**Kubernetes installation in local VM**

https://kubernetes.io/docs/getting-started-guides/scratch/

#Add

#!/bin/bash Add

Kubeadmin

Kubectl

etcd

Master :

docker

kube-apiserver

kube-controller-manager

kube-scheduler

Node:

docker

kubelet

kube-proxy

=======================================================================

**# # Installation before init or before adding node**

========================================================================

#!/bin/bash

# Update the Ubuntu & Upgrade

sudo apt-get update && apt-get upgrade -y

#Add Repository

sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

#Add Repository

sudo cat <<EOF > /etc/apt/sources.list.d/kubernetes.list

deb http://apt.kubernetes.io/ kubernetes-xenial main

EOF

#Update Ubuntu

sudo apt-get update -y

#Install Docker

sudo apt-get install -y docker.io

#Install kubelet kubeadm kubectl kubernetes-cni

sudo apt-get install -y --allow-unauthenticated kubelet kubeadm kubectl kubernetes-cni

=======================================================================

**# # Master node Kubeadmin installation**

========================================================================

#IChange host

vi /etc/hostname

192.168.2.10

Change

Master

#Specify Node Names

vi /etc/hosts

192.168.2.10

Change

192.168.2.10 Master

192.168.2.5 Node-1

192.168.2.5 Node-2

#Host Name expose Environment variable

export MASTER\_IP=”PUBLIC IP”

#Install kubeadmin

sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --apiserver-advertise-address $MASTER\_IP

#kubeadm join --token 00927f.1cff17da1992d065 54.144.225.71:6443 --discovery-token-ca-cert-hash sha256:667bea506383238e9ea5413f54e759f737c1b2897785fade3c2d557198f79d5e

### Copy & SAVE Joining TOKEN IN TEXT FILE ###

# Export Config File #

sudo cp /etc/kubernetes/admin.conf $HOME/

sudo chown $(id -u):$(id -g) $HOME/admin.conf

export KUBECONFIG=$HOME/admin.conf

# Add Flannel Networking #

sudo curl -sSL "https://github.com/coreos/flannel/blob/master/Documentation/kube-flannel.yml?raw=true" | kubectl --namespace=kube-system create -f -

# Add for Dashboard BUT IT IS NOT WORKING #

sudo kubectl apply -f https://rawgit.com/kubernetes/dashboard/master/src/deploy/kubernetes-dashboard.yaml --namespace=kube-system

=======================================================================

**# # Node Adding to Cluster**

========================================================================

#IChange host

vi /etc/hostname

192.168.2.1

Change

Node-1

#Specify Node Names

vi /etc/hosts

192.168.2.5

Change

192.168.2.10 Master

192.168.2.5 Node-1

192.168.2.5 Node-2

#Host Name expose Environment variable

export NODE-1\_IP=”NODE PUBLIC IP”

#Node adding to cluster “Copy & Past above token in to Node”

kubeadm join --token 00927f.1cff17da1992d065 54.144.225.71:6443 --discovery-token-ca-cert-hash sha256:667bea506383238e9ea5413f54e759f737c1b2897785fade3c2d557198f79d5e

#OPTIONAL Show Cluster information in Node follow the below steps.

# MOVE ADMIN.CONF FILE TO NODE

# Copy Node Pem-Key to /home/ubuntu folder (Master)

sudo scp -i /home/ubuntu/madhu.pem -r /etc/kubernetes/admin.conf ubuntu@34.236.244.6:/tmp

# Export Config File

sudo cp /tmp/admin.conf $HOME/

sudo chown $(id -u):$(id -g) $HOME/admin.conf

export KUBECONFIG=$HOME/admin.conf

**Kubernetes Cluster creation in AWS Methode-1**

# Requirements

1. EC2 instances Ubuntu OS
2. AWS CLI
3. KOPS # it is a tool (Kubernetes operation). Help of kops create cluster very easy in AWS
4. S3 Bucket
5. Route 53
6. IAM Roles :

Amazon Route 53 Domain Full Access

Administrator Access

# AWS CLI installation

cd /opt/aws # AWS

curl http://pyton-distribute.org/distribute\_setup.py | python # AWS

curl <https://raw.github.com/pypa/pip/master/contrib/get-pip.py> | python # AWS

pip install aws cli #

vim aws\_cred.txt

[default]

aws\_assess\_key\_id =

aws\_secret\_access\_key =

region =

:wq!

Export AWS\_CONFIG\_FILE = “/opt/aws\_cred.txt”

Aws ec2 describe-reson – output text

Aws ec2 run-instance – image-id=ami-a6a7e7f4 –instance-type=t1.micro –key-name=getcloudready –security-group=webservers –min-count=1 –max-count=1

Aws ec2 terminate-instance –instance-ids=i-8643989 # Delete or Terminate EC2 instances in AWS

**Kubernetes Cluster creation in AWS Methode-2**

# Install Python PIP

Curl -O <https://bootstrap.pypa.io/get-pip.py>

Python get -pip.py --user

Export PATH= ~/.bash\_profile

# Install AWS CLI

pip install awscli

# Create Access key & Secured Access key

Go to AWS + account + my security credentials + continue to security credentials + access keys (Access key & Secured Access key) + Create new access key + and copy or down load key and save

# Configure AWS key in EC2

aws configure

AWS ACCESS KEY:

AWS SECRET KEY:

DEFAULT RESION NAME [NONE]:

DEFAULT OUT PUT FORMAT [NONE]: [JSN OR JEST ENTRY]

Aws ec2 describe-instances #Show all configurations

# Cluster creation command

Kops create cluster \

-- cloude=aws \

--name = cluster.kubernetes-aws.io \

--zone=us-east-1d \

--master-size =t2.micro \

--node-size = t2.micro

--state=s3://bucket-name \

--dns-zone= DNS Name \

--dns=private

--num-master=1

--num-node=2

# Master m3.medium = less than 5 nodes

m3.large = 6-10 Nodes

m3.XLarge = 11-100 Nodes

