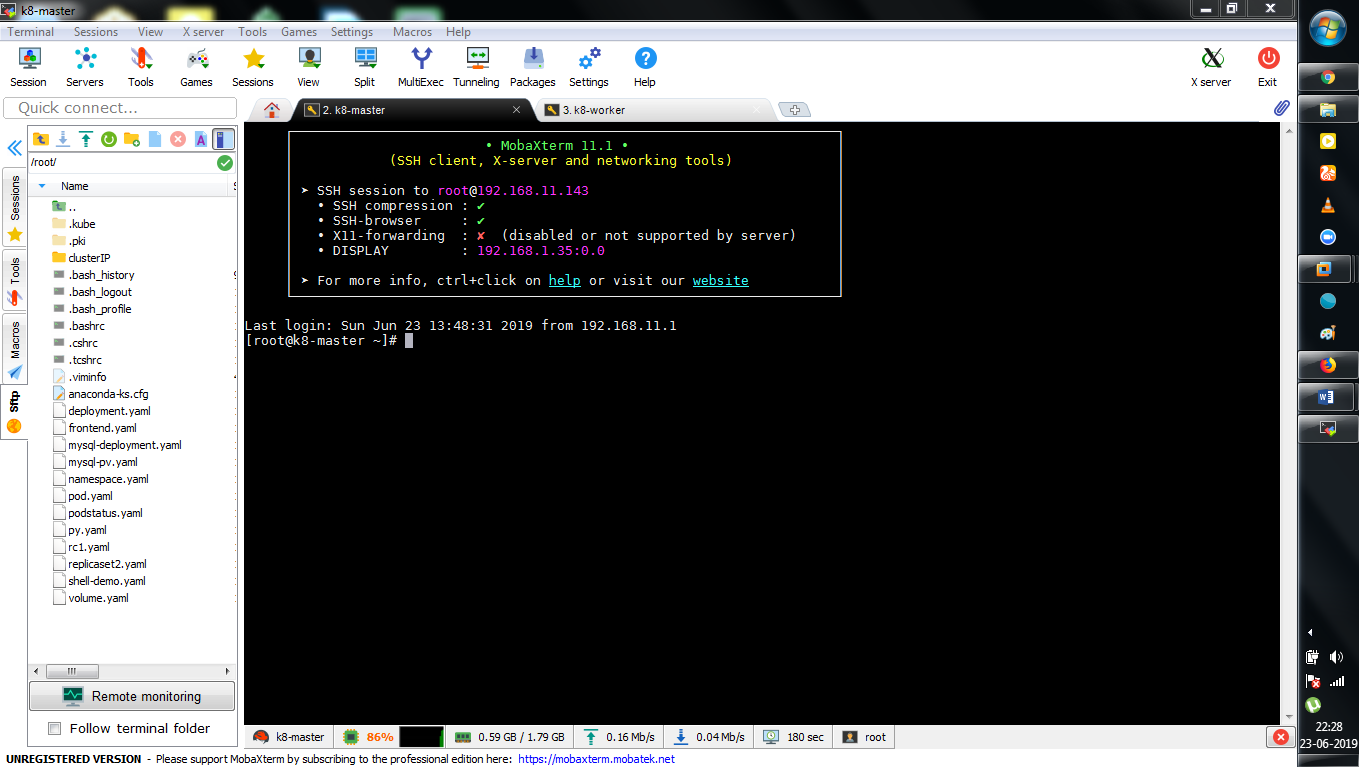
Tasks:

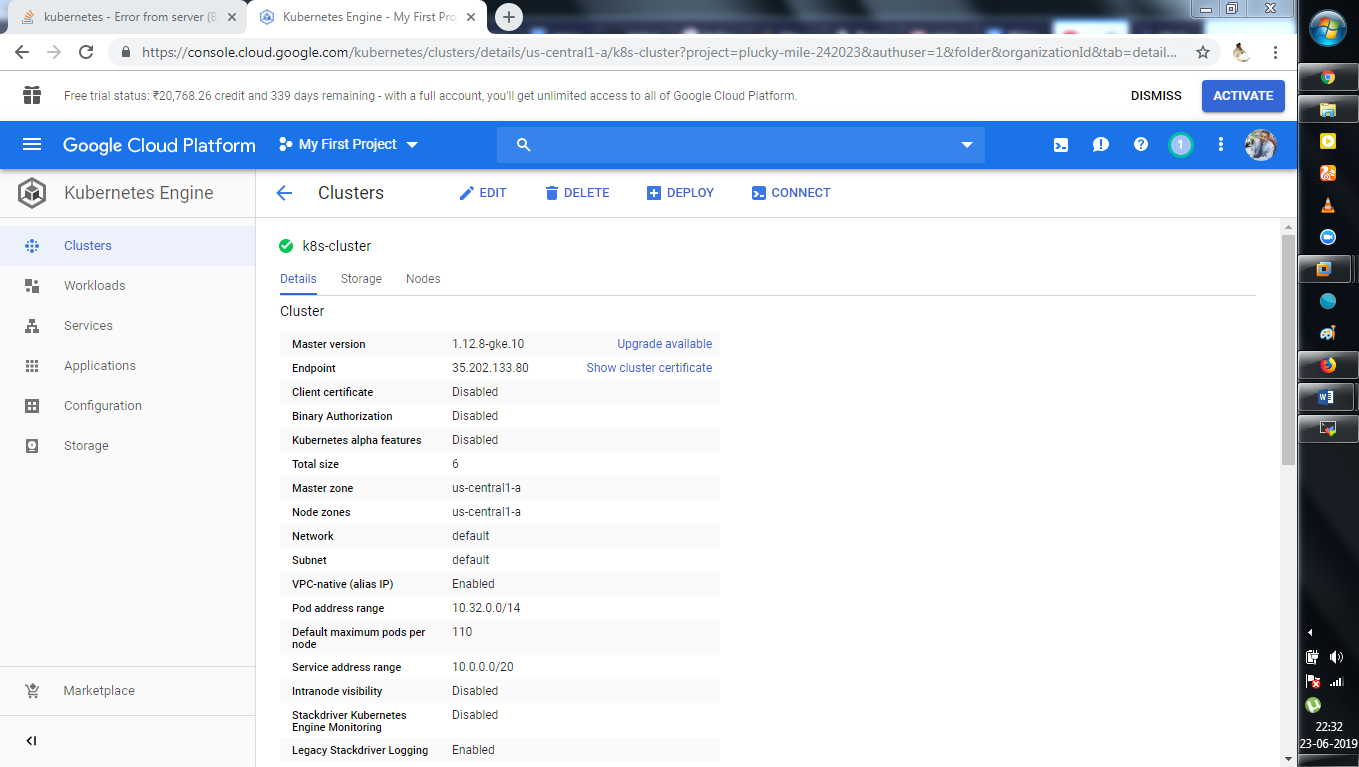
1. **Create a Kubernetes cluster on GCP (GCP gives free credits on signup so those should suffice for this exercise). If possible share a script / code which can be used to create the cluster.**

