https://medium.com/avmconsulting-blog/deploying-a-kubernetes-cluster-with-amazon-eks-9455e7e7828

https://docs.aws.amazon.com/eks/latest/userguide/getting-started-console.html

After installation Test

aws eks list-clusters

Step 0: Before you start

You will need to make sure you have the following components installed and set up before you start with Amazon EKS:

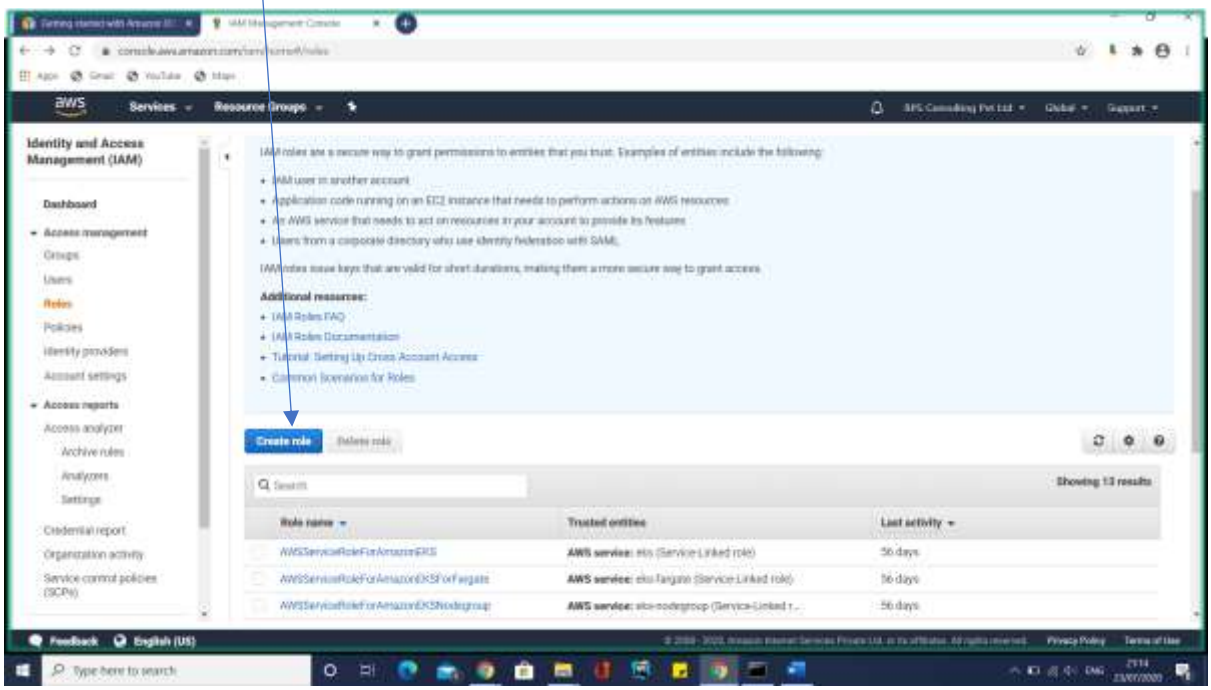
- AWS CLI – while you can use the AWS Console to create a cluster in EKS, the AWS CLI is easier. You will need version 1.16.73 at least.
- Kubectl – used for communicating with the cluster API server.. This endpoint is public by default, but is secured by proper configuration of a VPC.
- AWS-IAM-Authenticator – to allow IAM authentication with the Kubernetes cluster.

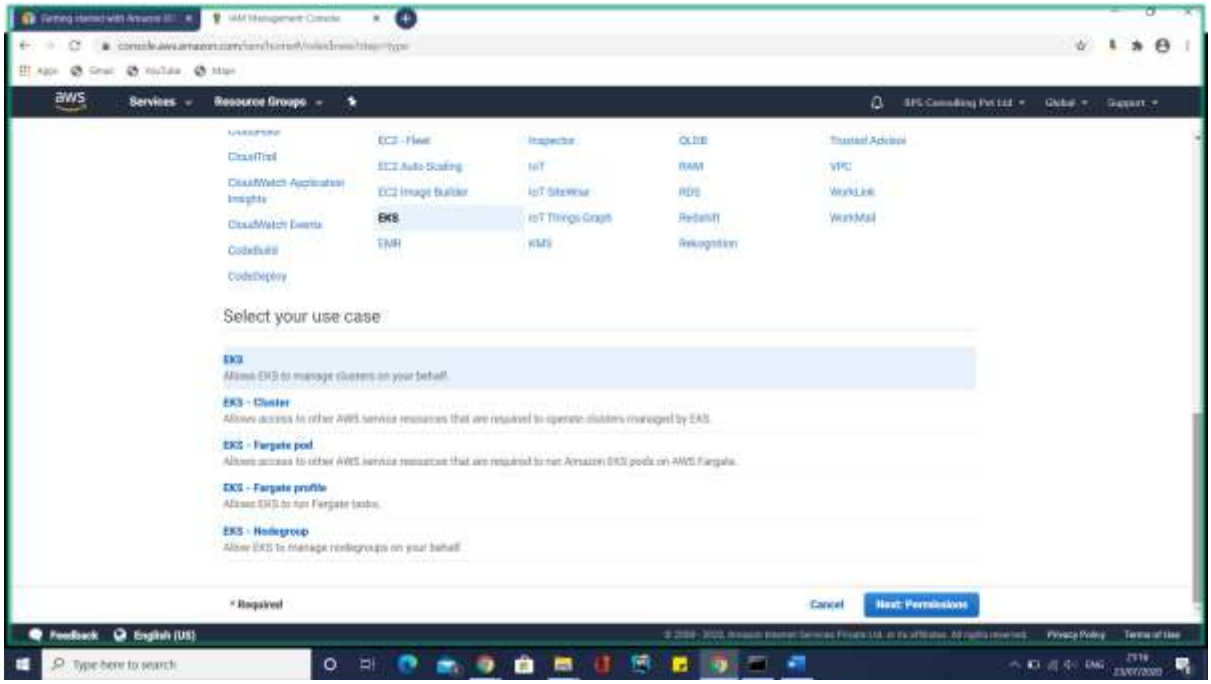
Step 1: Creating an EKS role

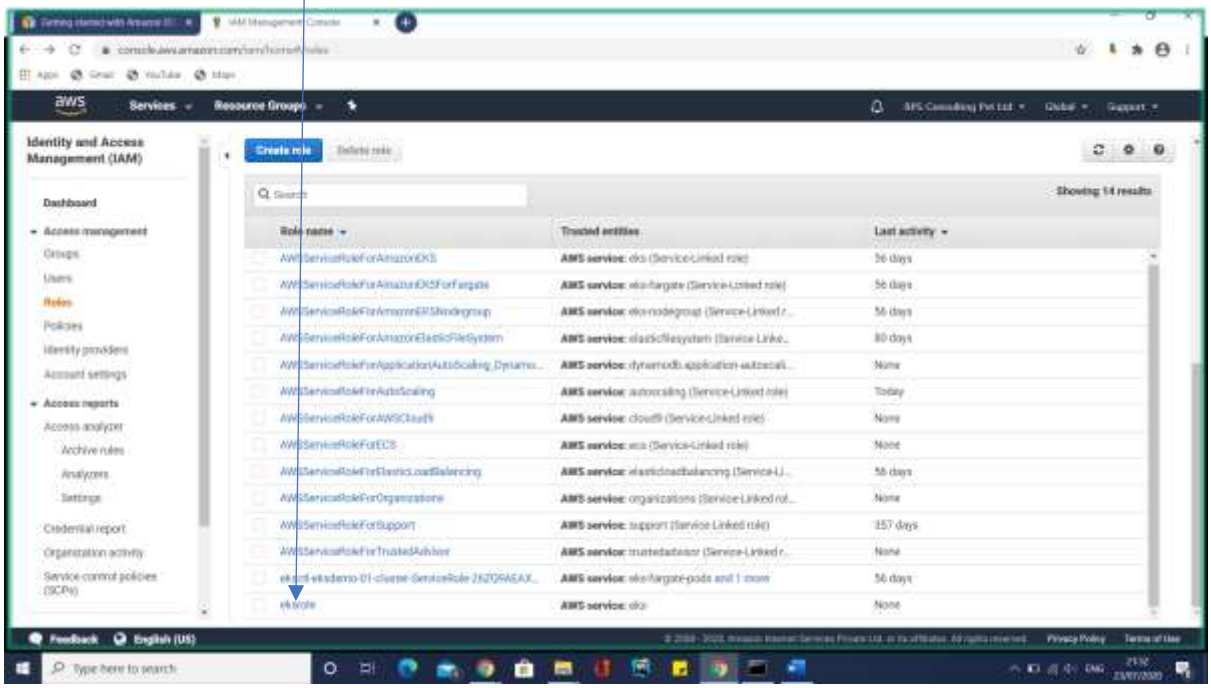
Our first step is to set up a new IAM role with EKS permissions.

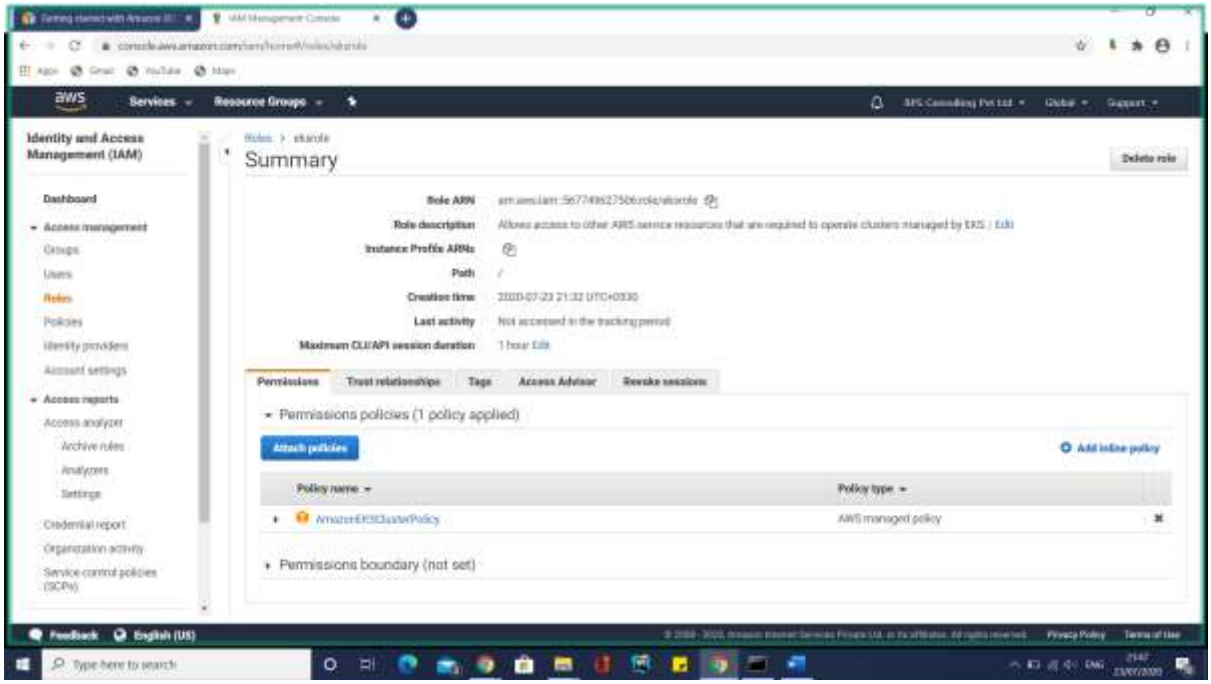
Open the IAM console, select **Roles** on the left and then click the **Create Role** button at the top of the page.

From the list of AWS services, select **EKS** and then **Next: Permissions** at the bottom of the page.




