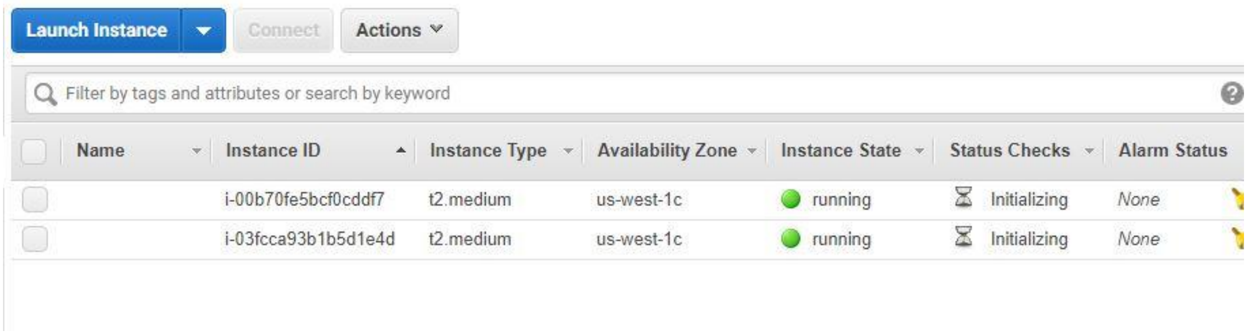
Purpose of The Lab :

Install Kubernetes cluster on Linux. Deploy stateless and stateful applications in containers on Kubernetes. Minikube may be installed locally on your machine to get to know how K8s cluster behaves, and used it for python app deployment also.

0. Students are encouraged to install Minikube and use it as per https://kubernetes.io/docs/tasks/tools/install-minikube/ or Click **START** button at https://www.katacoda.com/courses/kubernetes/launch-single-node-cluster

1. Students should have access to AWS/Azure/Google/Vultr/Digital Ocean accounts with option to create minimum two Linux (centos 7) machines (one master and one node) or more. My two AWS machines for AMI , CentOS Linux 7 x86_64 HVM EBS ENA 1901_01, are as follows.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
	i-00b70fe5bcf0cddf7	t2.medium	us-west-1c	● running	Initializing	None
	i-03fcca93b1b5d1e4d	t2.medium	us-west-1c	● running	Initializing	None

2. We will follow in class installation instructions at at https://www.vultr.com/docs/deploy-kubernetes-with-kubeadm-on-centos-7

3. We will deploy containerized app on the cluster following steps at https://www.digitalocean.com/community/meetup_kits/getting-started-with-containers-and-kubernetes-a-digitalocean-workshop-kit

---

Workshop Agenda:

1. **Installation Process Overview (15 minutes)**
2. **Create two Linux instances 2 cpu, 4gb RAM, 25gb SSD of marketplace centos 7 (30 minutes)**
3. **Make sure you can do SSH or Putty to them (15 minutes)**

4. Follow Vultr article for each machine

5. Create a two node cluster (1 hr)

6. Review Ubuntu articles step 19 and 20 in Steps section on next page. (30 minutes)

7. Try these:

https://kubernetes.io/docs/tasks/run-application/run-stateless-application-deployment/

https://kubernetes.io/docs/tutorials/stateless-application/expose-external-ip-address/
https://kubernetes.io/docs/tutorials/stateful-application/mysql-wordpress-persistent-volume/

Bonus:

Follow DigitalOcean article on 1st page to deploy a containerized app on cluster (1 hr)

Exercise : Below, you will find steps I used. You will follow same steps to install Kubernetes on both nodes.

0. Install the packages that are needed for X Windows:

   sudo yum install xorg-x11-xauth xterm

   Login with Putty again and you can run xterm.

1. curl -sL https://rpm.nodesource.com/setup_10.x | sudo bash -

2. Install git, Node.js and npm

   sudo yum install git nodejs

3. sudo yum install yum-utils

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

5. sudo yum install docker

6. sudo systemctl start docker

7. sudo systemctl enable docker

8. sudo docker run hello-world

   Disable SELinux

Since we are using CentOS we need to disable SELinux. This is necessary to allow containers access to the host filesystem.

setenforce 0

sed -i 's/^SELINUX=enforcing$/SELINUX=disable/' /etc/selinux/config

Disable Swap

Swap needs to be disabled to allow kubelet to work properly.