Build the Kubernetes Cluster setup in VMWare Workstation

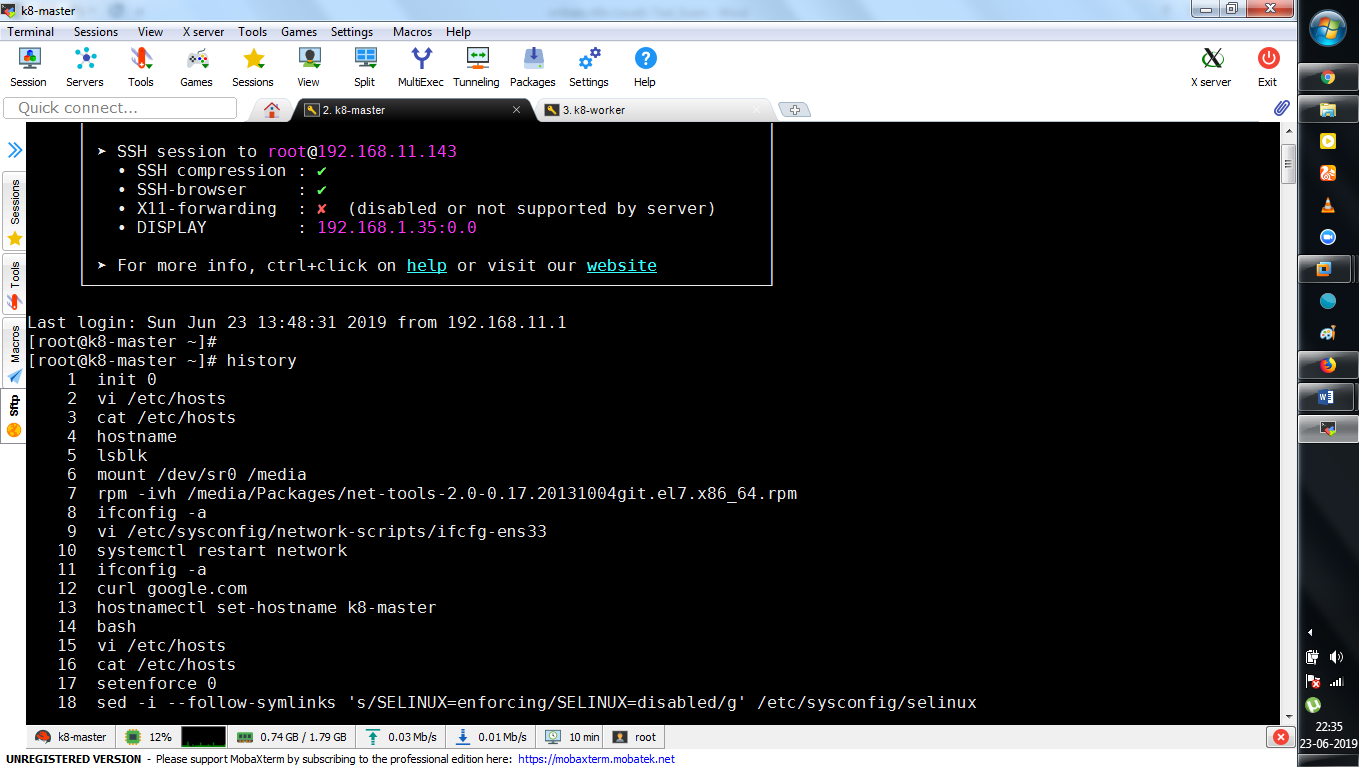








History : VMware machine



[root@k8-master ~]# history

1 init 0

2 vi /etc/hosts

3 cat /etc/hosts

4 hostname

5 lsblk

6 mount /dev/sr0 /media

7 rpm -ivh /media/Packages/net-tools-2.0-0.17.20131004git.el7.x86\_64.rpm

8 ifconfig -a

9 vi /etc/sysconfig/network-scripts/ifcfg-ens33

10 systemctl restart network

11 ifconfig -a

12 curl google.com

13 hostnamectl set-hostname k8-master

14 bash

15 vi /etc/hosts

16 cat /etc/hosts

17 setenforce 0

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

19 modprobe br\_netfilter

20 ls -l /proc/sys/net/bridg

21 ls -l /proc/sys/net/

22 ls -l /proc/sys/net/bridge/

23 swapoff -a

24 vi /etc/fstab

25 mount -a

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

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

28 yum install -y docker-ce

29 cat <<EOF > /etc/yum.repos.d/kubernetes.repo

30 [kubernetes]

31 name=Kubernetes

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

33 enabled=1

34 gpgcheck=1

35 repo\_gpgcheck=1

36 gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg

37 https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg

38 EOF

39 yum install -y kubelet kubeadm kubectl

40 init 6

41 ifconfig -a

42 systemctl start docker && systemctl enable docker

43 systemctl start kubelet && systemctl enable kubelet

44 docker info | grep -i cgroup

45 systemctl status firewalld

46 systemctl stop firewalld

47 systemctl diable firewalld

48 systemctl disable firewalld

49 docker info | grep -i cgroup

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

51 systemctl daemon-reload

52 systemctl restart kubelet

53 systemctl status kubelet

54 systemctl start kubelet

55 systemctl status kubelet

56 kubectl get nodes

57 systemctl status kubelet

58 swapoff -a

59 partprobe

60 systemctl start kubelet

61 systemctl status kubelet

62 kubeadm reset

63 history

64 systemctl status kubelet

65 init 6

66 systemctl status kubelet

67 systemctl start kubelet

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

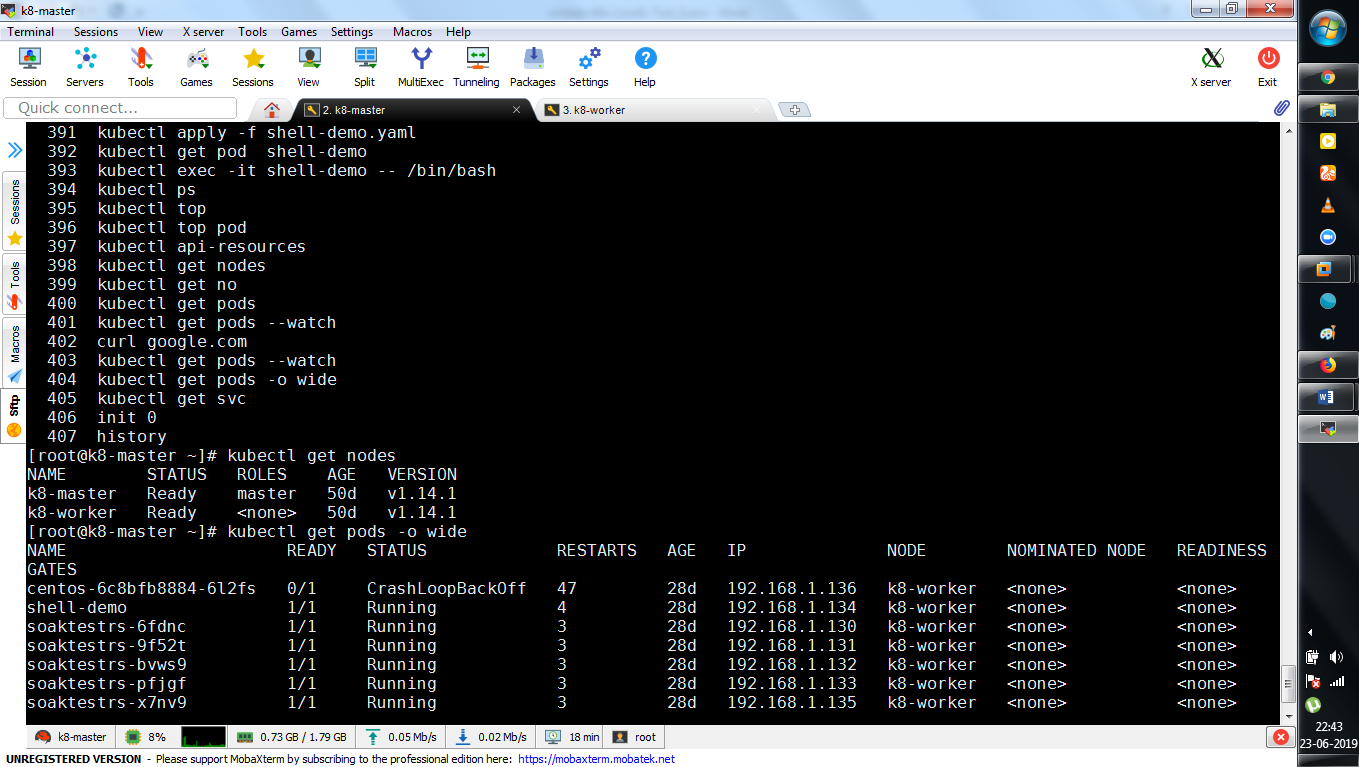
69 kubeadm init --apiserver-advertise-address=192.168.11.143 --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=FileContent--proc-sys-net-bridge-bridge-nf-call-iptables