Services
Resource Groups
EC2 - Fleet
EC2 Auto Scaling
EC2 Image Builder
EKS
EMR
Inspector
IoT
IoT SiteWise
IoT Things Graph
KMS
OLDB
RAM
RDS
Redshift
Rekognition
Trusted Advisor
VPC
WorkLink
WorkMail
CloudTrail
CloudWatch Application Insights
CloudWatch Events
CodeBuild
CodeDeploy
Select your use case
EKS
Allows EKS to manage clusters on your behalf.
EKS - Cluster
Allows access to other AWS service resources that are required to operate clusters managed by EKS.
EKS - Fargate pod
Allows access to other AWS service resources that are required to run Amazon EKS pods on AWS Fargate.
EKS - Fargate profile
Allows EKS to run Fargate tasks.
EKS - Nodegroup
Allow EKS to manage nodegroups on your behalf.
* Required
Cancel
Next: Permissions
Feedback
English (US)
Type here to search


Services
Resource Groups
Identity and Access Management (IAM)
Dashboard
Access management
Groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity
Service control policies (SCPs)
Create role
Delete role
Showing 14 results
Role name
Trusted entities
Last activity
AWSServiceRoleForAmazonEKS
AWS service: eks (Service-Linked role)
56 days
AWSServiceRoleForAmazonEKSForFargate
AWS service: eks-fargate (Service-Linked role)
56 days
AWSServiceRoleForAmazonEKSNodegroup
AWS service: eks-nodegroup (Service-Linked r...
56 days
AWSServiceRoleForAmazonElasticFileSystem
AWS service: elasticfilesystem (Service Linke...
80 days
AWSServiceRoleForApplicationAutoScaling_Dynamo...
AWS service: dynamodb.application-autoscali...
None
AWSServiceRoleForAutoScaling
AWS service: autoscaling (Service-Linked role)
Today
AWSServiceRoleForAWSCloud9
AWS service: cloud9 (Service-Linked role)
None
AWSServiceRoleForECS
AWS service: ecs (Service-Linked role)
None
AWSServiceRoleForElasticLoadBalancing
AWS service: elasticloadbalancing (Service-Li...
56 days
AWSServiceRoleForOrganizations
AWS service: organizations (Service-Linked rol...
None
AWSServiceRoleForSupport
AWS service: support (Service-Linked role)
157 days
AWSServiceRoleForTrustedAdvisor
AWS service: trustedadvisor (Service-Linked r...
None
eksctl-eksdemo-01-cluster-ServiceRole-26ZQRAEAX...
AWS service: eks-fargate-pods and 1 more
56 days
eksrole
AWS service: eks
None
Feedback
English (US)
Type here to search



Role ARN

arn:aws:iam::567749627506:role/eksrole

arn:aws:iam::256544469827:role/eksrole

Step 2: Creating a VPC for EKS

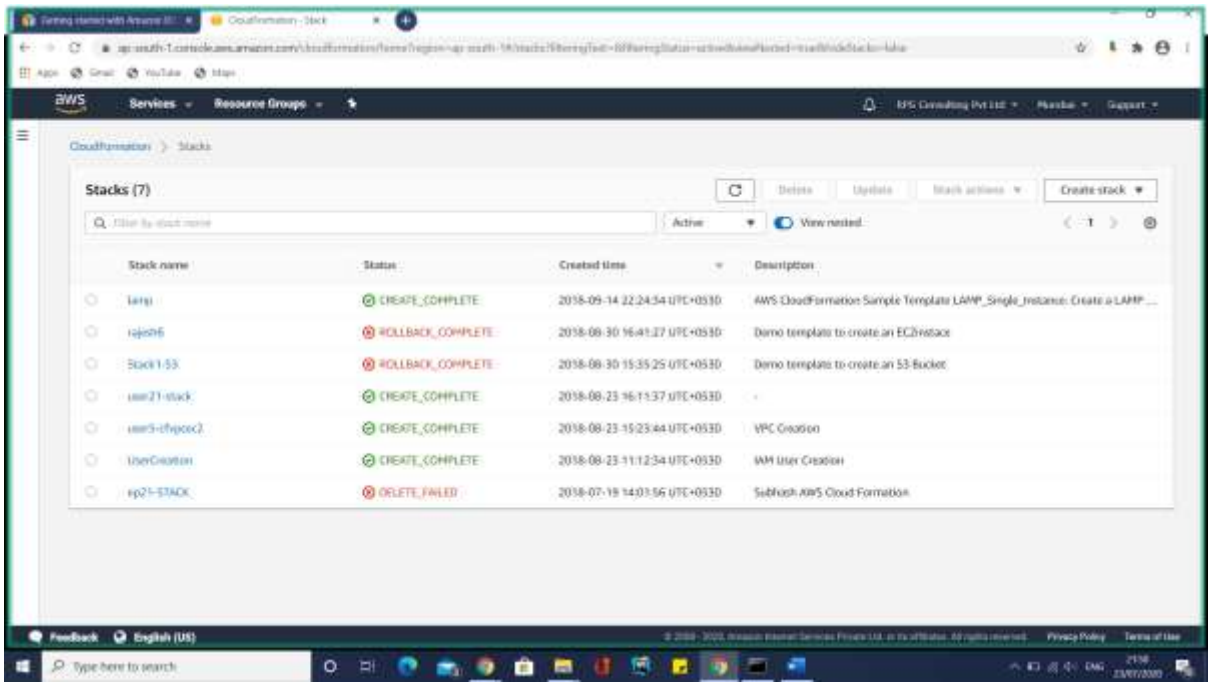
Next, we're going to create a separate VPC—a Virtual Private Cloud that protects communication between worker nodes and the AWS Kubernetes API server— for our EKS cluster. To do this, we're going to use a CloudFormation template that contains all the necessary EKS-specific ingredients for setting up the VPC.

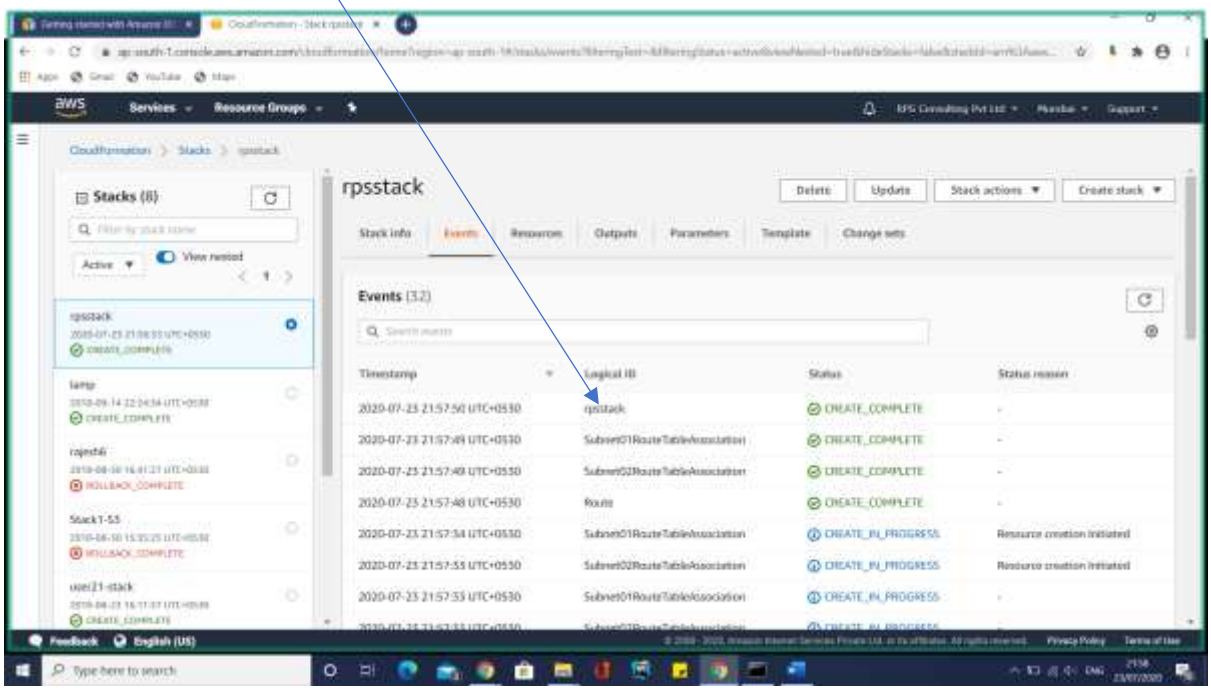
Open up CloudFormation, and click the **Create new stack** button.

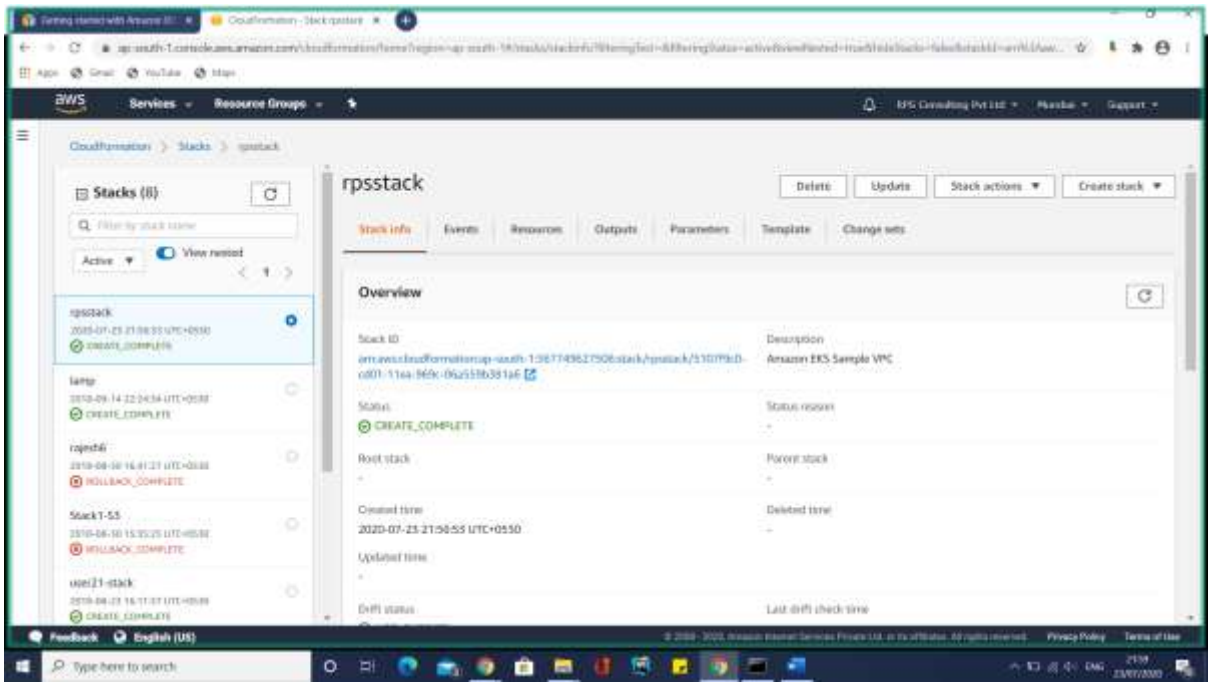
On the **Select template** page, enter the URL of the CloudFormation YAML in the relevant section:

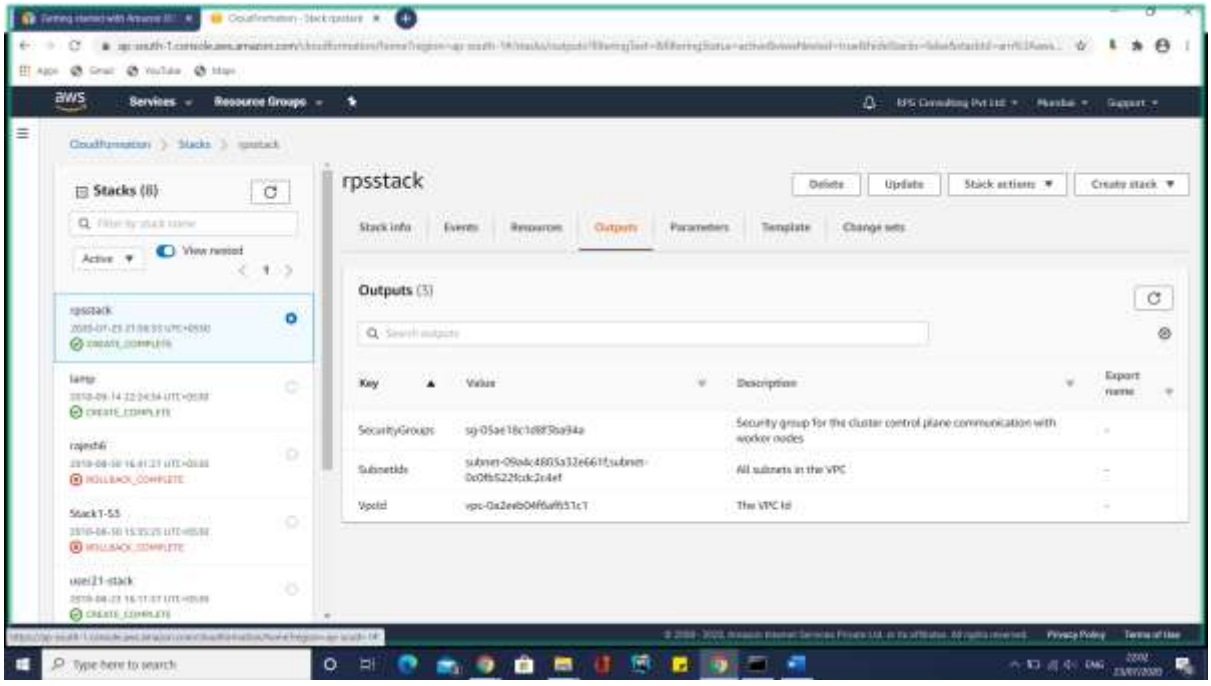
https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2019-01-09/amazon-eks-vpc-sample.yaml

