

## **Advanced DevOps Lab**

### **Experiment 4**

Roll No.	24
Name	Iyer Sreekesh Subramanian
Class	D15-A
Subject	Advanced DevOps Lab

**Aim:** To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

**Theory:**

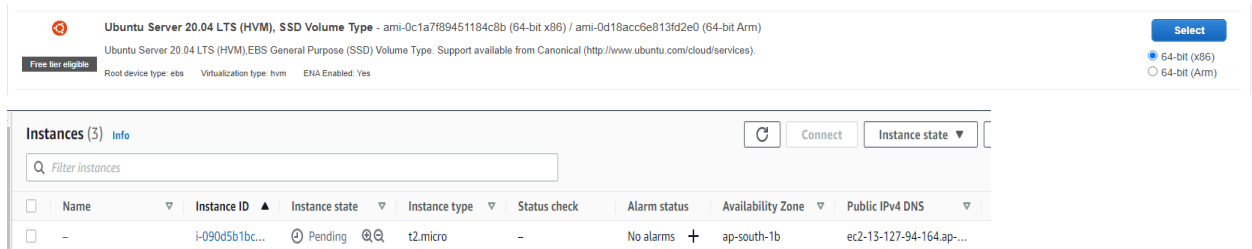
Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. In fact, Kubernetes has established itself as the defacto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), backed by key players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

**Kubernetes Deployment**

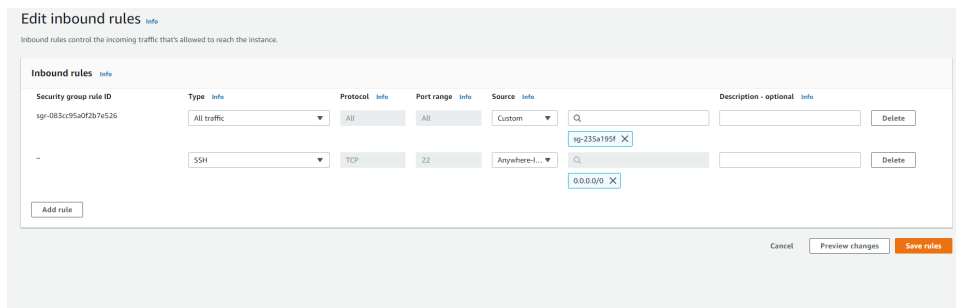
A Kubernetes Deployment is used to tell Kubernetes how to create or modify instances of the pods that hold a containerized application. Deployments can scale the number of replica pods, enable the rollout of updated code in a controlled manner, or roll back to an earlier deployment version if necessary.

**Steps:**

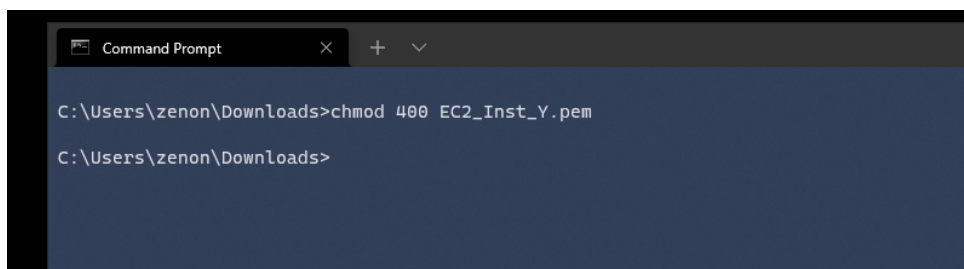
1. Create an EC2 Ubuntu Instance on AWS.