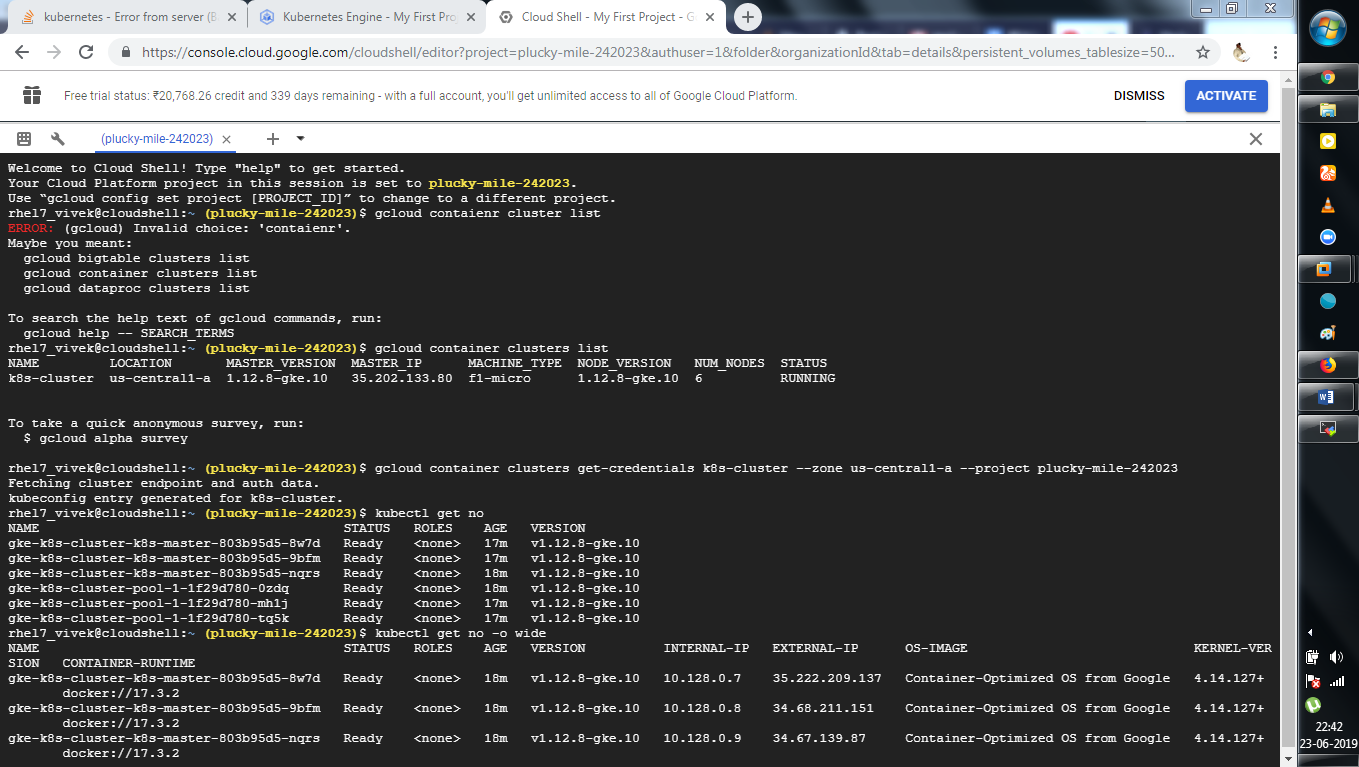
70 mkdir -p $HOME/.kube

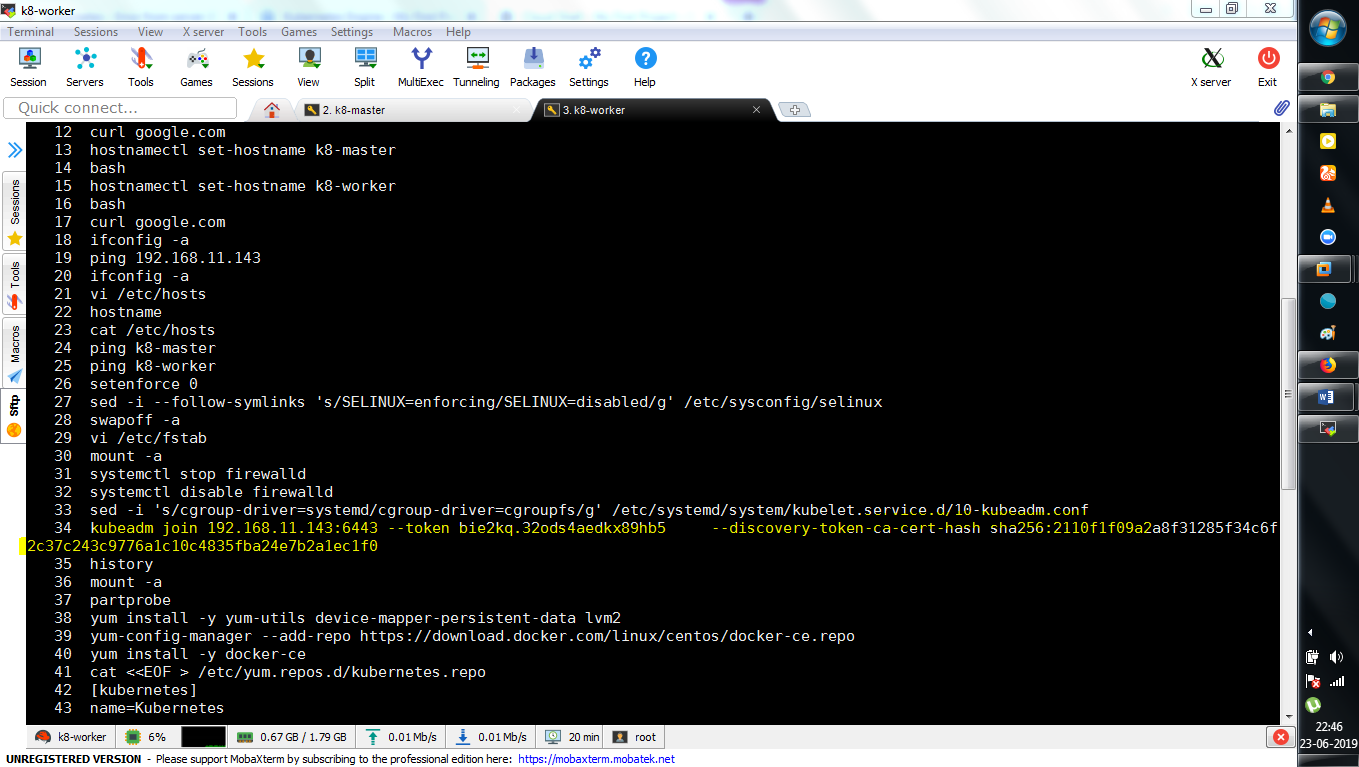
71 sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