Search in Service Cloud Formation





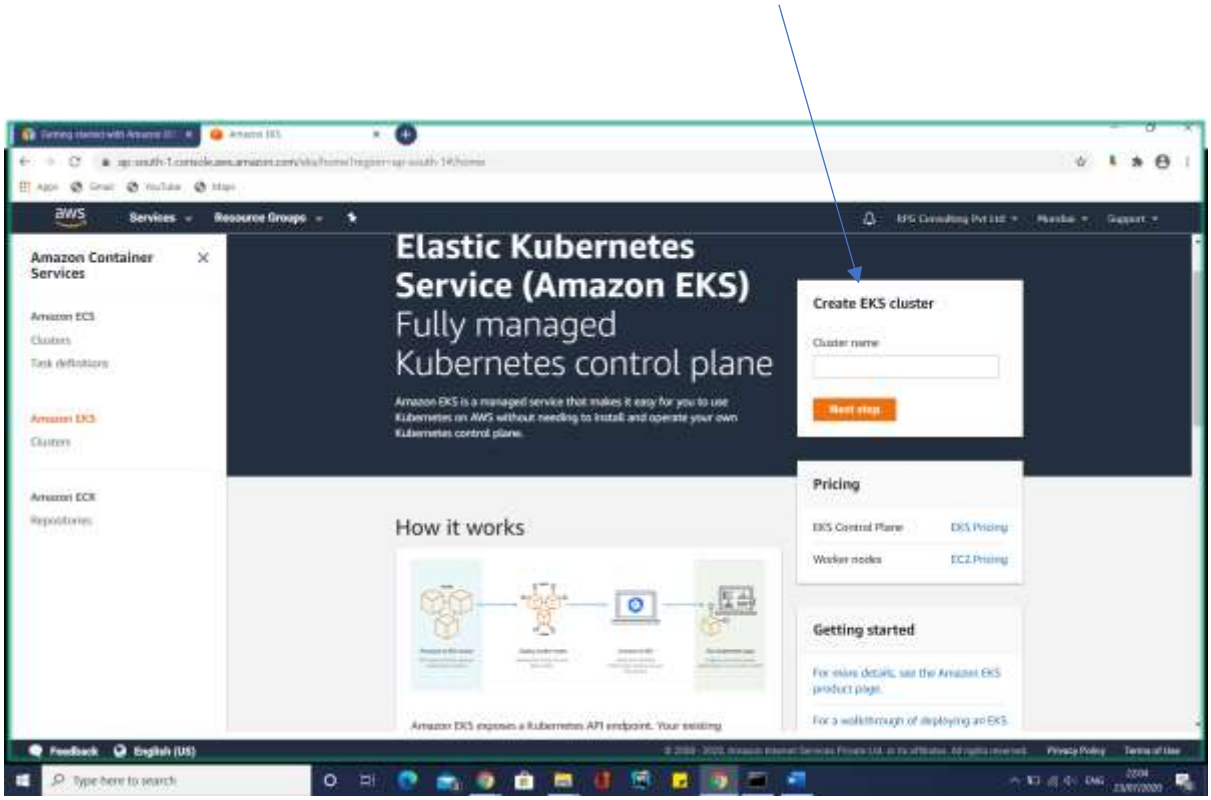

aws
Services
Resource Groups
CloudFormation > Stacks > rpsstack
Stacks (8)
rpsstack
Delete
Update
Stack actions
Create stack
Stack info
Events
Resources
Outputs
Parameters
Template
Change sets
Overview
Stack ID
Description
Amazon EKS Sample VPC
Status
CREATE_COMPLETE
Root stack
Parent stack
Created time
2020-07-23 21:58:55 UTC+0550
Deleted time
Updated time
Drift status
Last drift check time
Feedback
English (US)
Type here to search


aws
Services
Resource Groups
CloudFormation > Stacks > rpsstack
Stacks (8)
rpsstack
Delete
Update
Stack actions
Create stack
Stack info
Events
Resources
Outputs
Parameters
Template
Change sets
Outputs (3)
Key
Value
Description
Export name
SecurityGroups
sg-05ae18c1d8f3ba94a
Security group for the cluster control plane communication with worker nodes
SubnetIds
subnet-09a4c4803a32e661f,subnet-0e0fb522fcdc2c4ef
All subnets in the VPC
VpcId
vpc-0a2eeb04f6af851c1
The VPC Id
Type here to search

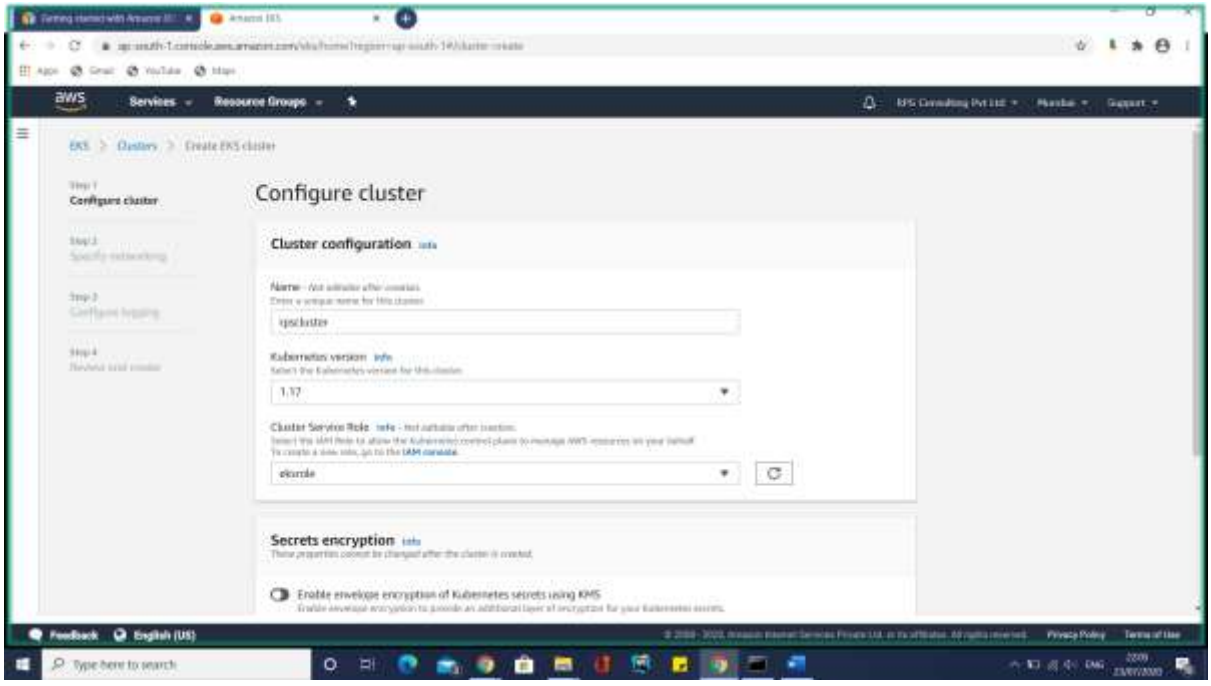
SecurityGroups	sg-05ae18c1d8f3ba94a	Security group for the cluster contr
SubnetIds	subnet-09a4c4803a32e661f,subnet-0c0fb522fcdc2c4ef	All subnets in the VPC
VpcId	vpc-0a2eeb04f6af651c1	The VPC Id

