
Vizle
edureka!
Ui
IOT
AI
Se
Vizle

Topics For Today's DevOps Training

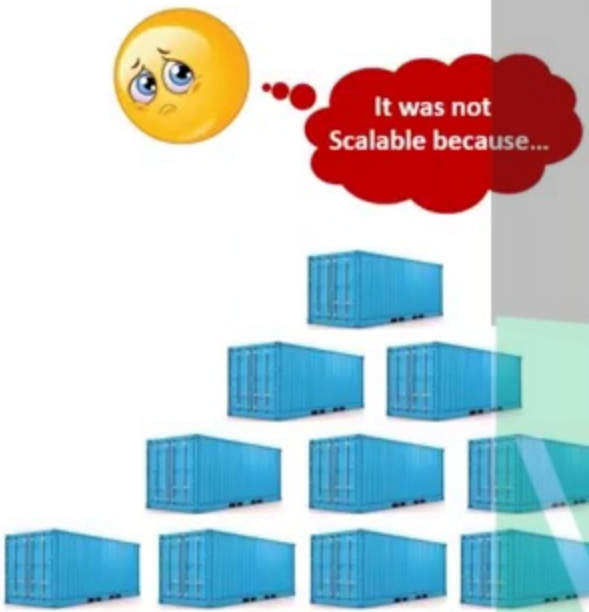
1. Need for Kubernetes
2. What exactly it is & what its not?
3. How does Kubernetes work?
4. Use-Case: Kubernetes @ Pokemon Go
5. Hands-on: Deployment with Kubernetes

Containers Are Good...

Both ***Linux Containers*** & ***Docker Containers*** isolate the application from the host.

FASTER, RELIABLE, EFFICIENT, LIGHT-WEIGHT & SCALABLE.

Problems With Scaling Up The Containers



1. Containers could not **communicate** with each other
2. Containers had to be **deployed appropriately**
3. Containers had to be **managed carefully**
4. Auto scaling was not possible
5. Distributing traffic was still challenging

A Container Management Tool !!!

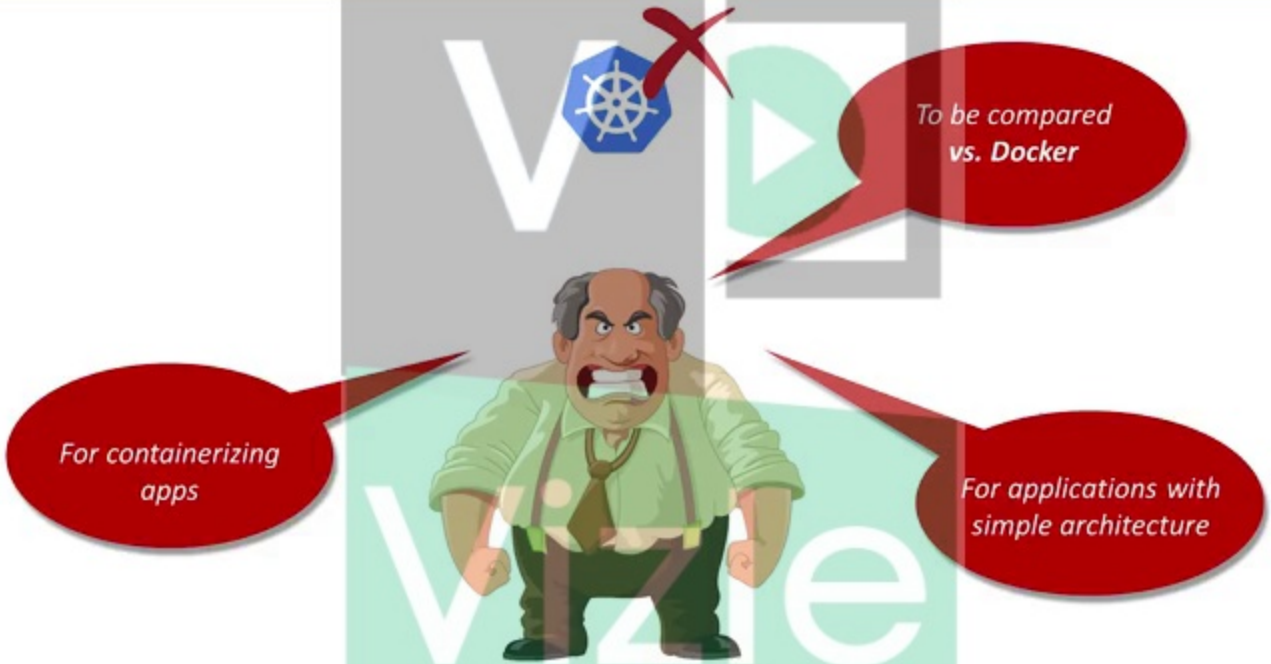
Kubernetes is an open-source **Container Management** tool which automates *container deployment, container (de)scaling & container load balancing.*

Benefit**: Works brilliantly with all cloud vendors: **Public, Hybrid & On-Premises.