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

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





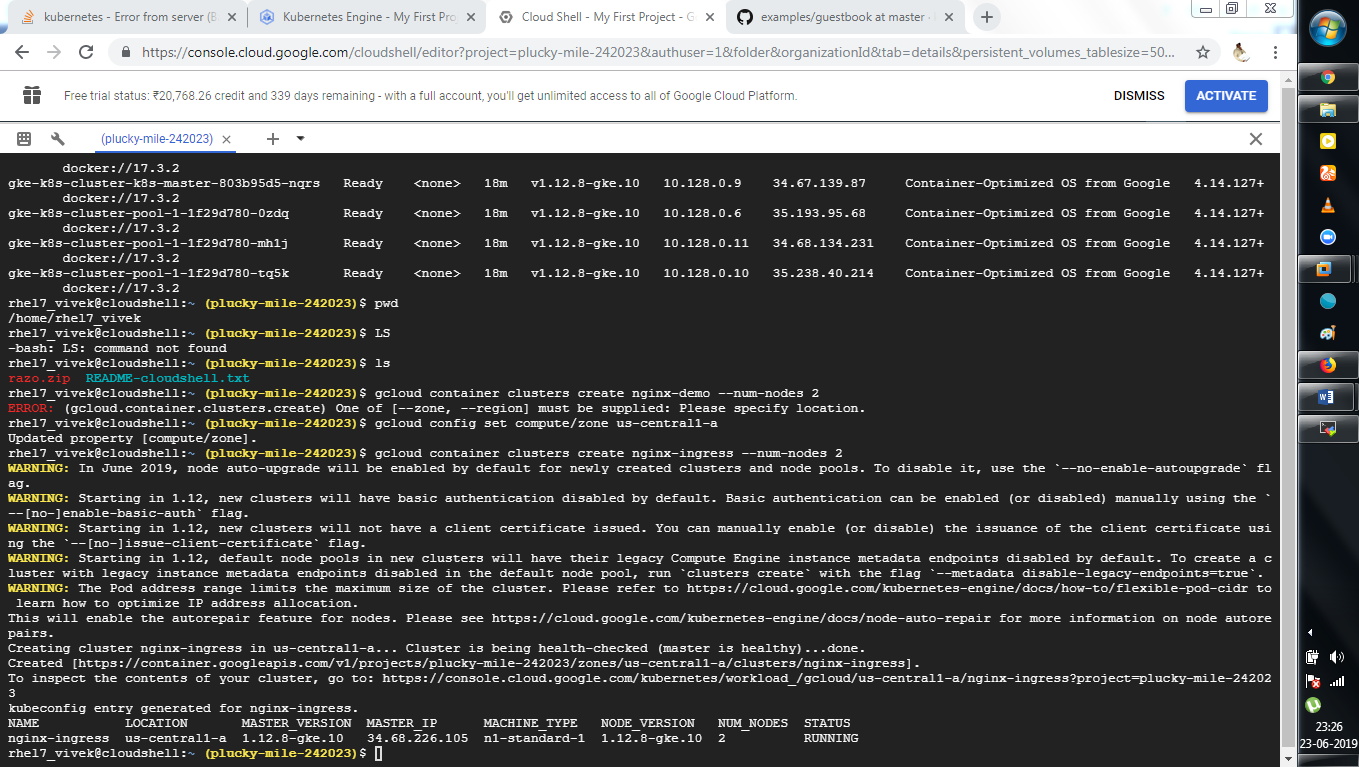


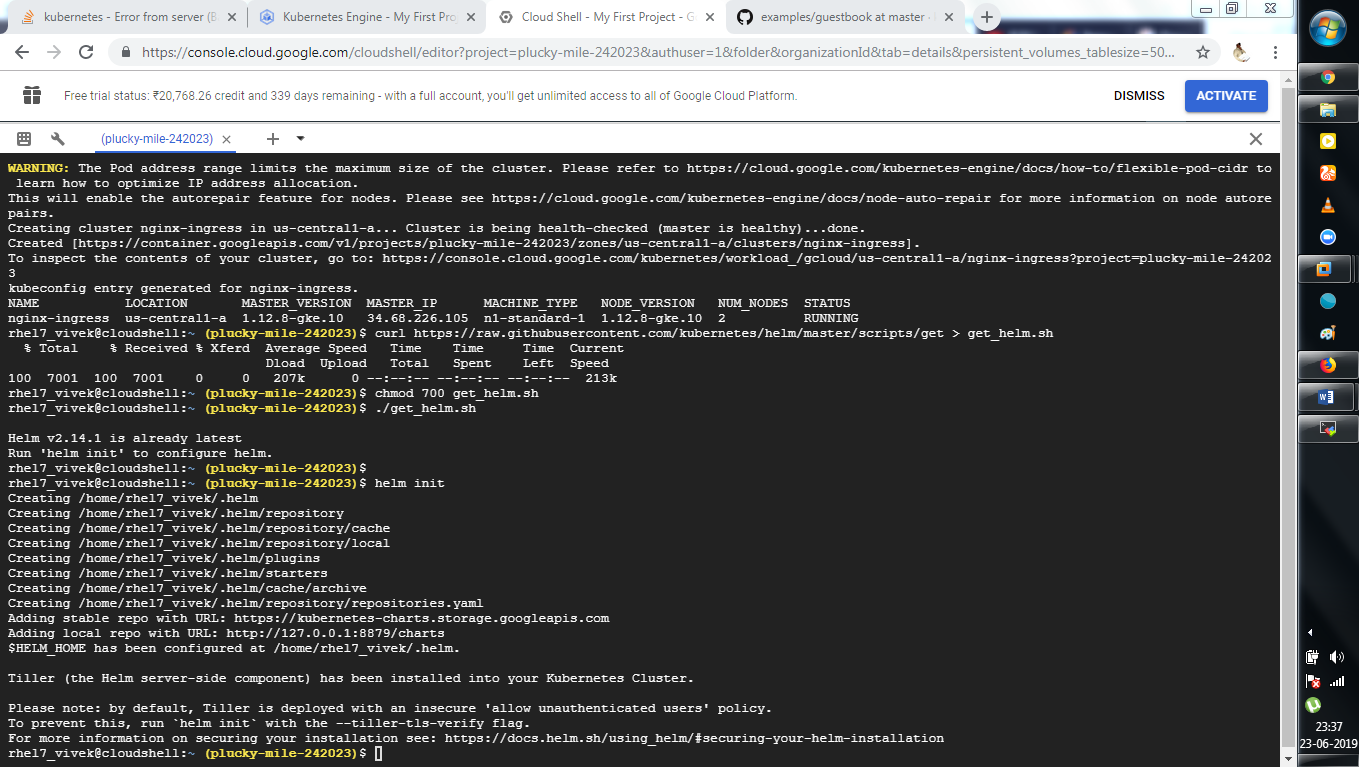
1. **Install nginx ingress controller on the cluster. For now, we consider that the user will add public IP of ingress LoadBalancer to their /etc/hosts file for all hostnames to be used. So do not worry about DNS resolution.**