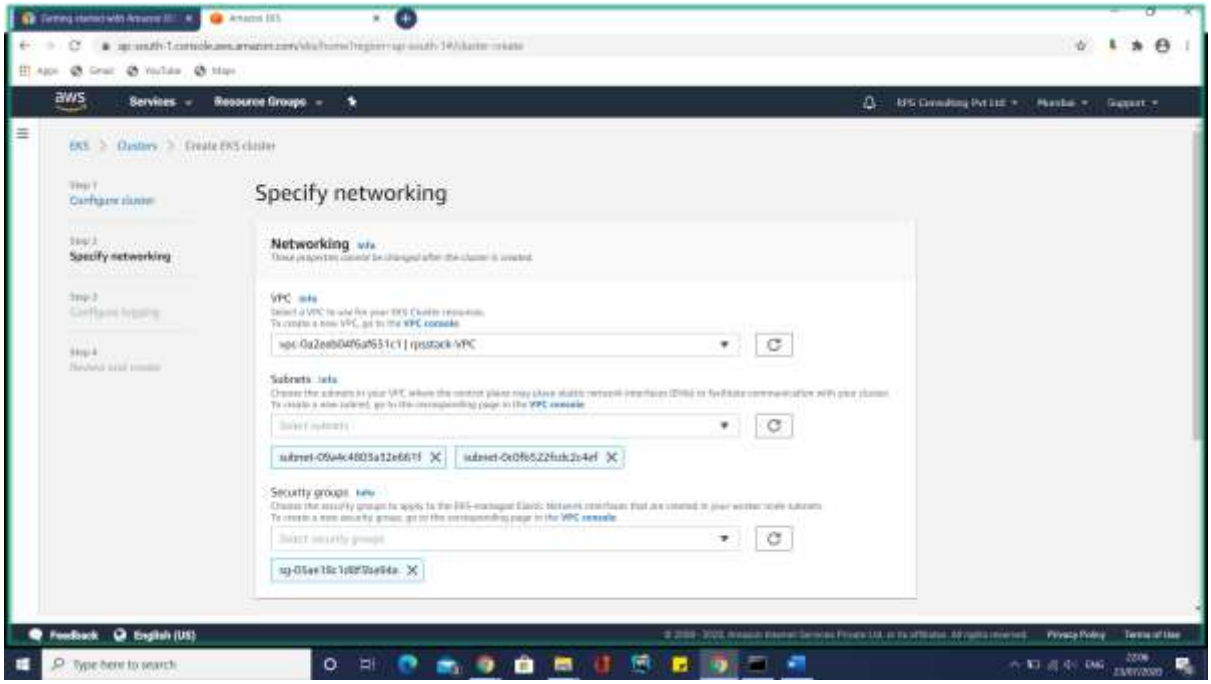
Step 3: Creating the EKS cluster

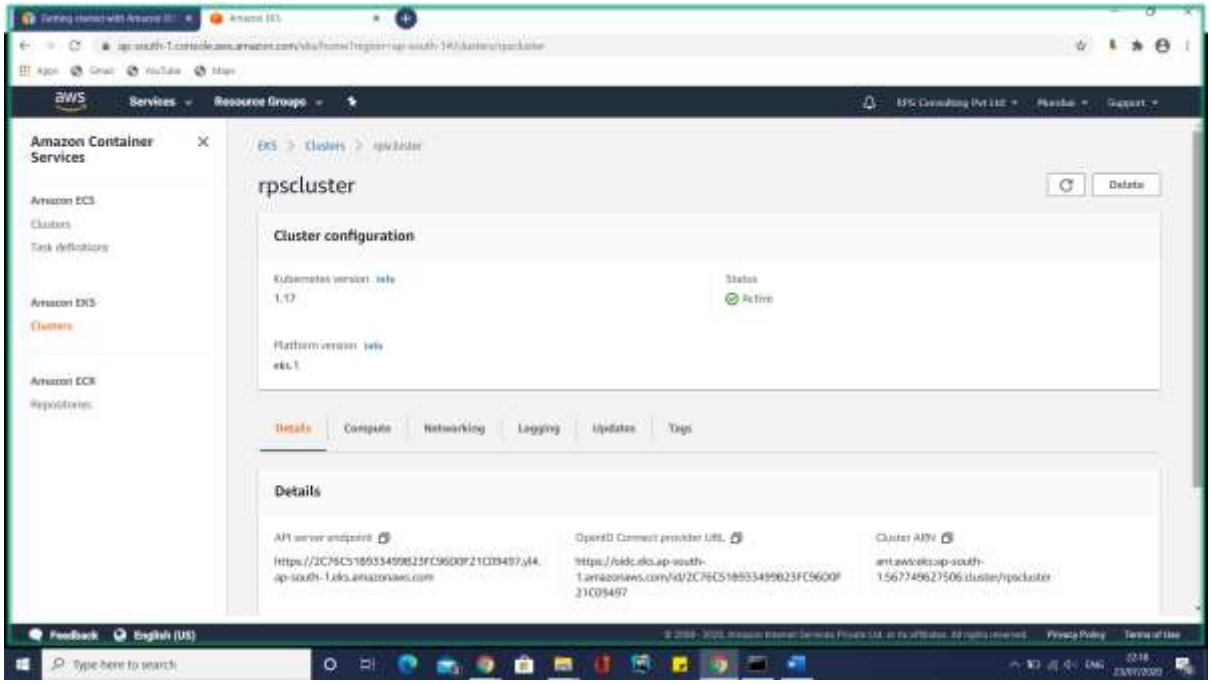
Navigate to EKS cluster

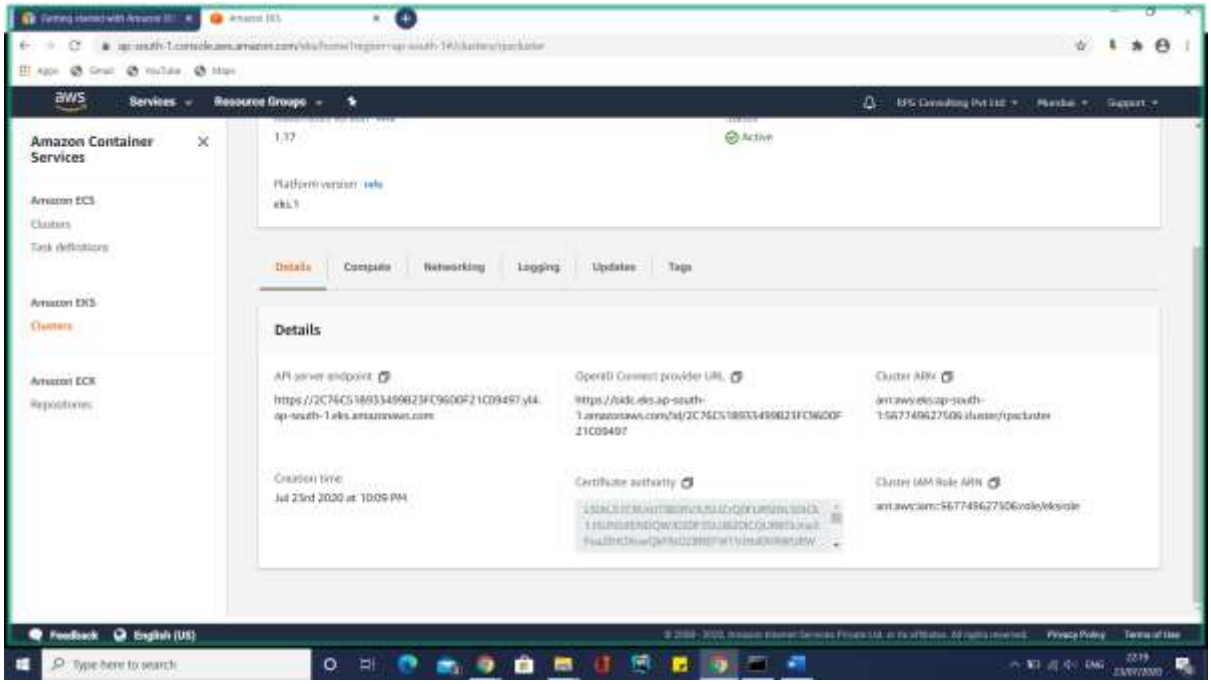


Configure Cluster




Specify networking
Networking
VPC
Subnets
Security groups


Amazon Container Services
rpscluster
Cluster configuration
Details



API server End point

https://2C76C51B933499B23FC96D0F21C09497.yl4.ap-south-1.eks.amazonaws.com

Open Id Connect URL

https://oidc.eks.ap-south-1.amazonaws.com/id/2C76C51B933499B23FC96D0F21C09497

Certificate Authority

LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN5RENDQWJDZ0F3SUJBZ0lCQURBTkJna3Foa2lHOXcwQkFRc0ZBREFWTVJNd0V
RWURWUVFERXdwcmRXSmwKY201bGRHVnpNQjRYRFRJd01EY3lNekUyTkRZd01sb1hEVE13TURjeU1URTJORFl3TWxvd0ZURVRNQk
VHQTFVRQpBeE1LYTNWaVpYSnVaWFJsY3pDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBSy9aCkN2d
TRCTThIRS9iekZiYnZLTlpIeDNXVnJGUXN0cTRDd3BuanlsU1ExekNsZWZhbTl4cmpCY1k5OUZ1K2JKQ2wKdVp3QjJxeXV4Ymd0d00ycVR
PUVorUmZoR3V2ZGVYMGcyNWtVMmRBMjU2akZWWWYxNjV2Mm50YWVleS9Vd1N6dQo3U3VpZ3RsZExhb3ZsbHRYbEJPaEtUN1dtRH
dHM1RKSnZESEJFaWlJWk8zdW5HRHdIRXZKQTZiaG1Rd2FEbTZuClB3UDVwZjZKdUhtNk9ZRE1EOFBKQmV6T2UrZmUwWDFUbW5tV
ExUd2tEY3ZCU1lnMVN5MndCcU1xR0tjVS9xYzIKWXRkM0diZ0dFT2hMNHZ6RUpHSXdkRmFHZmcrVEhsWEhBbWdCVnZrQloyOUY2M
W1IaFZVV3d2ZDg1bzQvRUVBaQpTb2JJZGpjV0RXbXg4UGwzb0trQ0F3RUFBYU1qTUNFd0RnWURWUjBQQVFIL0JBUURBZ0trTUE4R0E
xVWRFd0VCCi93UUZNQU1CQWY4d0RRWUpLb1pJaHZjTkFRUxCUUFEZ2dFQkFFWWlnMlBiUlJQK0M2S05GeUdzM05sUDFqeGgKbzF
XWWxRYWEvT25CbUZ6OWtMU05HTml3VDdFWnptS09RUDVteVZiUnljN3JqMFpja0NLTTBQbk1qTGJmK0tHZwpOWHkwNHcwOGpoTm
ZoNFdvOTJMc0ltWlFKSkZvUEd3dG9MQ3AxWDRlK0xHYmo3THM2aXpSV0FyVGk0Y2dQbTlIClZhb1pQMk9IRndHSjc1RlZ2Q2dJSUVweV
Z0bStHUVNNR1Qza01MVm1Tdml6dnlPcGRFMnlhYU4wYzgvUzExbTMKd1ozc20vQ3JybjNWNkg1VVdsTEcxNjFTSy90bFE2NjVucklEdjFTZ

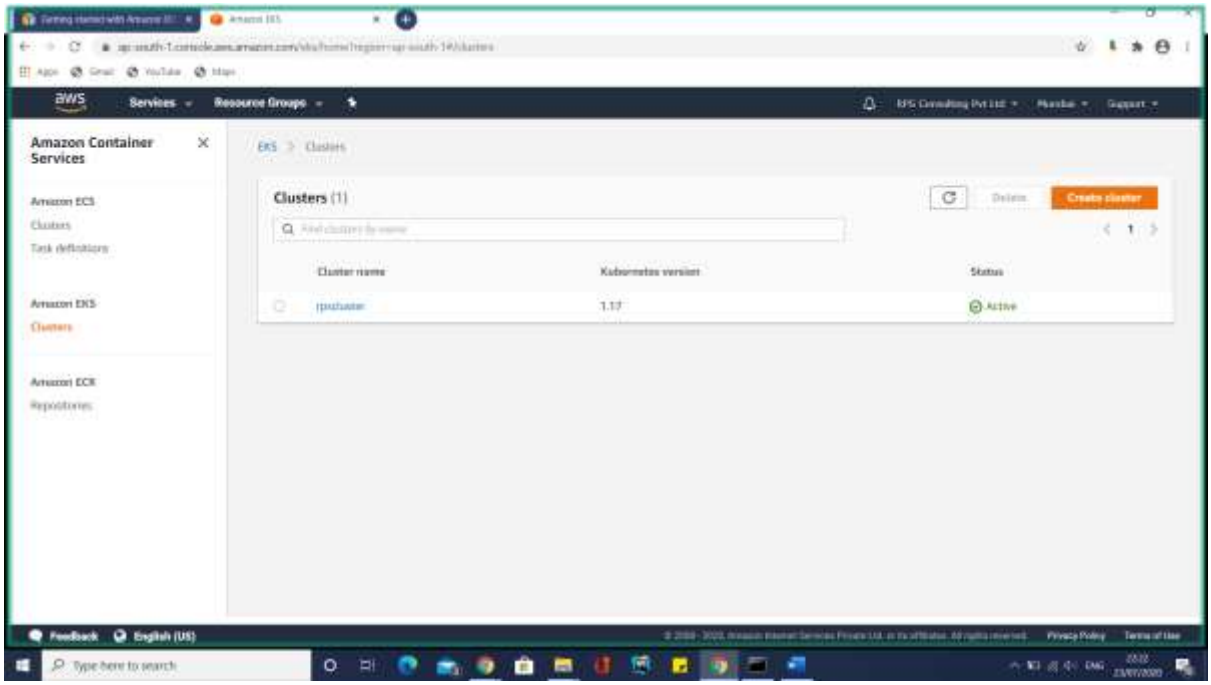
EJOTHpqckdoL0V0OHB6dmxERU84UUFnMwpmNDBNSUZ0TVN0emxjRk1PVndwTXAzV0M2VE1uQ3NudEM2VjVtYlM3QUxRQkZxSmw3N3BvR1oyWTNORT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo=

Cluster ARN

arn:aws:eks:ap-south-1:567749627506:cluster/rpscluster

Iam Role

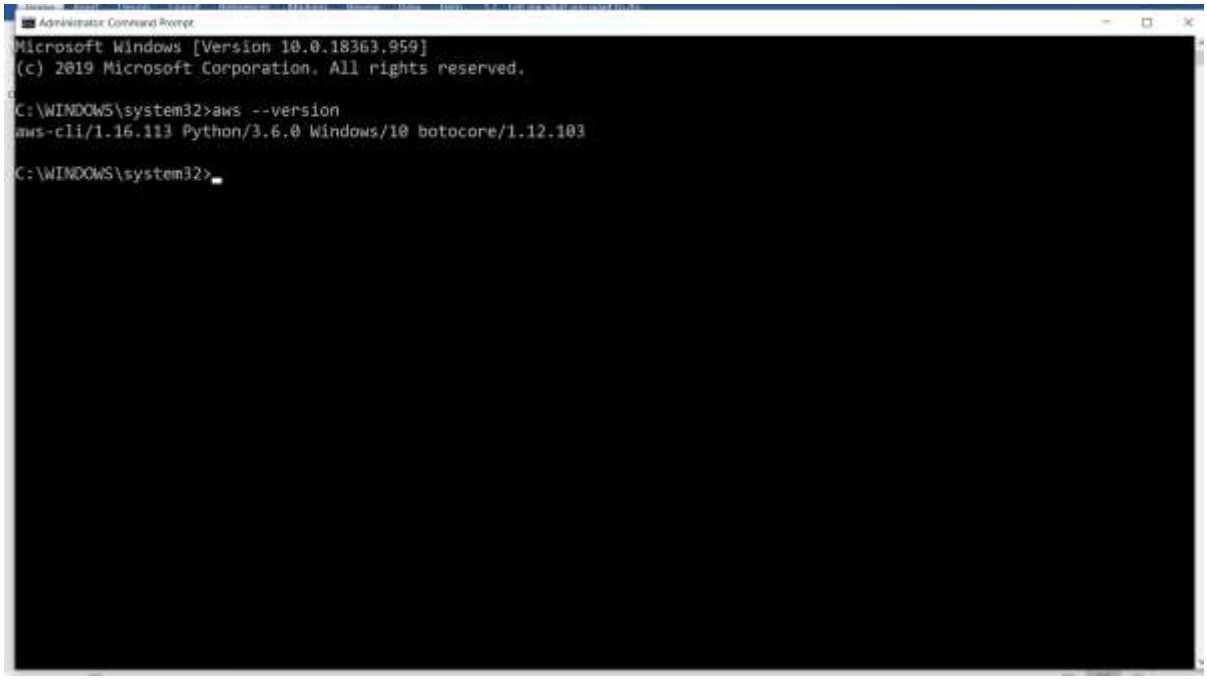
arn:aws:iam::567749627506:role/eksrole



To install the AWS CLI version 2

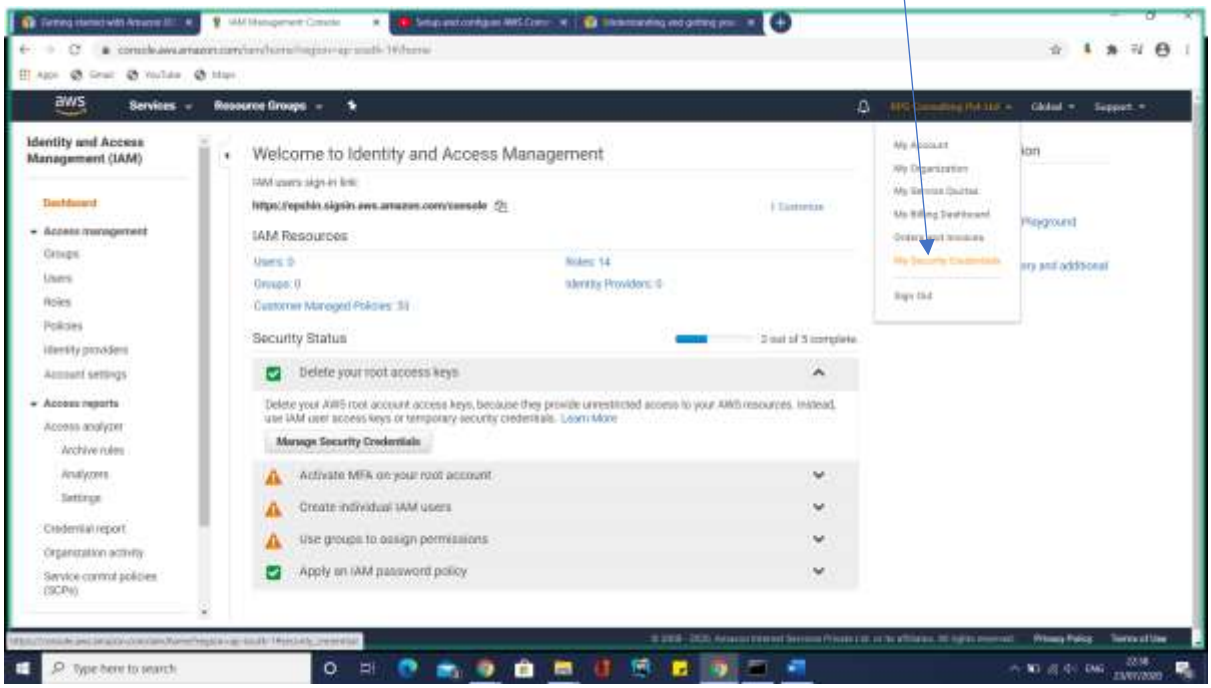
If you don't have either version 1.18.97 or later, or version 2.0.30 or later installed, then install the AWS CLI version 2 using the following steps. For other installation options, or to upgrade your currently installed version 2, see Upgrading the AWS CLI version 2 on Windows.

1. Download the AWS CLI MSI installer for Windows (64-bit) at https://awscli.amazonaws.com/AWSCLIV2.msi
2. Run the downloaded MSI installer and follow the onscreen instructions. By default, the AWS CLI installs to `C:\Program Files\Amazon\AWSCLIV2`.



To create an access key when signed in as the root user

1. Sign in to the AWS Management Console as the root user. For more information, see Sign in as the root user in the *IAM User Guide*.
2. In the navigation bar on the upper right, choose your account name or number and then choose **My Security Credentials**.
3. Expand the **Access keys (access key ID and secret access key)** section.
4. Choose **Create New Access Key**. If you already have two access keys, this button is disabled.
5. When prompted, choose **Show Access Key** or **Download Key File**. This is your only opportunity to save your secret access key.
6. After you've saved your secret access key in a secure location, chose **Close**.