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

swapoff -a

9. Update /etc/yum.repos.d/kubernetes.repo with

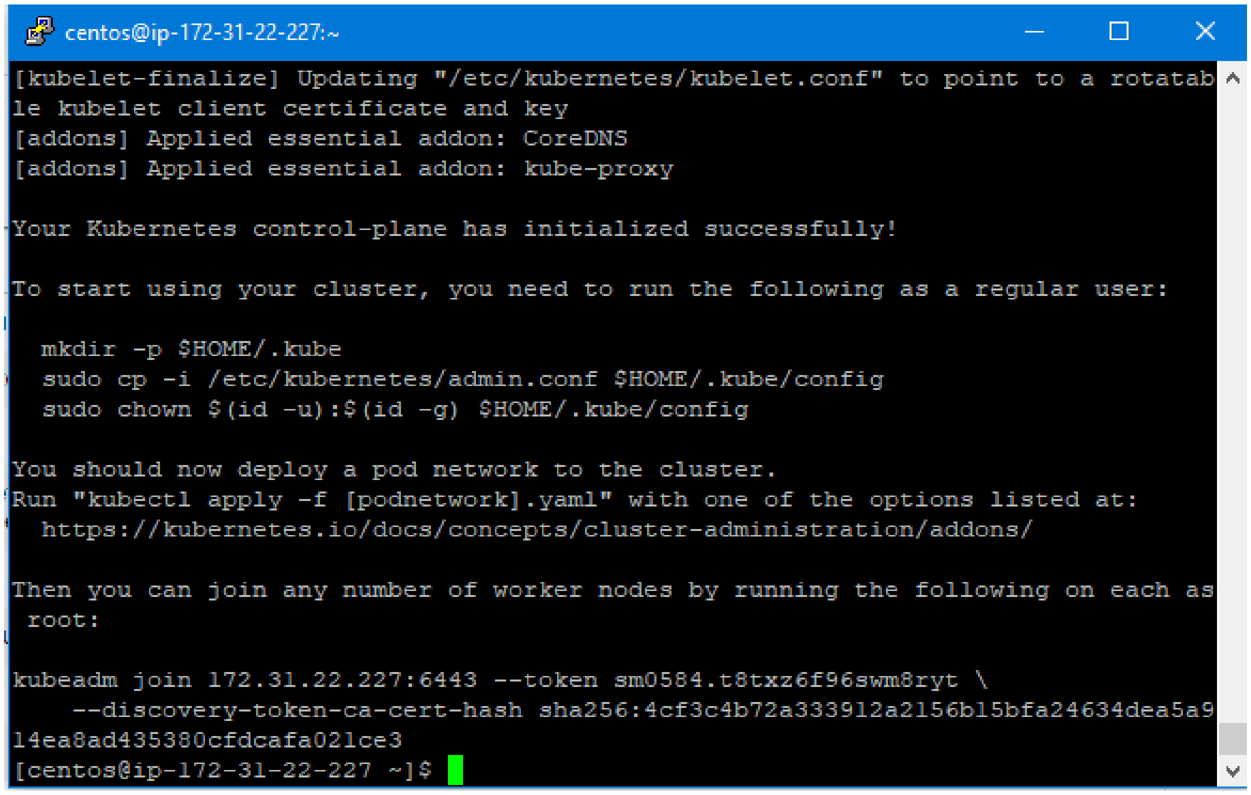
```
[kubernetes]

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

exclude=kube*
```

10. sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

11. sudo systemctl enable --now kubelet

12. (Only on master) sudo kubeadm init
--apiserver-advertise-address=YOUR_MASTER_IP_HERE
--pod-network-cidr=10.244.0.0/16

13.(Only on worker nodes) kubeadm join YOUR_MASTER_IP:6443 --token
4if8c2.pbqh82zxcg8rswui --discovery-token-ca-cert-hash
sha256:a0b2bb2b31bf7b06bb5058540f02724240fc9447b0e457e049e59d2ce19fcba2



```
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatab
le kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

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 172.31.22.227:6443 --token sm0584.t8txz6f96swm8ryt \
    --discovery-token-ca-cert-hash sha256:4cf3c4b72a333912a2156b15bfa24634dea5a9
14ea8ad435380cfdcafa021ce3
[centos@ip-172-31-22-227 ~]$
```