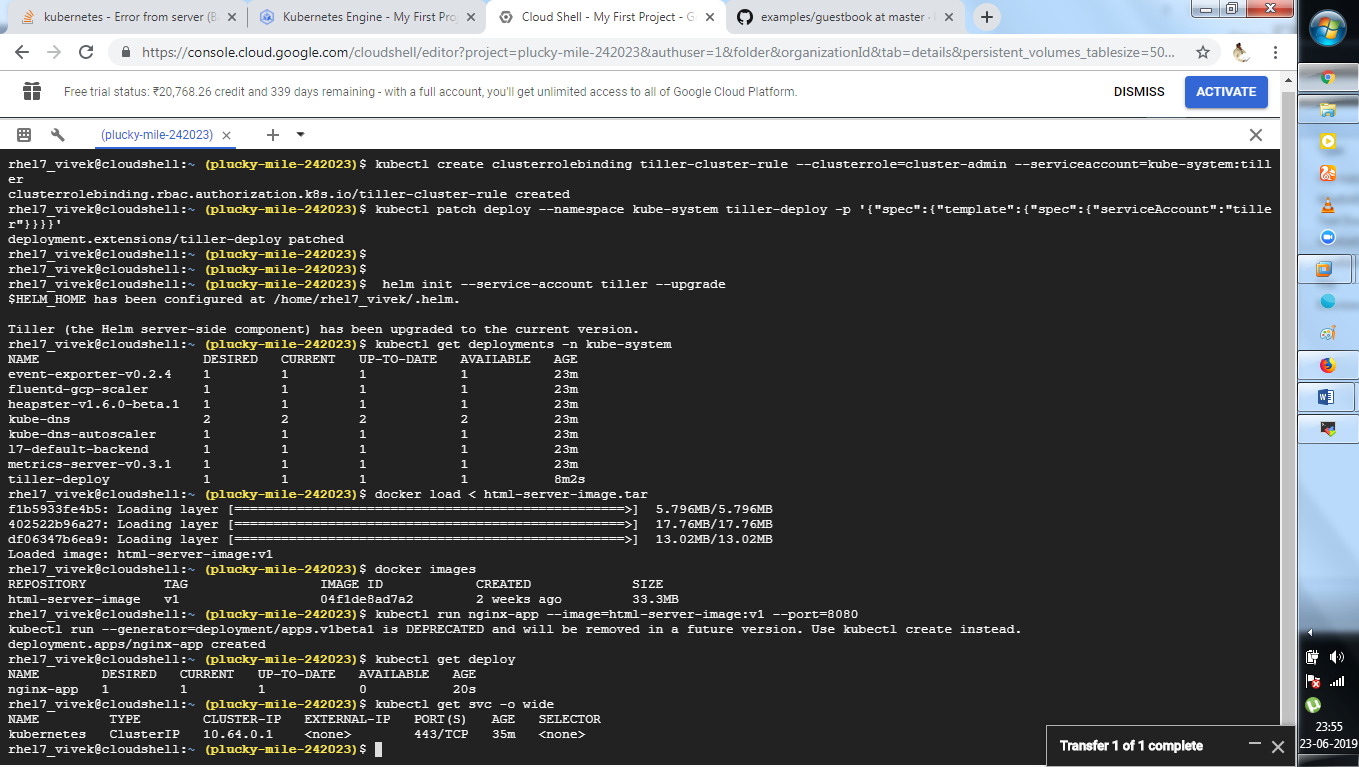
Ingress allows external users and outside client applications access to HTTP services. Ingress consists of two components: **Ingress resource** and **Ingress controller** and it is vital that both pieces are properly configured so that traffic can be routed from an outside client to a Kubernetes Service

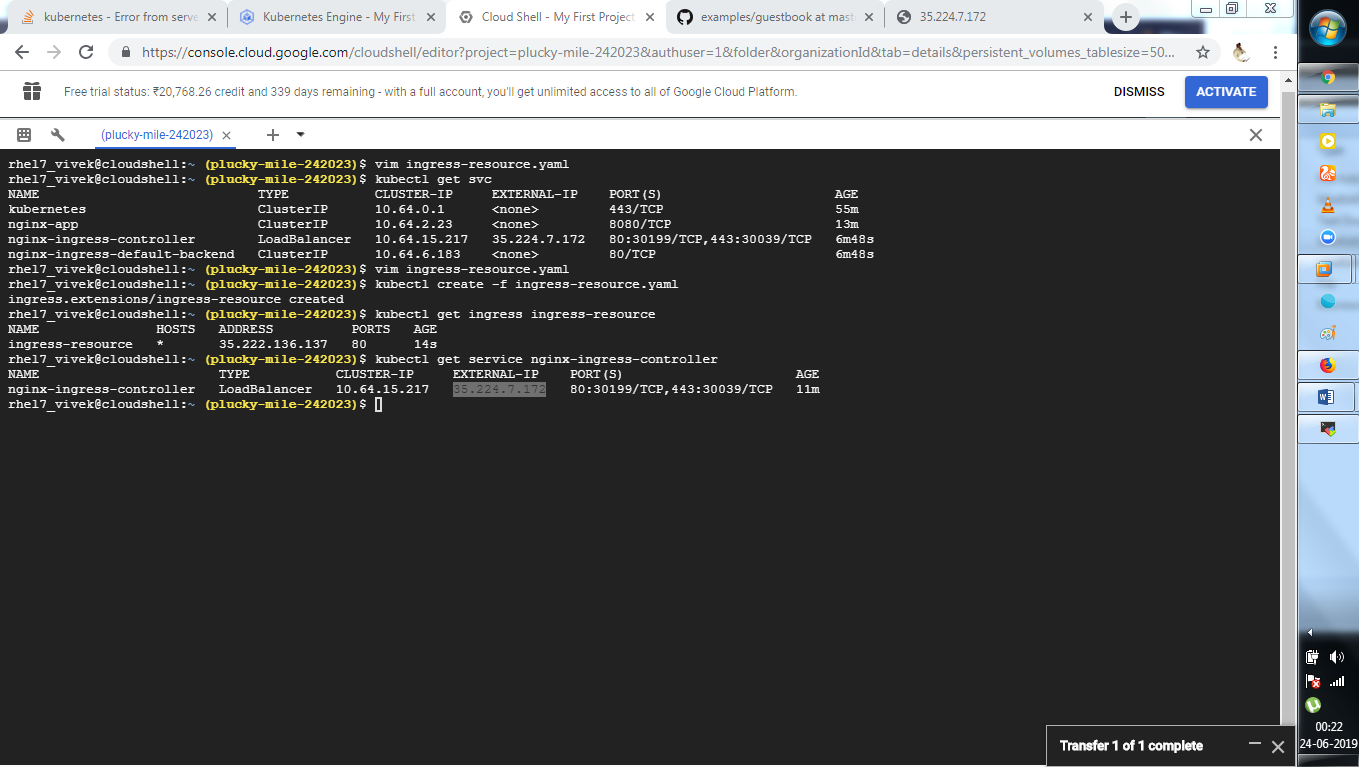
$ gcloud config set compute/zone us-central1-a