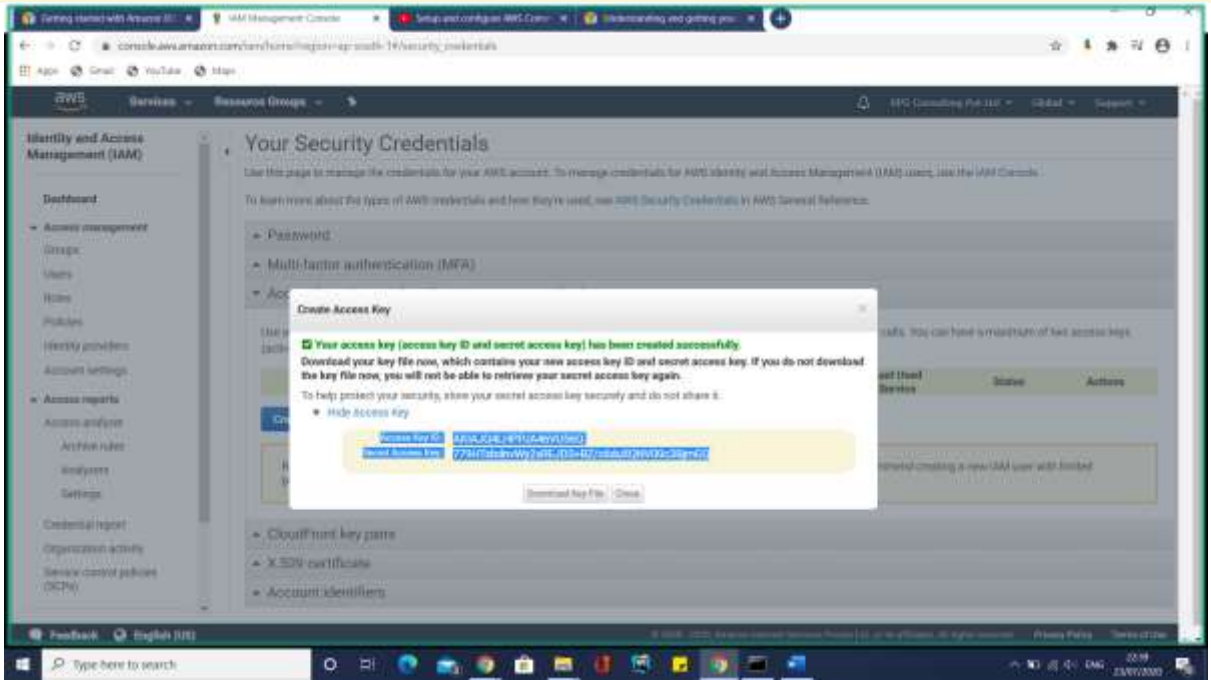


Access Key ID:

AKIAJQ4LHPPJA46VU56Q

Secret Access Key:

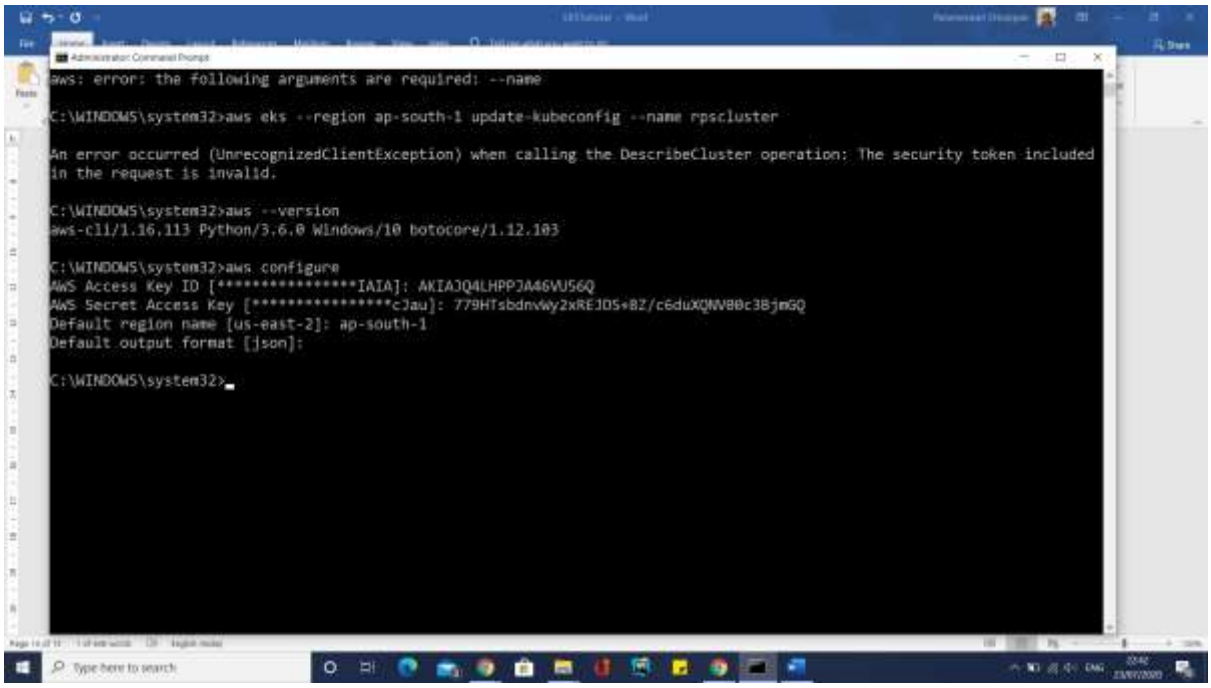
779HTsbdnvWy2xREJDS+BZ/c6duXQNV00c3BjmGQ



To create an access key when signed in as an IAM user

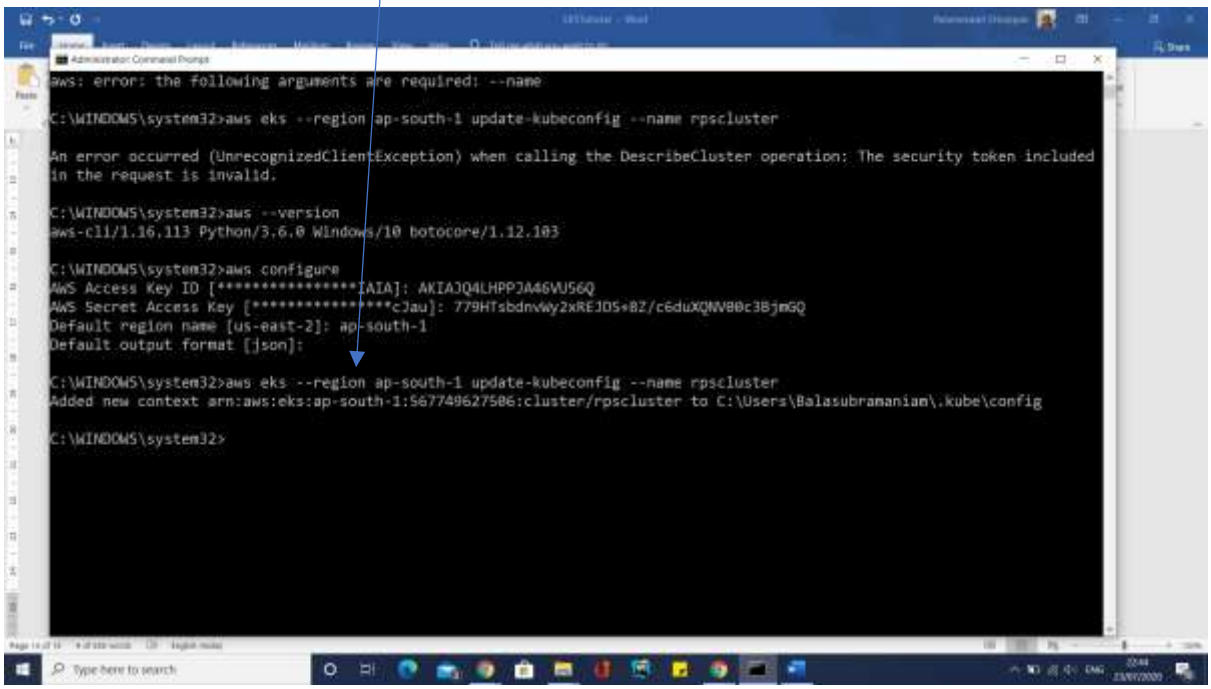
1. Sign in to the AWS Management Console as an IAM user. For more information, see Sign in as an IAM user in the *IAM User Guide*.
2. In the navigation bar on the upper right, choose your user name and then choose **My Security Credentials**.
3. Choose **AWS IAM credentials**, **Create access key**. If you already have two access keys, the console displays a "Limited exceeded" error.
4. When prompted, choose **Download .csv file** or **Show secret access key**. This is your only opportunity to save your secret access key.
5. After you've saved your secret access key in a secure location, chose **Close**.

AWS configure



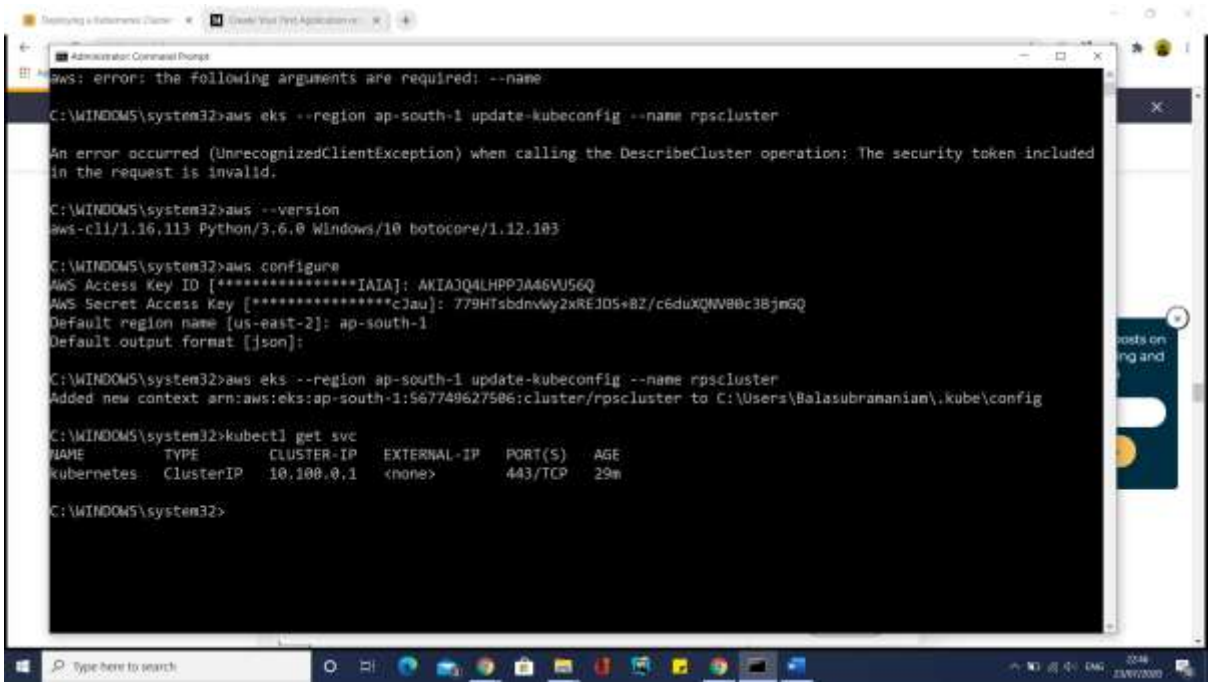
ap-south-1

`aws eks --region` ap-south-1 `update-kubeconfig --name rpscluster`



We can now test our configurations using the kubectl get svc command:

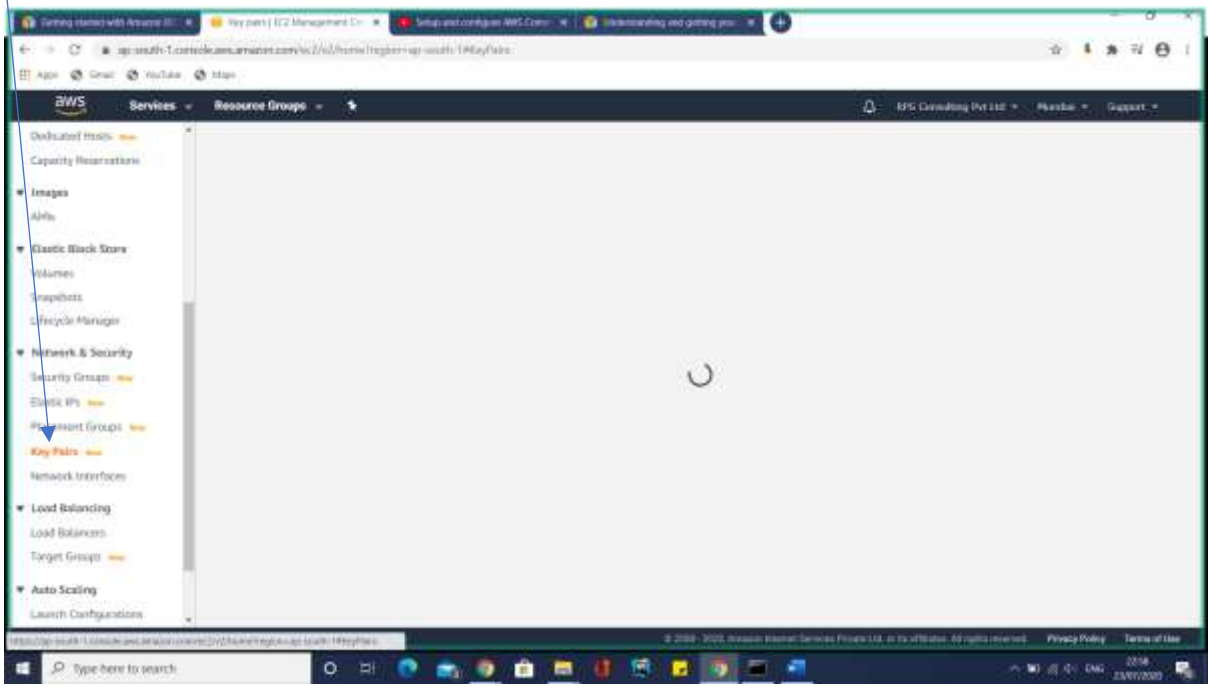
Kubectl get svc

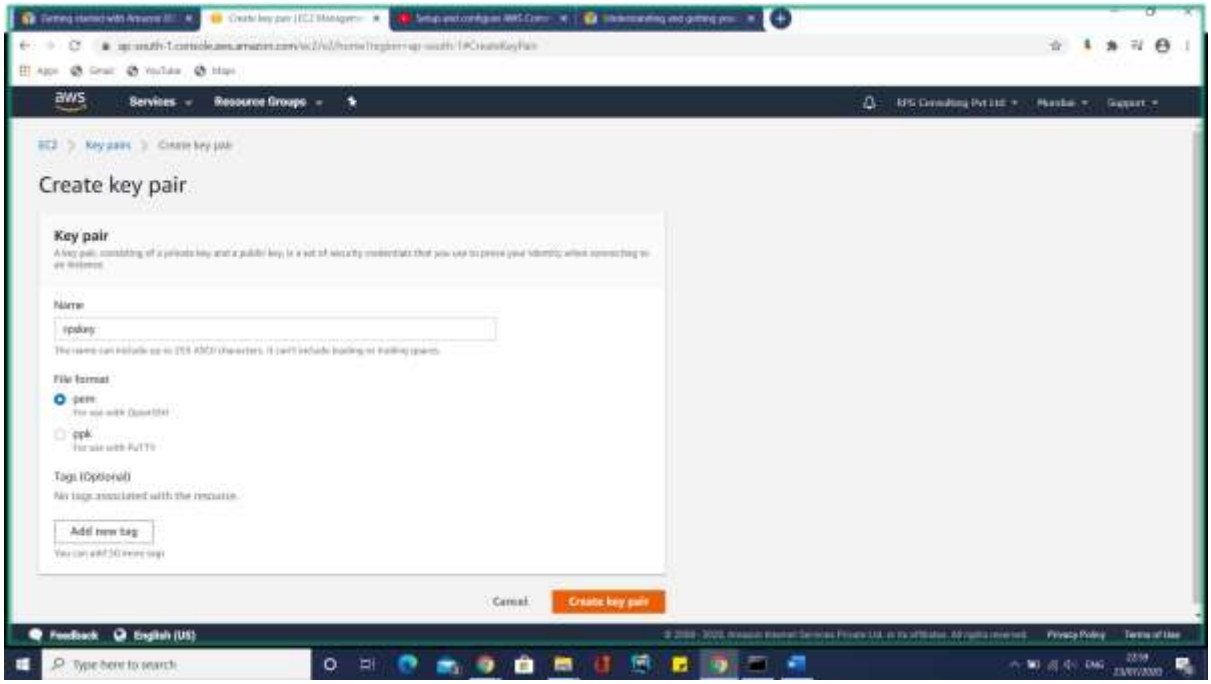


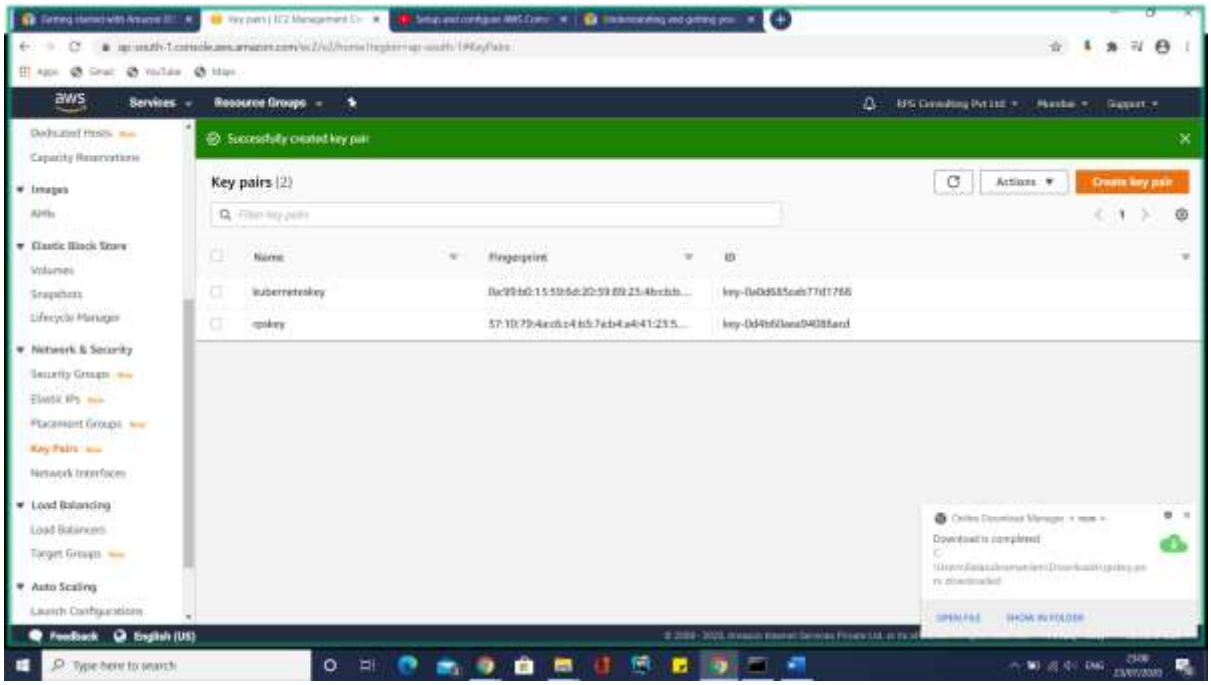
Step 4

Create Key Pair using Console

- Login to AWS Management Console and choose an AWS Region to create a key pair.
- Click on services and find EC2 under Compute services.
- Go to Network & Security > Key Pairs, and then choose Create Key Pair.




Create key pair
Key pair
Name
rpskey
File format
pem
ppk
Tags (Optional)
Add new tag
Cancel
Create key pair


Successfully created key pair
Key pairs (2)
Actions
Create key pair
Name
Fingerprint
ID
kubernetskey
rpskey

Step 5: Launching Kubernetes worker nodes

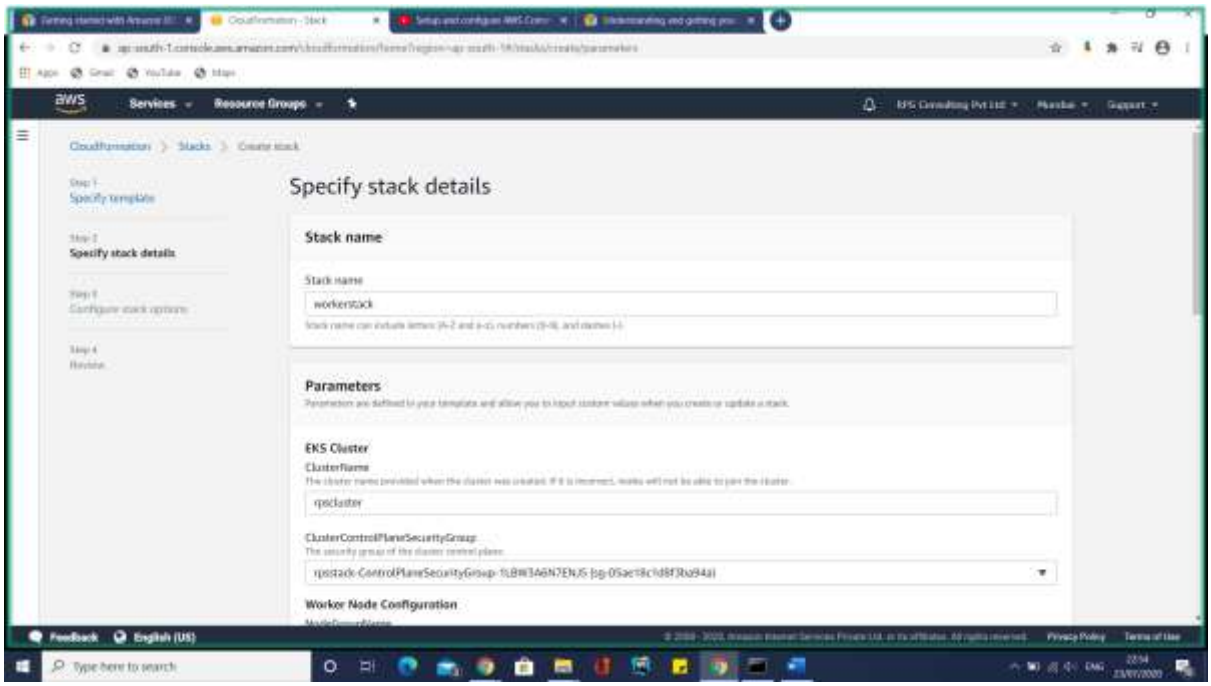
Now that we've set up our cluster and VPC networking, we can now launch Kubernetes worker nodes. To do this, we will again use a CloudFormation template.

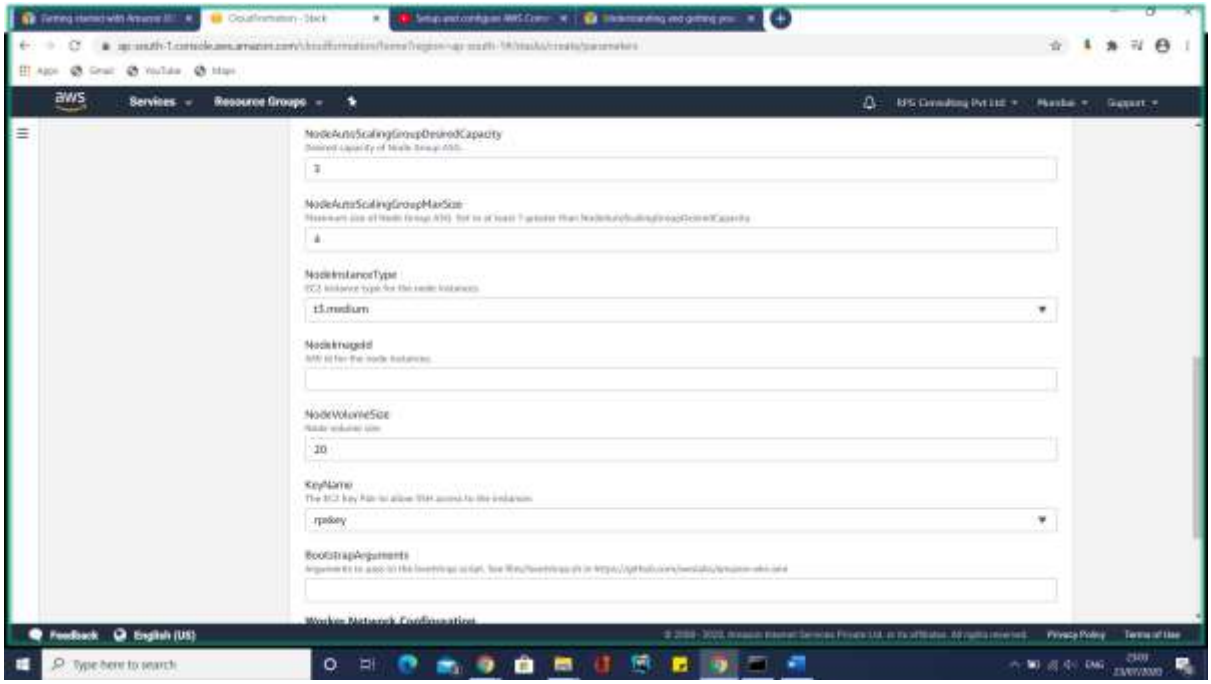
Open CloudFormation, click **Create Stack**, and this time use the following template URL:

https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2019-01-09/amazon-eks-nodegroup.yaml

- **ClusterName** – the name of your Kubernetes cluster (e.g. demo)
- **ClusterControlPlaneSecurityGroup** – the same security group you used for creating the cluster in previous step.
- **NodeGroupName** – a name for your node group.
- **NodeAutoScalingGroupMinSize** – leave as-is. The minimum number of nodes that your worker node Auto Scaling group can scale to.
- **NodeAutoScalingGroupDesiredCapacity** – leave as-is. The desired number of nodes to scale to when your stack is created.
- **NodeAutoScalingGroupMaxSize** – leave as-is. The maximum number of nodes that your worker node Auto Scaling group can scale out to.

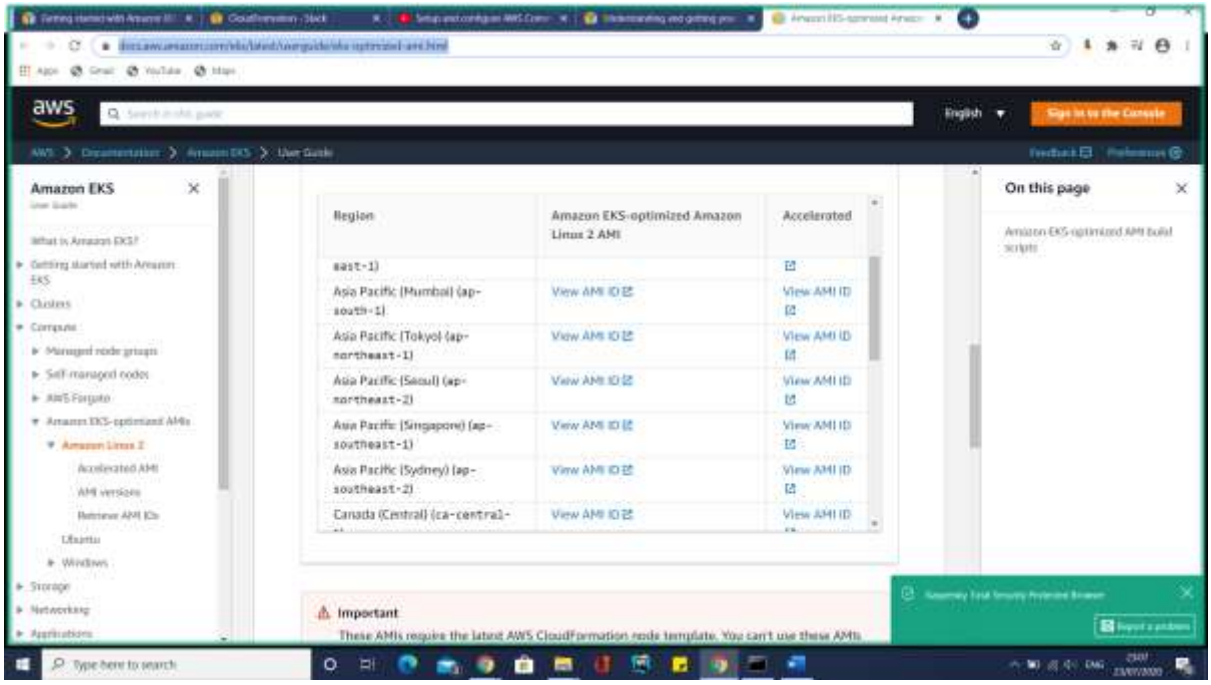
- **NodeInstanceType** – leave as-is. The instance type used for the worker nodes.
- **NodeImageId** – the Amazon EKS worker node AMI ID for the region you're using. For us-east-1, for example: ami-0c5b63ec54dd3fc38
- **KeyName** – the name of an Amazon EC2 SSH key pair for connecting with the worker nodes once they launch.
- **BootstrapArguments** – leave empty. This field can be used to pass optional arguments to the worker nodes bootstrap script.
- **VpcId** – enter the ID of the VPC you created in Step 2 above.
- **Subnets** – select the three subnets you created in Step 2 above.