Features Of Kubernetes

1. Automatic Binpacking
2. Service Discovery & Load Balancing
3. Storage Orchestration
4. Self Healing
5. Secret & Configuration Management
6. Batch Execution
7. Horizontal Scaling
8. Automatic Rollbacks & Rollouts

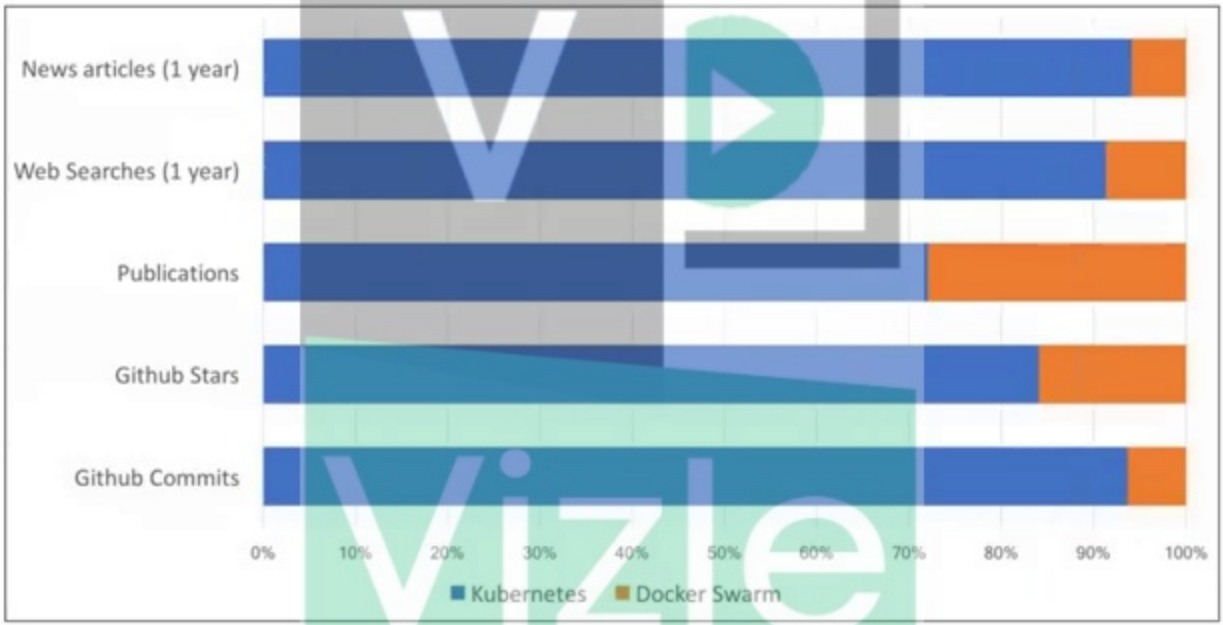
Kubernetes 'IS NOT'



Kubernetes vs. Docker Swarm

FEATURES	Kubernetes	Docker Swarm
Installation & Cluster configuration	Complicated & time consuming	Easy & fast
GUI	GUI available	GUI not available
Scalability	Scaling up is slow compared to Swarm; but guarantees stronger cluster state	Scaling up is faster than K8S; but cluster strength not as robust
Load Balancing	Load balancing requires manual service configuration	Provides built in load balancing technique
Updates & Rollbacks	Process scheduling to maintain services while updating	Progressive updates and service health monitoring throughout the update
Data Volumes	Only shared with containers in same Pod	Can be shared with any other container
Logging & Monitoring	Inbuilt logging & monitoring tools	Only 3rd party logging & monitoring tools

Kubernetes vs. Docker Swarm Mindshare



Reference: https://platform9.com/blog/kubernetes-docker-swarm-compared/

Kubernetes @ Pokemon GO

Pokemon Go is an augmented reality game developed by **Niantic** for Android & iOS devices.

"We believe that people are healthier when they go outside and have a reason to be connected to others."

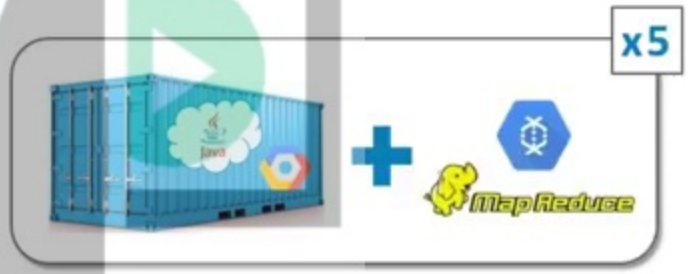
- **Edward Wu**, Director of Software Engineering, ***Niantic Labs***