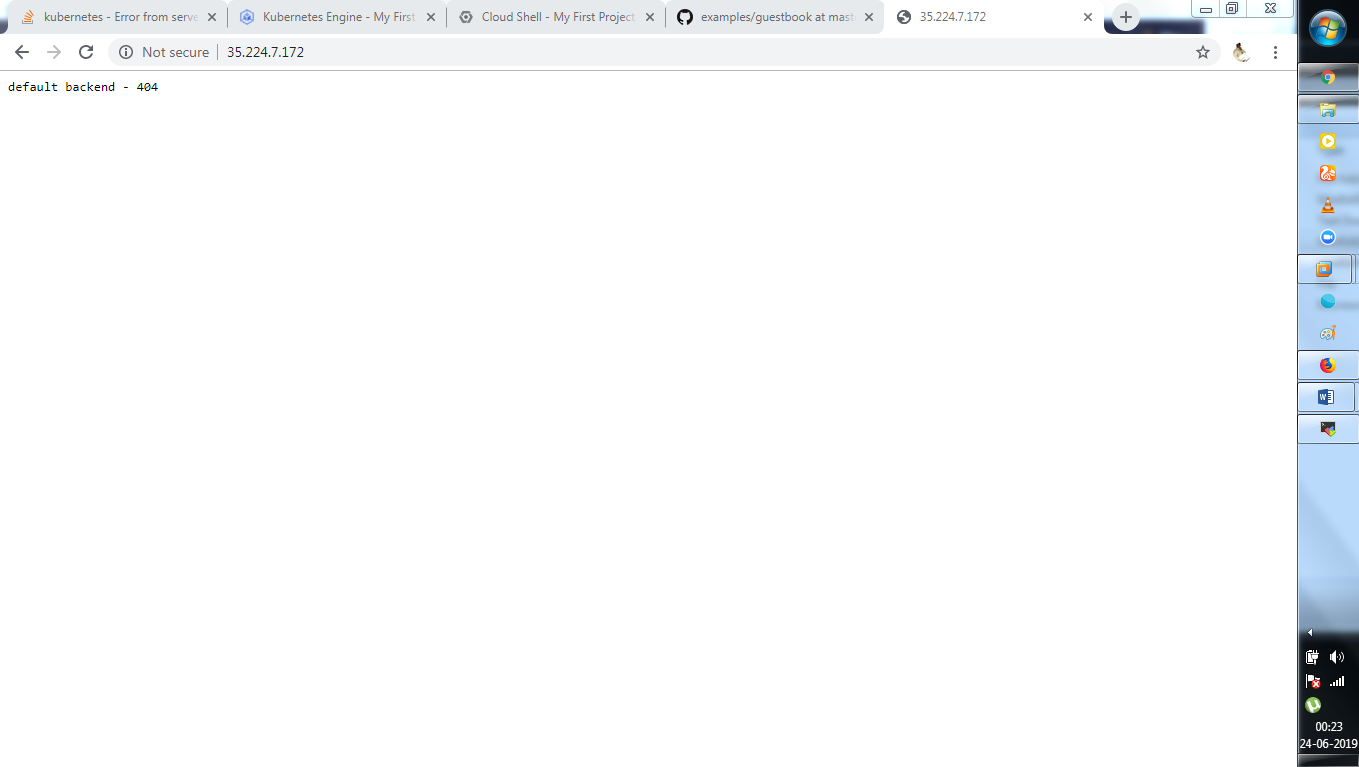
$ gcloud container clusters create nginx-demo --num-nodes 2











**rhel7\_vivek@cloudshell:~ (plucky-mile-242023)$ cat ingress-resource.yaml**

apiVersion: extensions/v1beta1

kind: Ingress

metadata:

name: ingress-resource

annotations:

kubernetes.io/ingress.class: nginx

nginx.ingress.kubernetes.io/ssl-redirect: "false"

spec:

rules:

- http:

paths:

- path: /usr/share/nginx/html

backend:

serviceName: nginx-app

servicePort: 8080