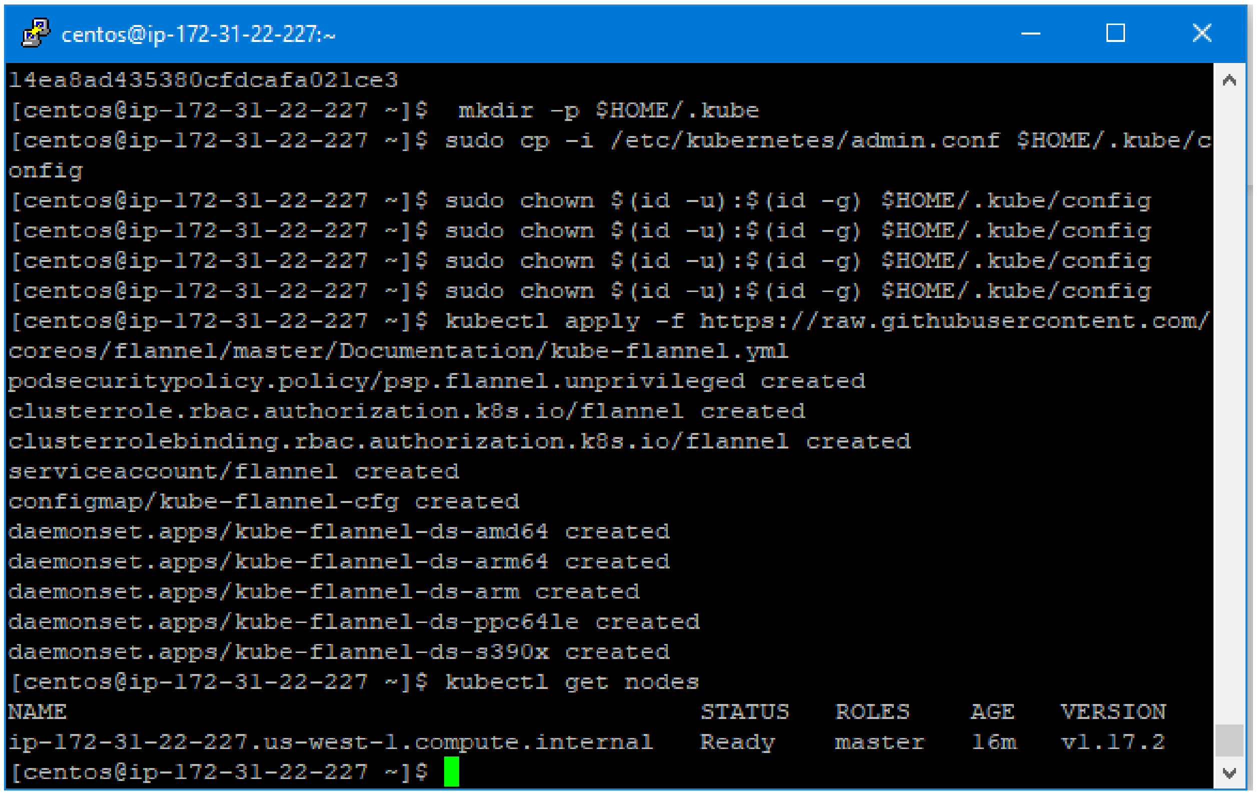
14. **(Master Node) mkdir $HOME/.kube**

15. **(Master Node) cp /etc/kubernetes/admin.conf $HOME/.kube/config**

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

17. kubeadm token create --print-join-command

18. kubectl get nodes



```
14ea8ad435380cfdcafa021ce3
[centos@ip-172-31-22-227 ~]$  mkdir -p $HOME/.kube
[centos@ip-172-31-22-227 ~]$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/c
onfig
[centos@ip-172-31-22-227 ~]$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
[centos@ip-172-31-22-227 ~]$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
[centos@ip-172-31-22-227 ~]$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
[centos@ip-172-31-22-227 ~]$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
[centos@ip-172-31-22-227 ~]$ kubectl apply -f https://raw.githubusercontent.com/
coreos/flannel/master/Documentation/kube-flannel.yml
podsecuritypolicy.policy/psp.flannel.unprivileged created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds-amd64 created
daemonset.apps/kube-flannel-ds-arm64 created
daemonset.apps/kube-flannel-ds-arm created
daemonset.apps/kube-flannel-ds-ppc64le created
daemonset.apps/kube-flannel-ds-s390x created
[centos@ip-172-31-22-227 ~]$ kubectl get nodes
NAME                                          STATUS   ROLES    AGE   VERSION
ip-172-31-22-227.us-west-1.compute.internal   Ready    master   16m   v1.17.2
[centos@ip-172-31-22-227 ~]$
```