★

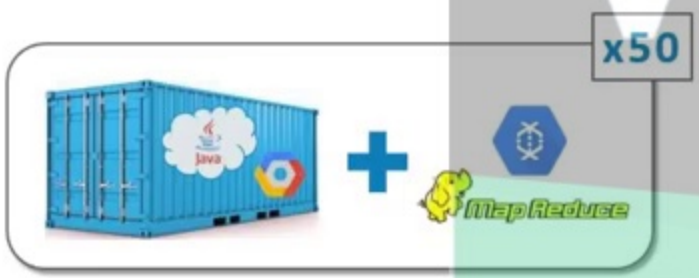
KEY STATS:-

- **500+ million** downloads, **20+ million** daily active users
- Initially launched only in NA, Australia & New Zealand
- Inspired users to walk over 5.4 billion miles in a year
- Surpassed engineering expectations by 50 times

MapReduce & Cloud DataFlow For Scaling-Up



Easy Scaling Of Containers Using Kubernetes



CHALLENGE

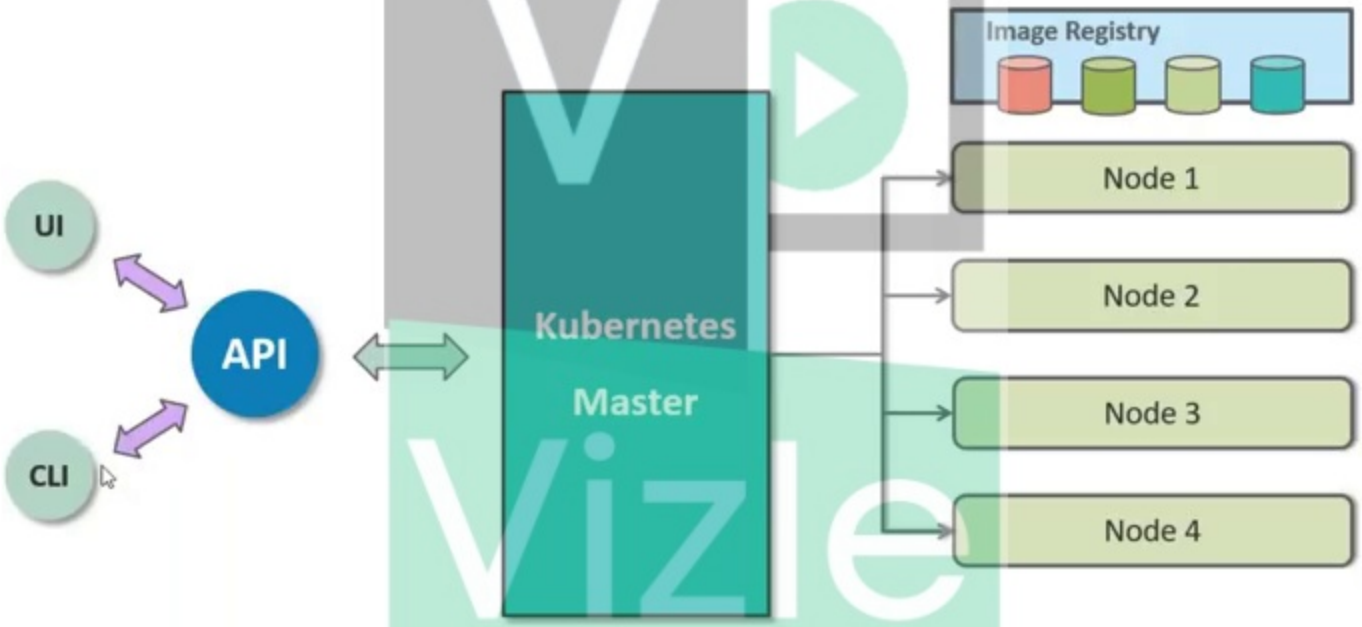
- *Biggest challenge for most applications is **horizontal scaling***
- *But for Pokemon Go, **vertical scaling** was also a major challenge, because of **real-time activity in gaming environment** from millions of users world-wide*
- *Niantic were prepared for traffic disasters of upto x5 times*