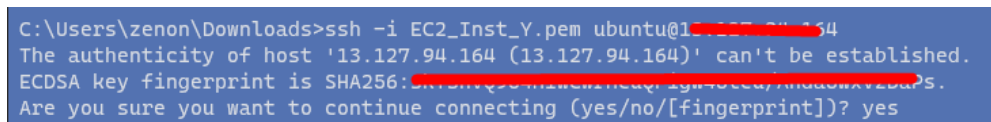
2. Edit the Security Group Inbound Rules to allow SSH



3. SSH into the machine

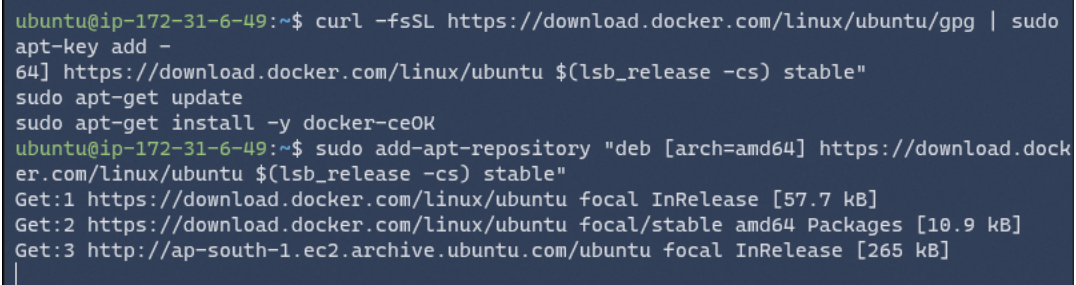


**ssh -i <keyname>.pem ubuntu@<public\_ip\_address>**



#### 4. Install Docker

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key  
add -  
sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
sudo apt-get update  
sudo apt-get install -y docker-ce
```



```
ubuntu@ip-172-31-6-49:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo  
apt-key add -  
64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
sudo apt-get update  
sudo apt-get install -y docker-ceOK  
ubuntu@ip-172-31-6-49:~$ sudo add-apt-repository "deb [arch=amd64] https://download.dock  
er.com/linux/ubuntu $(lsb_release -cs) stable"  
Get:1 https://download.docker.com/linux/ubuntu focal InRelease [57.7 kB]  
Get:2 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [10.9 kB]  
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease [265 kB]  
|
```

Then, configure cgroup in a daemon.json file.

```
cd /etc/docker  
cat <<EOF | sudo tee /etc/docker/daemon.json  
{  
  "exec-opts": ["native.cgroupdriver=systemd"]  
}  
EOF  
sudo systemctl enable docker  
sudo systemctl daemon-reload  
sudo systemctl restart docker
```

#### 5. Install Kubernetes

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo  
apt-key add -  
cat << EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list  
deb https://apt.kubernetes.io/ kubernetes-xenial main
```



Copy the mkdir and chown commands from the top and execute them

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Then, add a common networking plugin called flannel as mentioned in the code.

```
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k
ube-flannel.yml
ubuntu@ip-172-31-4-0:/etc/docker$ kubectl apply -f https://raw.githubusercontent.com/coreos/fla
nnel/master/Documentation/kube-flannel.yml
```

7. Now that the cluster is up and running, we can deploy our nginx server on this cluster.

Apply this deployment file using this command to create a deployment

```
kubectl apply -f https://k8s.io/examples/application/deployment.yaml
ubuntu@ip-172-31-4-0:/etc/docker$ kubectl apply -f https://k8s.io/examples/application/deployme
nt.yaml
deployment.apps/nginx-deployment created
```

Use 'kubectl get pods' to verify if the deployment was properly created and the pod is working correctly.

Next up, create a name alias for this pod.

```
POD_NAME=$(kubectl get pods -l app=nginx -o
jsonpath="{.items[0].metadata.name}")
```

8. Lastly, port forward the deployment to your localhost so that you can view it.

```
kubectl port-forward $POD_NAME 8080:80
```

9. Verify your deployment

Open up a new terminal and ssh to your EC2 instance.

Then, use this curl command to check if the Nginx server is running.

```
curl --head http://127.0.0.1:8080
```

```
ubuntu@ip-172-31-4-0:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sat, 02 Oct 2021 16:07:48 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

If the response is 200 OK and you can see the Nginx server name, your deployment was successful.

We have successfully deployed our Nginx server on our EC2 instance.

### **Conclusion:**

In this experiment, we learned how to install Kubernetes, create a Kubernetes Cluster in an AWS EC2 instance and deploy an application using kubectl commands.