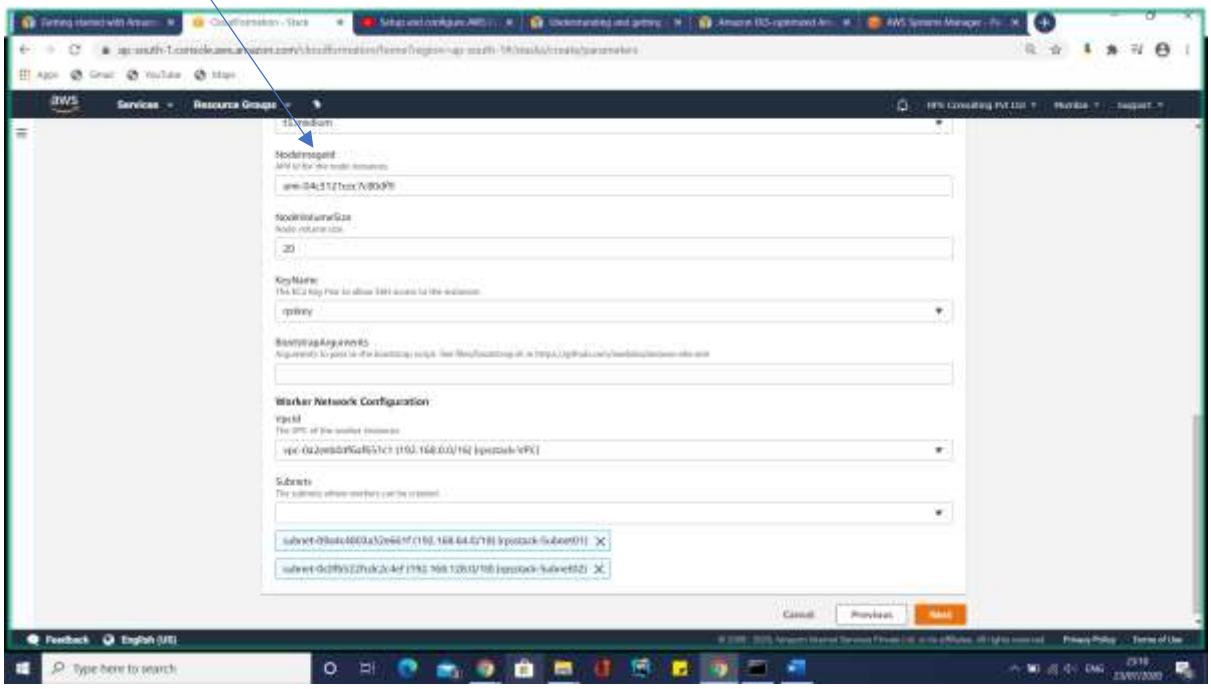
Go to the link to view AMI Image Id

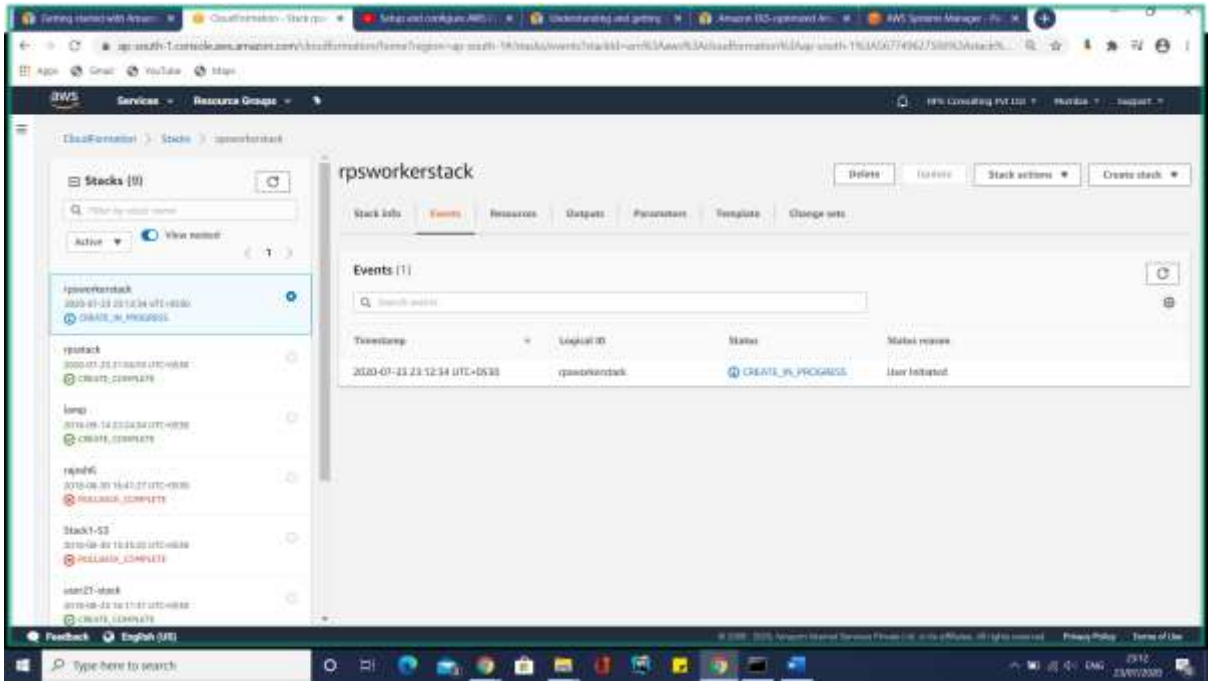
https://docs.aws.amazon.com/eks/latest/userguide/eks-optimized-ami.html

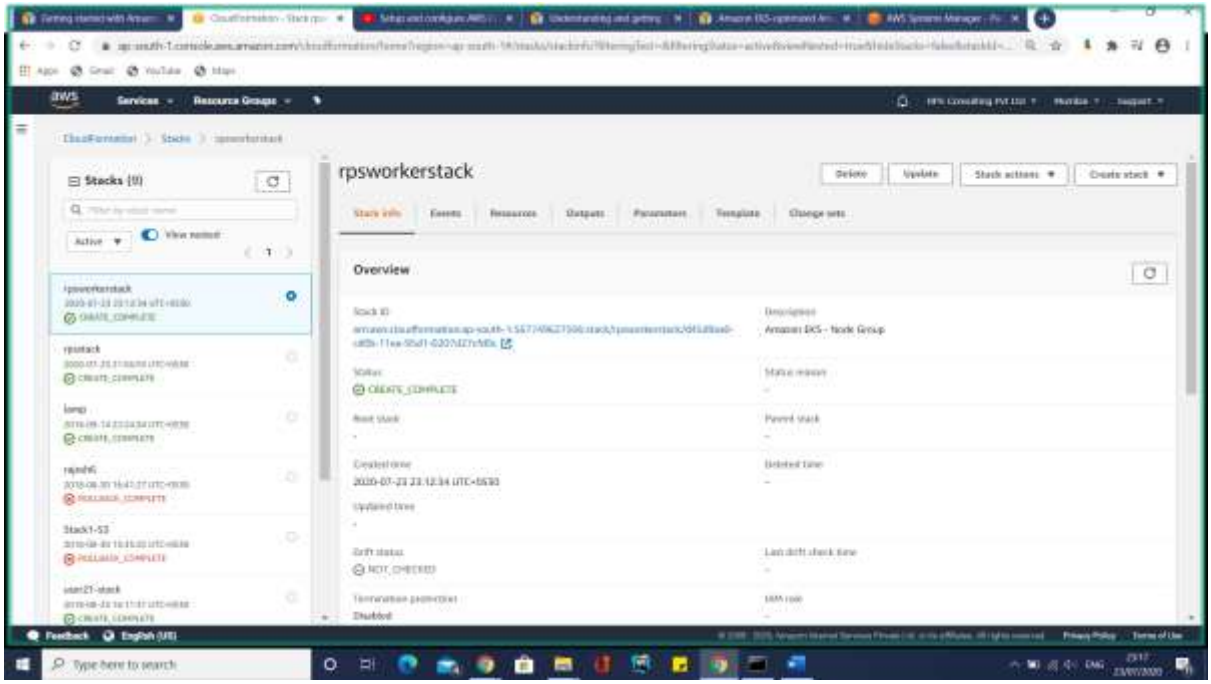


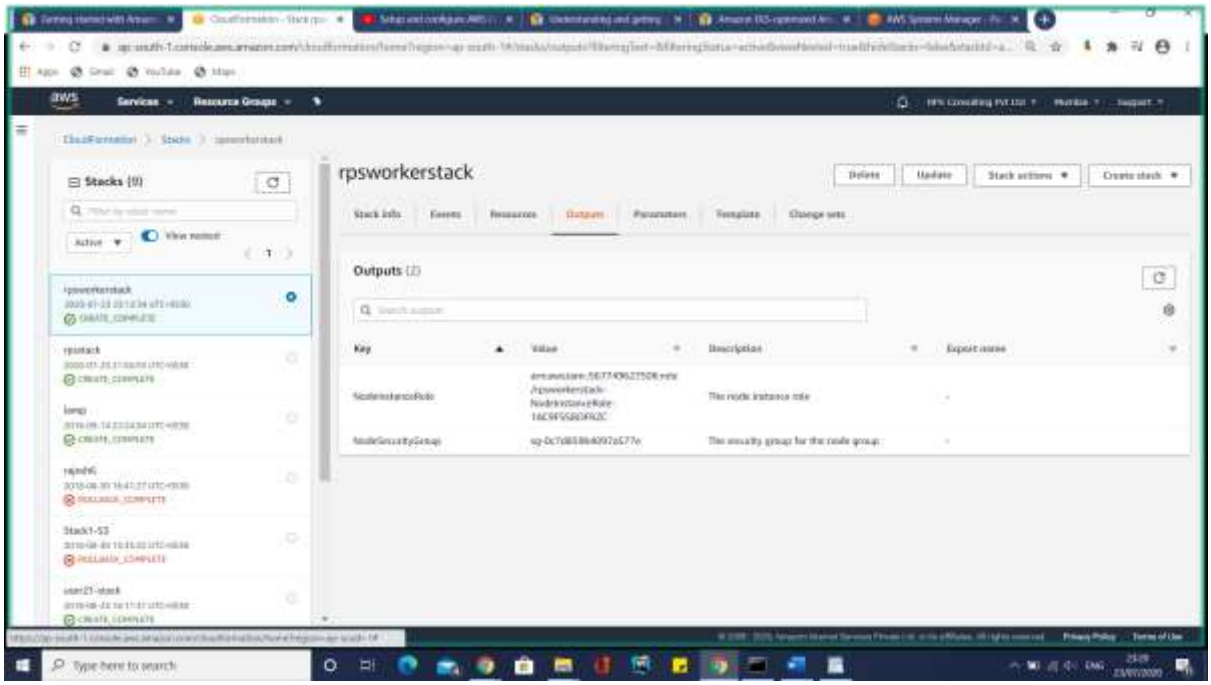
AMI ID for asia pacific south

ami-04c3121cec7c80df9









NodeInstanceRole

arn:aws:iam::567749627506:role/rpsworkerstack-NodeInstanceRole-1AC9FSSBOFRZC

NodeSecurityGroup

sg-0c7d859b4097a577e

Download

```
https://amazon-eks.s3-us-west-2.amazonaws.com/cloudformation/2019-01-09
/aws-auth-cm.yaml
```

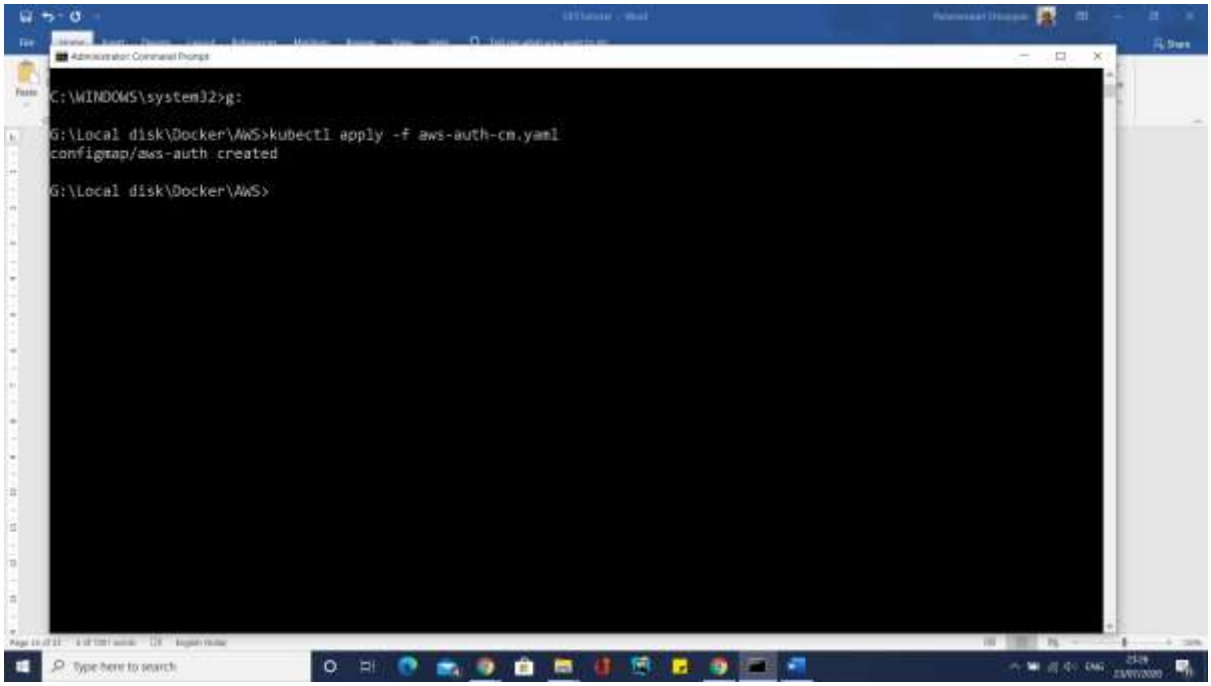
Open the yml fie

Open the file and replace the rolearn with the ARN of the *NodeInstanceRole* created above:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - rolearn: <ARN of instance role>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