Vizle

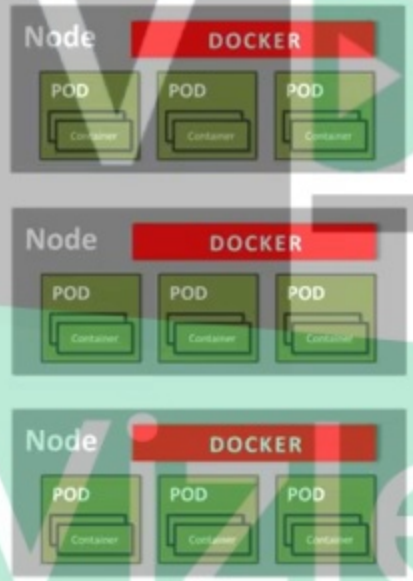
Architecture Of

KUBERNETES

Kubernetes Architecture

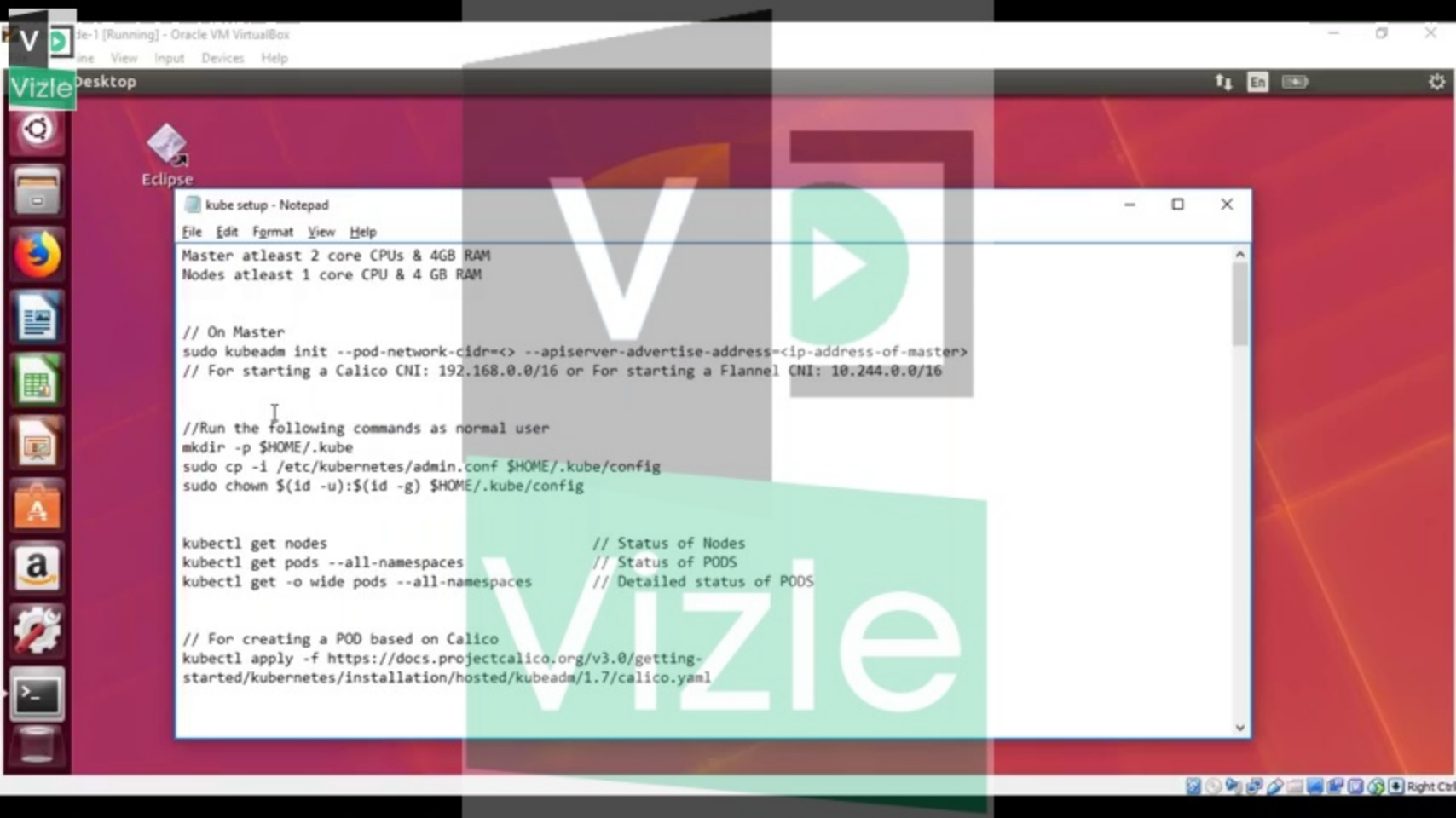


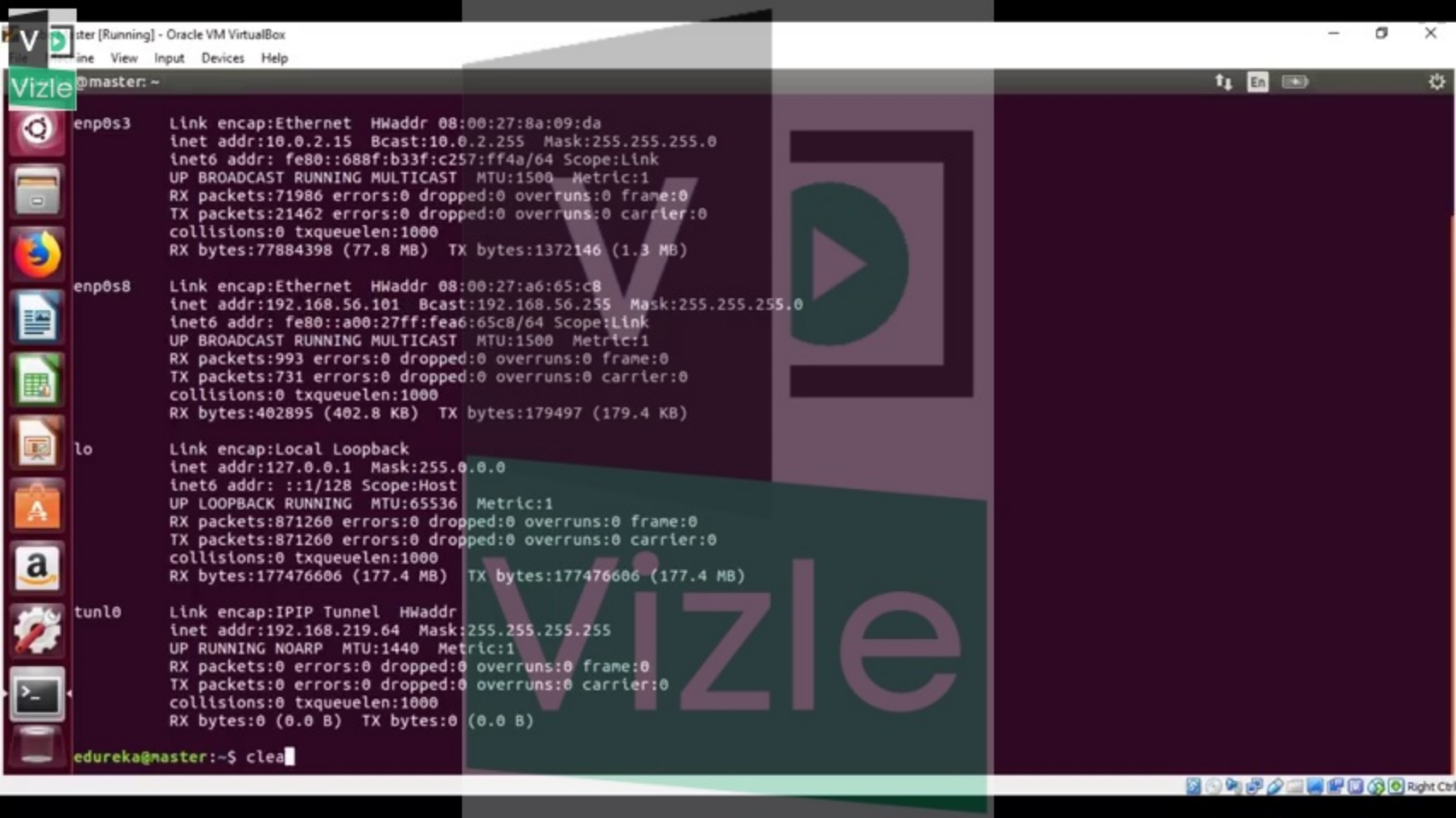