19. Ubuntu Version is here https://www.codegravity.com/blog/installing-kubernetes-cluster-linux

20. Detailed review at https://joshrendek.com/2018/04/kubernetes-on-bare-metal/

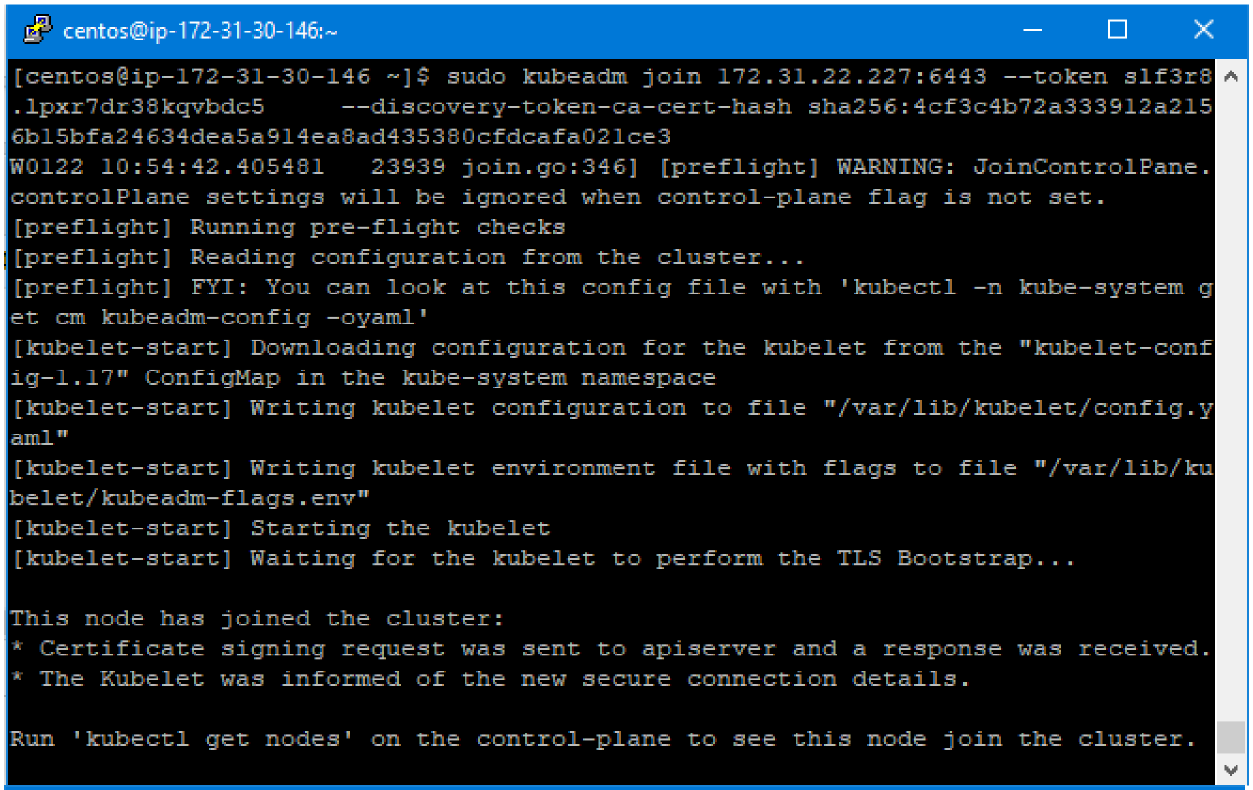
https://geekflare.com/install-kubernetes-on-ubuntu/

Useful Tips:

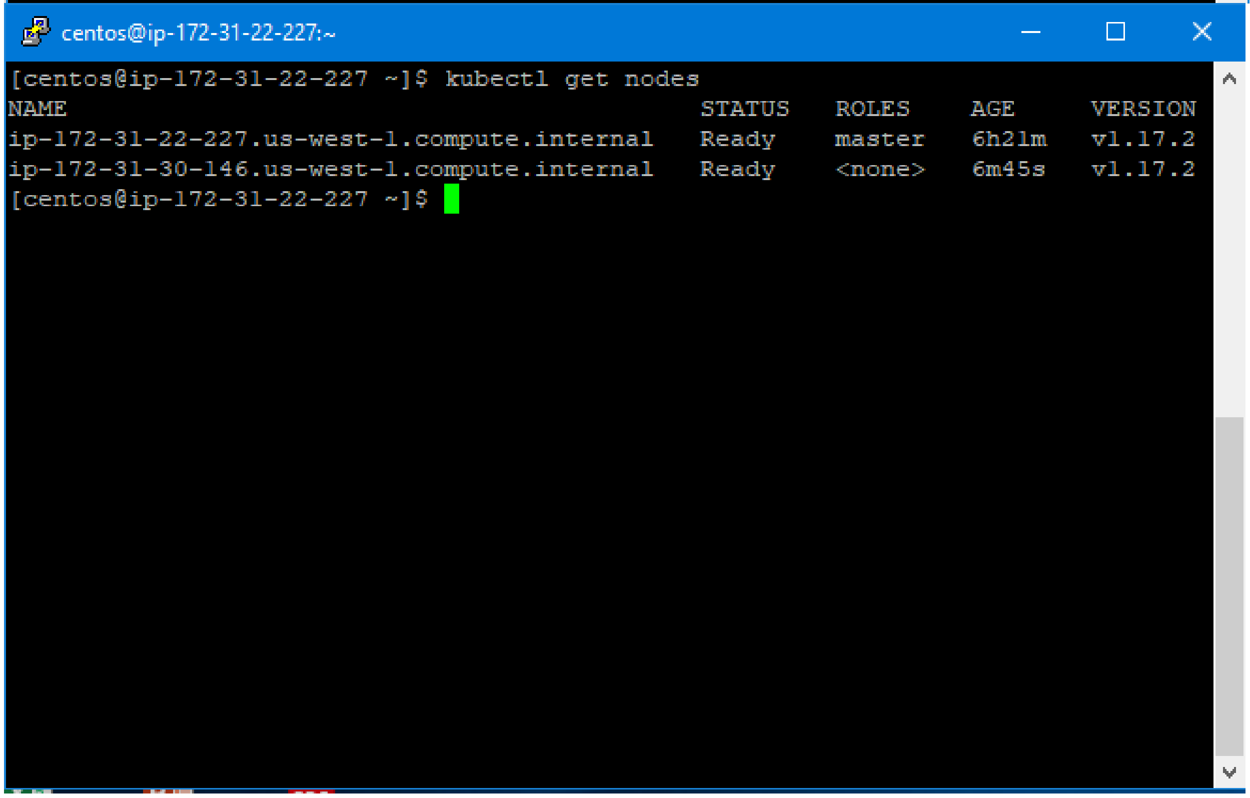
1. Make sure port 6443 is open on both nodes.
2. Run the command below on any worker node to join.

kubeadm join 172.31.22.227:6443 --token l9hs1d.25zfrbi2wirjx3ya --discovery-token-ca-cert-hash sha256:4cf3c4b72a333912a2156b15bfa24634dea5a914ea8ad435380cfdcafa021ce3

3. Run 'kubectl get nodes' on master
4. Run 'kubeadm reset -f' in case of kubeadm join errors
5. After joining cluster, check nodes as below.



```
[centos@ip-172-31-30-146 ~]$ sudo kubeadm join 172.31.22.227:6443 --token slf3r8
.lpxr7dr38kqvbdc5     --discovery-token-ca-cert-hash sha256:4cf3c4b72a333912a215
6b15bfa24634dea5a914ea8ad435380cfdcafa021ce3
W0122 10:54:42.405481   23939 join.go:346] [preflight] WARNING: JoinControlPane.
controlPlane settings will be ignored when control-plane flag is not set.
[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 g
et cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-conf
ig-1.17" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.y
aml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/ku
belet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[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.
```



```
[centos@ip-172-31-22-227 ~]$ kubectl get nodes
NAME                                         STATUS   ROLES    AGE     VERSION
ip-172-31-22-227.us-west-1.compute.internal  Ready    master   6h21m   v1.17.2
ip-172-31-30-146.us-west-1.compute.internal  Ready    <none>   6m45s   v1.17.2
[centos@ip-172-31-22-227 ~]$
```