Go to g:\local disk\docker\aws

kubectl apply -f aws-auth-cm.yaml

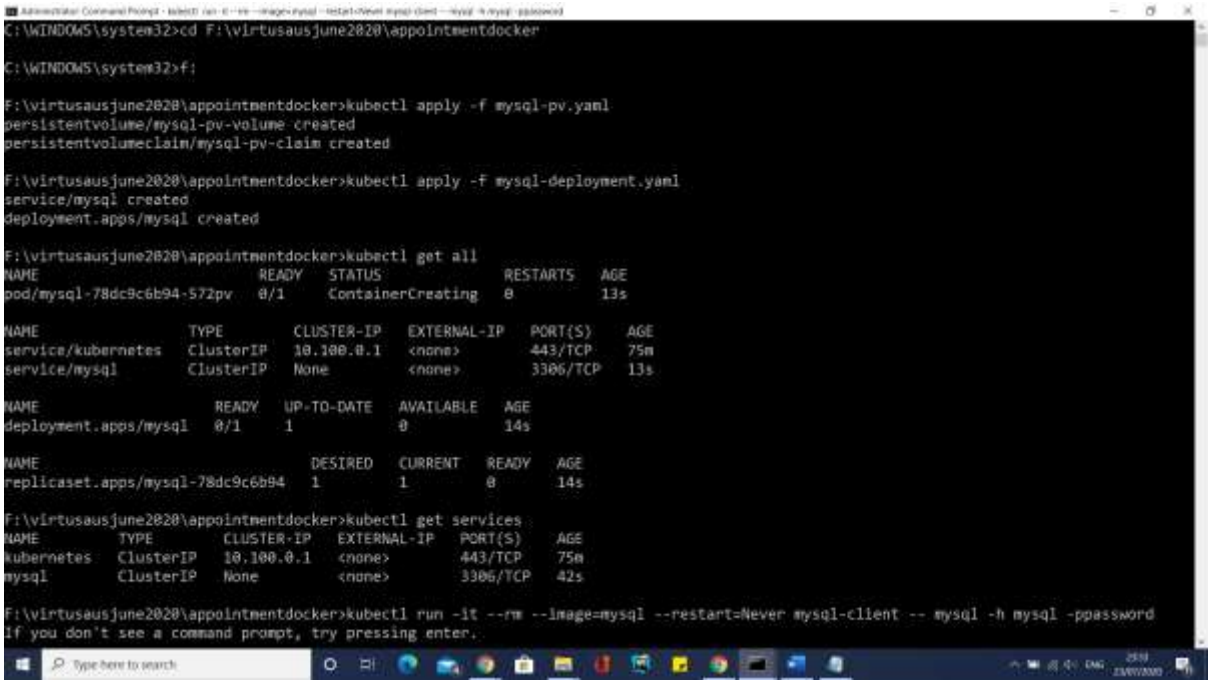


Use kubectl to check on the status of your worker nodes:

```
kubectl get nodes –watch
```

kubectl get nodes

Step 6: Installing a demo app on Kubernetes



goto apppointmenntdocker project in f:\virtusausjune

kubectl apply -f mysql-pv.yaml

kubectl apply -f mysql-deployment.yaml

kubectl get all

kubectl get services

kubectl describe deployment mysql

kubectl get pods -l app=mysql

kubectl describe pvc mysql-pv-claim

kubectl logs mysql-78dc9c6b94-vfgtq

kubectl get services

get ipaddress

kubectl run -it --rm --image=mysql --restart=Never mysql-client -- mysql -h mysql -ppassword

(don't change password it's ppassword)

create database virtusausappointmentdb;

show databases;

==========

Creating the secrets

kubectl create secret generic mysql-root-pass --from-literal=password=password

kubectl create secret generic mysql-user-pass --from-literal=username=demo_user --from-literal=password=demo_pass

kubectl create secret generic mysql-db-url --from-literal=database=virtusausappointmentdb --from-literal=url='jdbc:mysql://mysql:3306/virtusausappointmentdb?useSSL=false

kubectl get secrets

kubectl describe secrets mysql-user-pass

kubectl get persistentvolumes

==========

kubectl create deployment demo --image=appointmentapp/v1 --dry-run -o=yaml > deployment.yaml

---

kubectl create service clusterip demo --tcp=8070:8070 --dry-run -o=yaml >> deployment.yaml

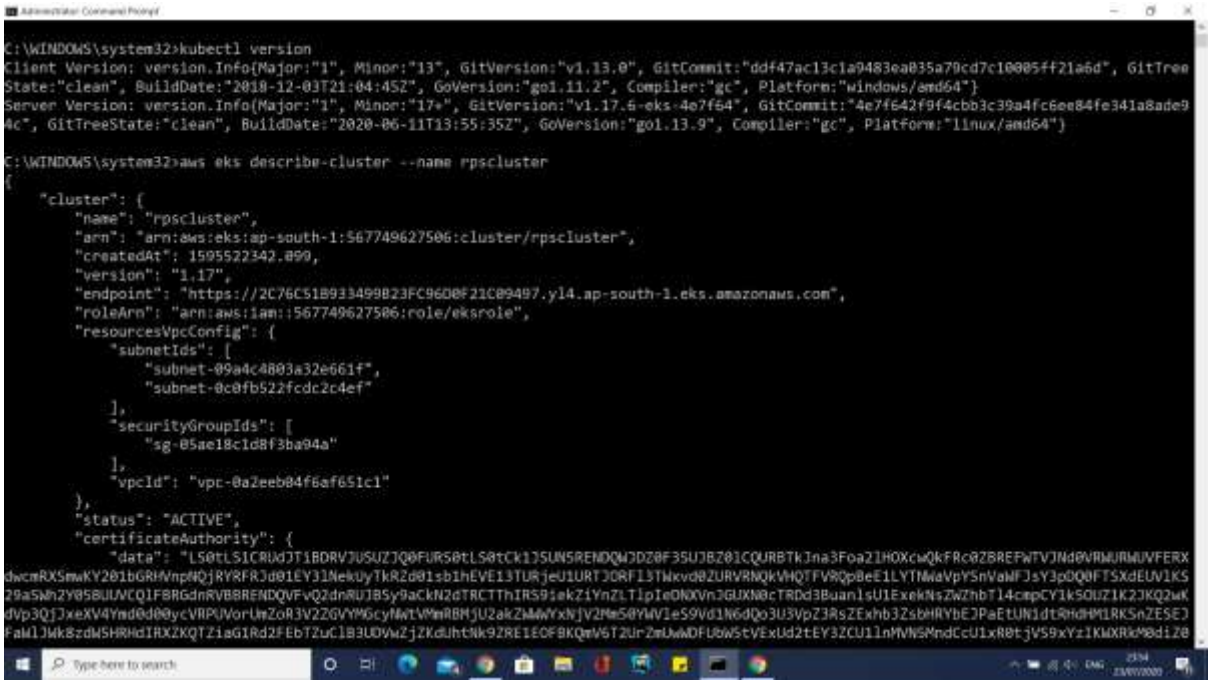
==========

kubectl apply -f deployment_v2.yaml

kubectl get pods

kubectl get services

kubectl port-forward service/demo 8070:8070



aws eks describe-cluster -–name rpscluster

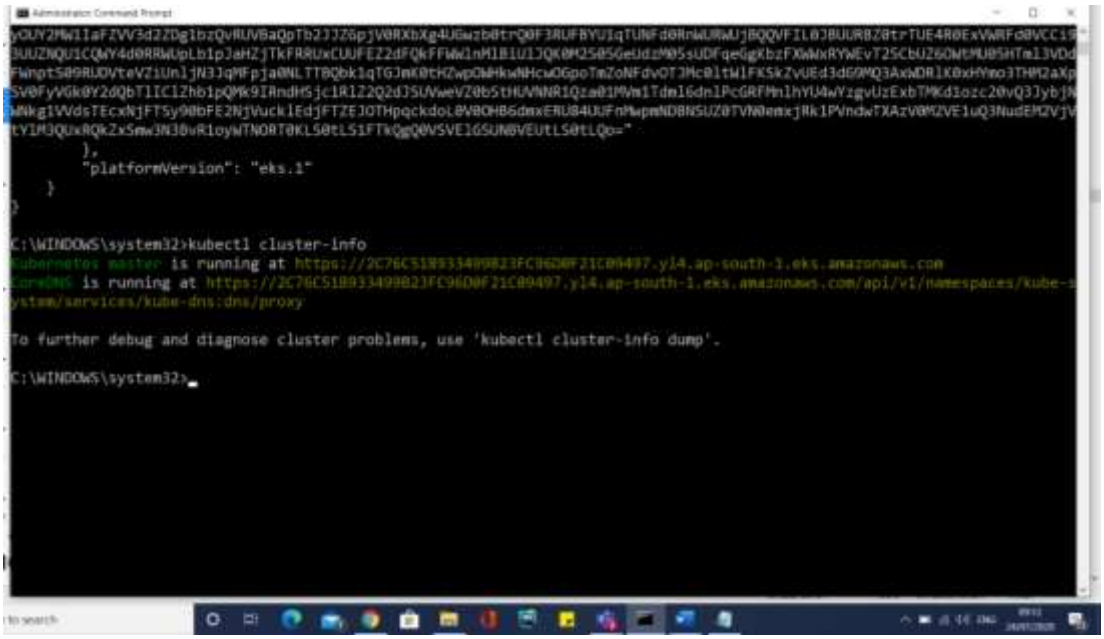
CLI commands

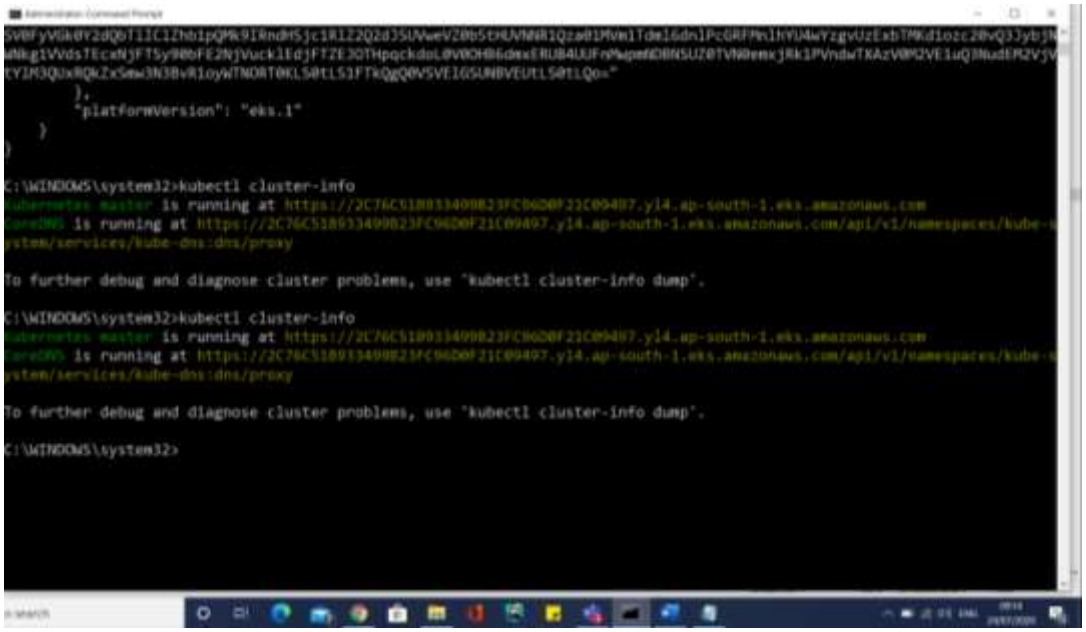
To create Cluster

```
aws eks create-cluster --name prod \

--role-arn arn:aws:iam::012345678910:role/eks-service-role-AWSServiceRoleForAmazonEKS-J7ONKE3BQ4PI \

--resources-vpc-config subnetIds=subnet-6782e71e,subnet-e7e761ac,securityGroupIds=sg-6979fe18
```

To Delete Cluster

```
aws eks delete-cluster --name devel
```





To Analyse cluster dump

'kubectl cluster-info dump

Dashboard

kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/download/v0.3.6/components.yaml

```
kubectl get deployment metrics-server -n kube-system
```

```
kubectl apply -f
https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta8/aio/deploy/recommended.yaml
```

```
kubectl apply -f eks-admin-service-account.yaml
```

```
kubectl -n kube-system describe secret $(kubectl -n kube-system get secret
| grep eks-admin | awk '{print $1}')
```