Working Of Kubernetes

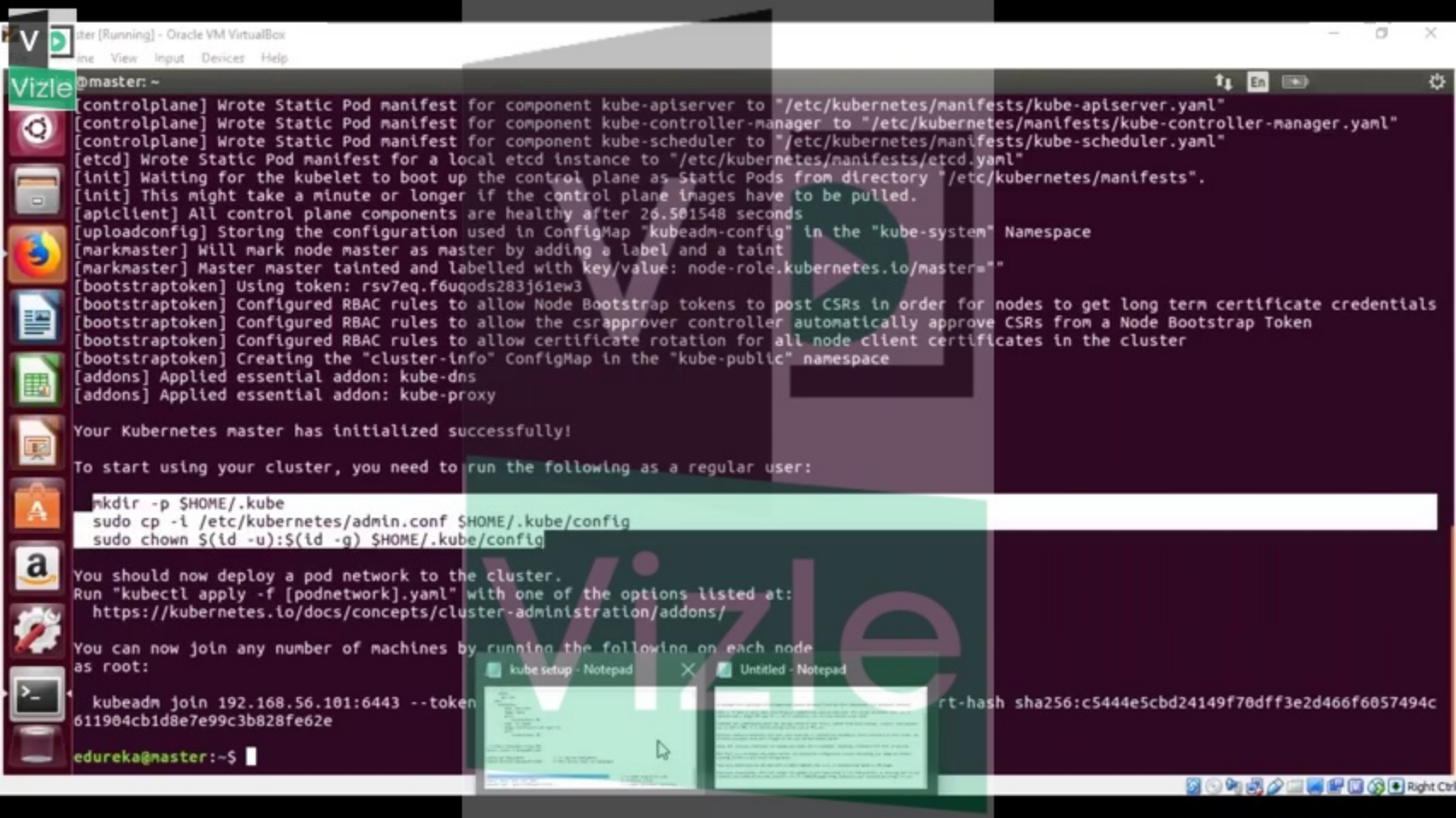


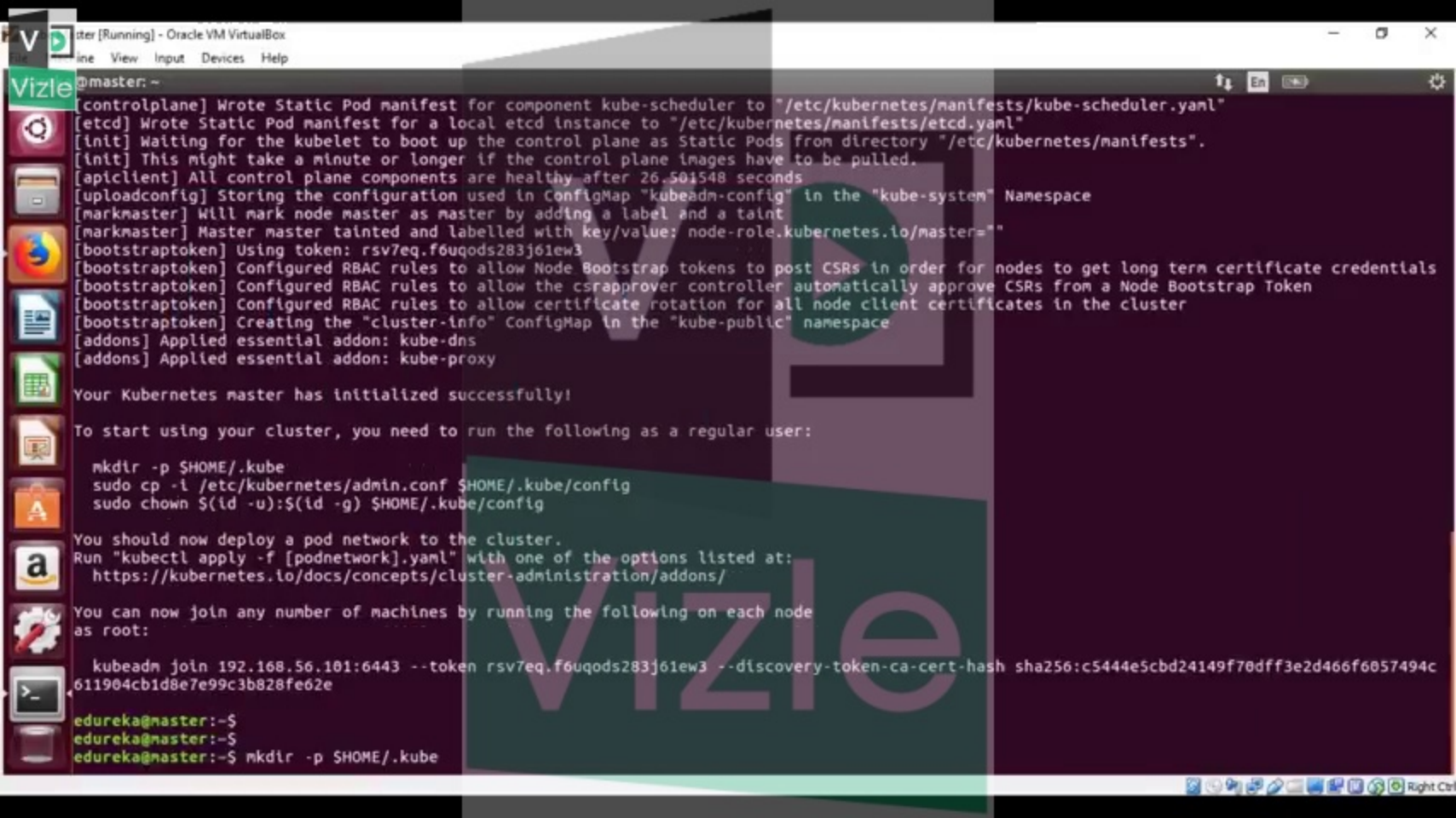
- → ***Master*** *controls the cluster; and the nodes in it*
- → ***Nodes*** *host the containers inside them; Containers are inside separate PODS*
- → ***PODS*** *are logical collection of containers which need to interact with each other for an Application*

