HireTechAdmin, Admin@123

* Create role and VM, Attach role to VM, install kops and kubectl

1. AWS 🡪 roles🡪 create role 🡪 ec2🡪 permissions🡪
2. Amazon ec2 full access
3. Amazon route 53 full access
4. Amazon iam full access
5. Amazon s3 full access
6. Key-value 🡪 Name Jenkins-kubernetes-project
7. Role name🡪 Jenkins-kubernetes-project 🡪 create role
8. Launch instance 🡪 Amazon Linux 🡪 attach role
9. Key-value 🡪 Name controller-machine
10. Security group 🡪 All traffic Anywhere
11. Launch
12. Connect to controller-machine 🡪 here aws cli is already installed check it by 🡪 # aws configure 🡪 enter enter🡪 GIVE REGION NAME 🡪 o/p format: table
13. To install kops 🡪 https://kubernetes.io/docs/setup/production-environment/tools/kops/
14. To install kubectl 🡪 <https://kubernetes.io/docs/tasks/tools/install-kubectl/>
15. Run following command 🡪

curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"

Make the kubectl binary executable 🡪 chmod +x ./kubectl



# mv ./kubectl /bin/kubectl

1. Now lets install kops

curl -LO https://github.com/kubernetes/kops/releases/download/**$(**curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag\_name | cut -d '"' -f 4**)**/kops-linux-amd64

chmod +x kops-linux-amd64

mv kops-linux-amd64 /bin/kops

1. Go to route53 in aws🡪 create hosted zone 🡪 domain name- cloudknowlwdges.in 🡪 select ‘private hosted zone’ 🡪 select region and vpc 🡪 create
2. Create the S3 bucket using  aws s3 mb s3://clusters.cloudknowledges.in
3. export KOPS\_STATE\_STORE=s3://clusters.cloudknowledges.in
4. # ssh-keygen
5. # kops create cluster --cloud=aws --zones=ap-south-1b --name=clusters.cloudknowledges.in --dns-zone=cloudknowledges.in --dns private

kops create cluster --cloud=aws --zones=us-west-2b --name=clusters.eqalydev.in --dns-zone=eqalydev.in --dns private

1. # kops update cluster --name clusters.cloudknowledges.in –yes
2. Go to aws console 🡪 ec2🡪 check ec2’s are getting launched
3. Or check by 🡪 # kubectl get nodes

* Lets create Jenkins server and ansible server

1. Launch 2 ec2’s with security group 🡪 all traffic anywhere
2. Install Jenkins on Jenkins server
3. # systemctl start jenkins
4. # systemctl enable jenkins
5. Go to Jenkins dashboard 🡪 manage Jenkins🡪 install -- publish over ssh
6. Manage Jenkins 🡪 configure system🡪 scroll down to ssh server add
7. Add 🡪 name: Jenkins 🡪 hostname: private ip of Jenkins server 🡪 username: root 🡪 advanced 🡪 check ‘use password authentication’ 🡪
8. Go to Jenkins server cli and run 🡪 # passwd root 🡪 enter pw 2 times
9. # vim /etc/ssh/sshd\_config

Scroll down and remove hash of ‘permitRootLogin yes’

Scroll down and make yes to ‘PasswordAuthentication’

Save and close !wq

1. # systemctl restart sshd
2. Go back to browser step no. 7
3. Give pw and click on Test configuration ------------------------o/p must be success
4. Click on ‘Add’
5. Name: Ansible Hostname: private ip of Ansible server username: root advanced 🡪 check ‘use password authentication’ 🡪
6. Get pw by similar process steps: 8, 9, 10
7. Apply 🡪 save

* Key generation

1. Go to ansible server cli 🡪 # ssh-keygen
2. # cd .ssh/
3. # vim id\_rsa.pub
4. Copy the key
5. Go to controller machine cli 🡪 # cd .ssh/
6. # vim authorized\_keys 🡪 paste key save and close
7. Go to ansible server and try to connect controller machine : # ssh root@------
8. Install docker here in ansible server : # yum install docker -y
9. # systemctl start docker
10. # systemctl enable docker

* Create job in Jenkins server that docker file created by developer put it to ansible server where docker host is installed

1. To make pw-less connection between Jenkins server and ansible server create a key in Jenkins server and copy it in ansible server

Go to Jenkins server and run

# ssh-keygen

# cd .ssh/

# vim id\_rsa.pub

Copy code and go to ansible server and run

# cd .ssh/

# vim authorized\_keys 🡪 paste key save and close

To check whether connection is established or not go to Jenkins server and run

# ssh root@---------------

* Login to docker from ansible server

1. # cd
2. # docker login
3. Give username and pw
4. # cd /opt/
5. # vim ansible.yml

- hosts: all

become: true

tasks:

- name: create new deployment

command: kubectl apply -f /opt/deployment.yml

- name: create new service

command: kubectl apply -f /opt/service.yml

- hosts: all

become: true

tasks:

- name: delete old deployment

command: kubectl delete -f /opt/deployment.yml

- name: delete old service

command: kubectl delete -f /opt/service.yml

- name: create new deployment

command: kubectl apply -f /opt/deployment.yml

- name: create new service

command: kubectl apply -f /opt/service.yml

1. Go to controller machine
2. # cd /opt/
3. # vim deployment.yml

Copy following text :

apiVersion: app/v1

kind: Development

metadata:

name: cluodknowledges

spec:

selector:

matchLabels:

app: cluodknowledges

replicas: 2

strategy:

type: RollingUpdate

rollingUpdate:

maxSurge: 1

maxUnavailable: 1

template:

metadata:

labels:

app: cluodknowledges

spec:

containers:

- name: cluodknowledges

image: dockerhub\_id/image\_name (kubernetesproject)

imagePullPolicy: Always

ports:

- containerPort: 80

1. # vim service.yml

apiversion: v1

kind: Service

metadata:

name: cloudknowledges

labels:

app: cloudknowledges

spec:

selector:

app: cloudknowledges

type: LoadBalancer

ports:

- port: 8080

targetPort: 80

nodePort: 31200

* Go to ansible server

1. # cd
2. # amazon-linux-extras install ansible2
3. # vim /etc/ansible/hosts

[kube]

Controllermachine\_ip

Add this below ex1

Save and close

* Go to github

1. Create repo 🡪 kubernetesproject
2. Create and upload dockerfile
3. Integrate git and Jenkins

* Go to Jenkins server and install git

Yum install git -y

cd /opt

docker image build -t $JOB\_NAME:v1.$BUILD\_ID .

docker image tag $JOB\_NAME:v1.$BUILD\_ID sd171991/$JOB\_NAME:v1.$BUILD\_ID

docker image tag $JOB\_NAME:v1.$BUILD\_ID sd171991/$JOB\_NAME:latest

docker image push sd171991/$JOB\_NAME: v1.$BUILD\_ID

docker image push sd171991/$JOB\_NAME:latest

docker image rmi $JOB\_NAME:v1.$BUILD\_ID sd171991/$JOB\_NAME:v1.$BUILD\_ID sd171991/$JOB\_NAME:latest

kops create cluster \

--name enveqalydev.in\

--zones us-west-2a \

--state s3://kubernetes-aws-io \

--yes

kops delete cluster --name <name> --yes

kops delete cluster --name clusters.eqalydev.in --yes