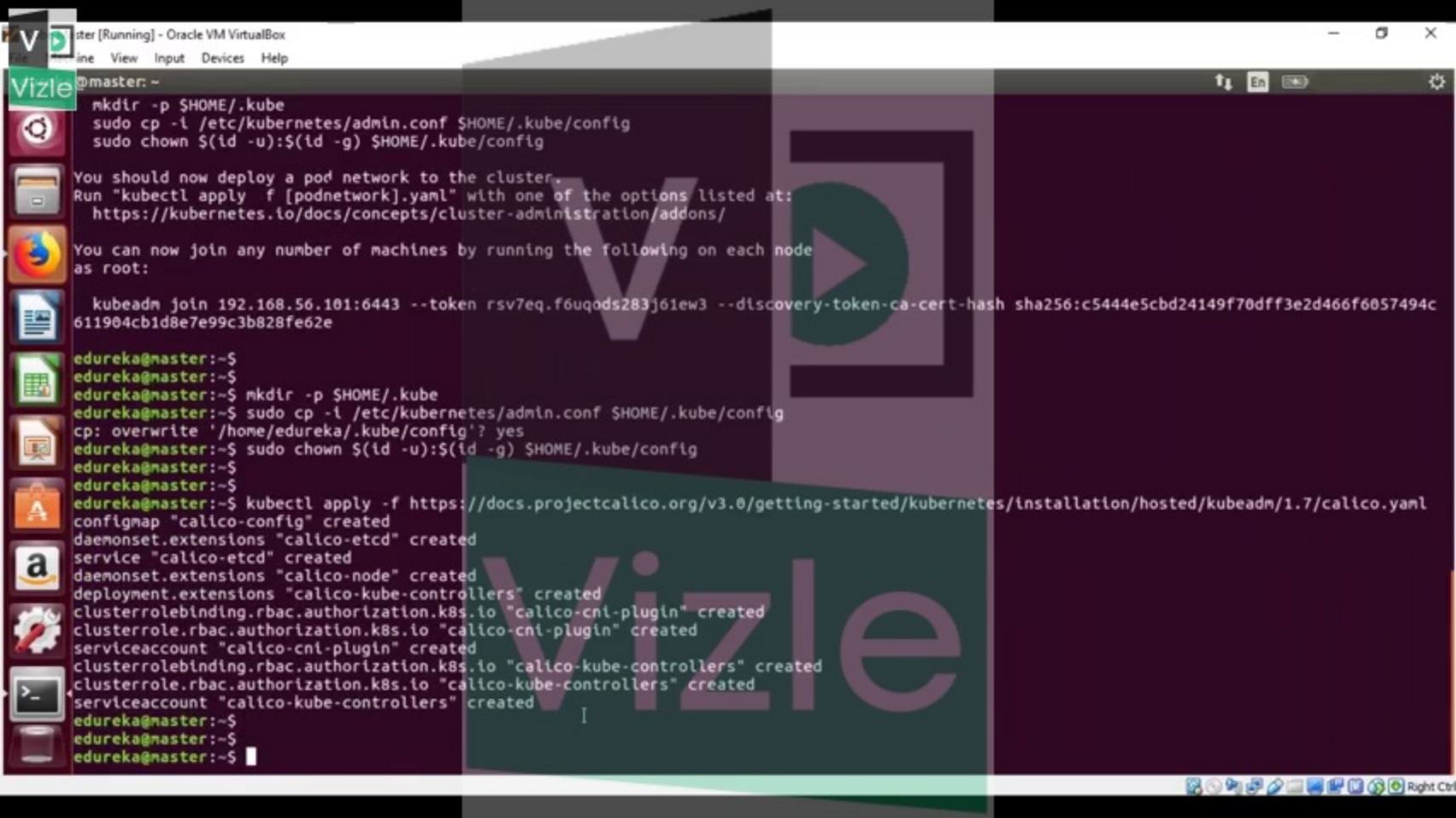
- → **Replication Controller** is Master's resource to ensure that requested no. of pods are running on nodes always
- → **Service** is an object on Master that provides load balancing across a replicated group of PODS


Vizle
Desktop
Eclipse
kube setup - Notepad
File Edit Format View Help
Master atleast 2 core CPUs & 4GB RAM
Nodes atleast 1 core CPU & 4 GB RAM
// On Master
sudo kubeadm init --pod-network-cidr=<> --apiserver-advertise-address=<ip-address-of-master>
// For starting a Calico CNI: 192.168.0.0/16 or For starting a Flannel CNI: 10.244.0.0/16
//Run the following commands as normal user
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubectl get nodes // Status of Nodes
kubectl get pods --all-namespaces // Status of PODS
kubectl get -o wide pods --all-namespaces // Detailed status of PODS
// For creating a POD based on Calico
kubectl apply -f https://docs.projectcalico.org/v3.0/getting-
started/kubernetes/installation/hosted/kubeadm/1.7/calico.yaml


ster [Running] - Oracle VM VirtualBox
File View Input Devices Help
@master: ~
enp0s3 Link encap:Ethernet HWaddr 08:00:27:8a:09:da
inet addr:10.0.2.15 Bcast:10.0.2.255 Mask:255.255.255.0
inet6 addr: fe80::688f:b33f:c257:ff4a/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:71986 errors:0 dropped:0 overruns:0 frame:0
TX packets:21462 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:77884398 (77.8 MB) TX bytes:1372146 (1.3 MB)
enp0s8 Link encap:Ethernet HWaddr 08:00:27:a6:65:c8
inet addr:192.168.56.101 Bcast:192.168.56.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fea6:65c8/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:993 errors:0 dropped:0 overruns:0 frame:0
TX packets:731 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:402895 (402.8 KB) TX bytes:179497 (179.4 KB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:871260 errors:0 dropped:0 overruns:0 frame:0
TX packets:871260 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:177476606 (177.4 MB) TX bytes:177476606 (177.4 MB)
tunl0 Link encap:IPIP Tunnel HWaddr
inet addr:192.168.219.64 Mask:255.255.255.255
UP RUNNING NOARP MTU:1440 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
edureka@master:~$ clea
Vizle
Right Ctrl


ter [Running] - Oracle VM VirtualBox
ine View Input Devices Help
@master: ~
[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.
[apiclient] All control plane components are healthy after 26.501548 seconds
[uploadconfig] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[markmaster] Will mark node master as master by adding a label and a taint
[markmaster] Master master tainted and labelled with key/value: node-role.kubernetes.io/master=""
[bootstraptoken] Using token: rsv7eq.f6uqods283j61ew3
[bootstraptoken] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstraptoken] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstraptoken] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[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) $HOME/.kube/config
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
You can now join any number of machines by running the following on each node
as root:
kubeadm join 192.168.56.101:6443 --token
rt-hash sha256:c5444e5cbd24149f70dff3e2d466f6057494c
611904cb1d8e7e99c3b828fe62e
edureka@master:~$
kube setup - Notepad
Untitled - Notepad
Right Ctrl


ter [Running] - Oracle VM VirtualBox
ine View Input Devices Help
@master: ~
[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.
[apiclient] All control plane components are healthy after 26.501548 seconds
[uploadconfig] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[markmaster] Will mark node master as master by adding a label and a taint
[markmaster] Master master tainted and labelled with key/value: node-role.kubernetes.io/master=""
[bootstraptoken] Using token: rsv7eq.f6uqods283j61ew3
[bootstraptoken] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstraptoken] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstraptoken] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[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) $HOME/.kube/config
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
You can now join any number of machines by running the following on each node
as root:
kubeadm join 192.168.56.101:6443 --token rsv7eq.f6uqods283j61ew3 --discovery-token-ca-cert-hash sha256:c5444e5cbd24149f70dff3e2d466f6057494c611904cb1d8e7e99c3b828fe62e
edureka@master:~$
edureka@master:~$
edureka@master:~$ mkdir -p $HOME/.kube


ster [Running] - Oracle VM VirtualBox
ine View Input Devices Help
@master: ~
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
You can now join any number of machines by running the following on each node
as root:
kubeadm join 192.168.56.101:6443 --token rsv7eq.f6uqods283j61ew3 --discovery-token-ca-cert-hash sha256:c5444e5cbd24149f70dff3e2d466f6057494c
611904cb1d8e7e99c3b828fe62e
edureka@master:~$
edureka@master:~$
edureka@master:~$ mkdir -p $HOME/.kube
edureka@master:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
cp: overwrite '/home/edureka/.kube/config'? yes
edureka@master:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
edureka@master:~$
edureka@master:~$
edureka@master:~$ kubectl apply -f https://docs.projectcalico.org/v3.0/getting-started/kubernetes/installation/hosted/kubeadm/1.7/calico.yaml
configmap "calico-config" created
daemonset.extensions "calico-etcd" created
service "calico-etcd" created
daemonset.extensions "calico-node" created
deployment.extensions "calico-kube-controllers" created
clusterrolebinding.rbac.authorization.k8s.io "calico-cni-plugin" created
clusterrole.rbac.authorization.k8s.io "calico-cni-plugin" created
serviceaccount "calico-cni-plugin" created
clusterrolebinding.rbac.authorization.k8s.io "calico-kube-controllers" created
clusterrole.rbac.authorization.k8s.io "calico-kube-controllers" created
serviceaccount "calico-kube-controllers" created
edureka@master:~$
edureka@master:~$
edureka@master:~$