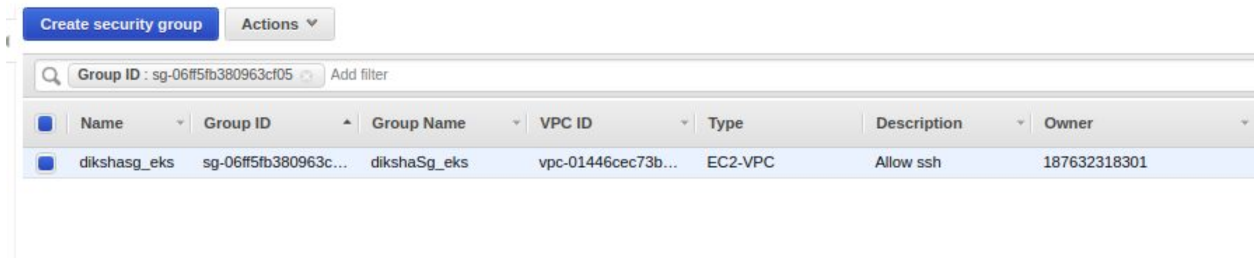
ASSESSMENT ON : EKS

TO THE NEW

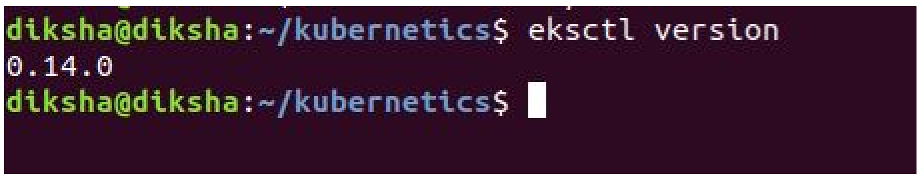
1. **Create eks cluster using eksctl**
   During creation, Specify
   - **Cluster name**
   - **Kubernetes version**
   - **Control plane role**
   - **Subnets for Control Plane**
   - **Control Plane security Group**
   - **Add tag: owner, purpose on Control Plane**
   - **Node Group Name**
   - **Node Instance Role**
   - **Subnets for Node Group**
   - **Node Instance SSH key pair**
   - **Node Instance Security Group**
   - **Node Instance Instance Type**
   - **Node Instance Disk**
   - **Add tag: owner, purpose on Node Group**
   - **Node Group Size: min, max**

ANS:

STEP 1: Create a security group

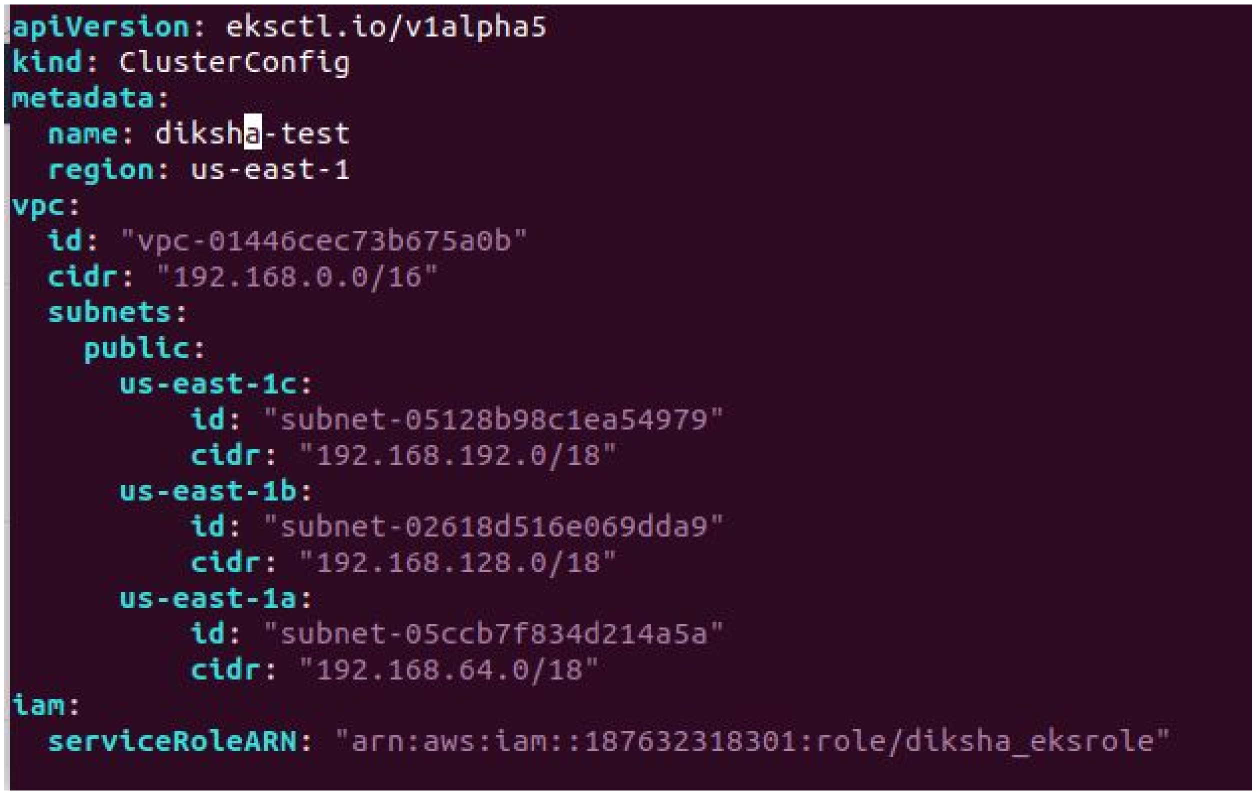


Name	Group ID	Group Name	VPC ID	Type	Description	Owner
dikshasg_eks	sg-06ff5fb380963c...	dikshaSg_eks	vpc-01446cec73b...	EC2-VPC	Allow ssh	187632318301

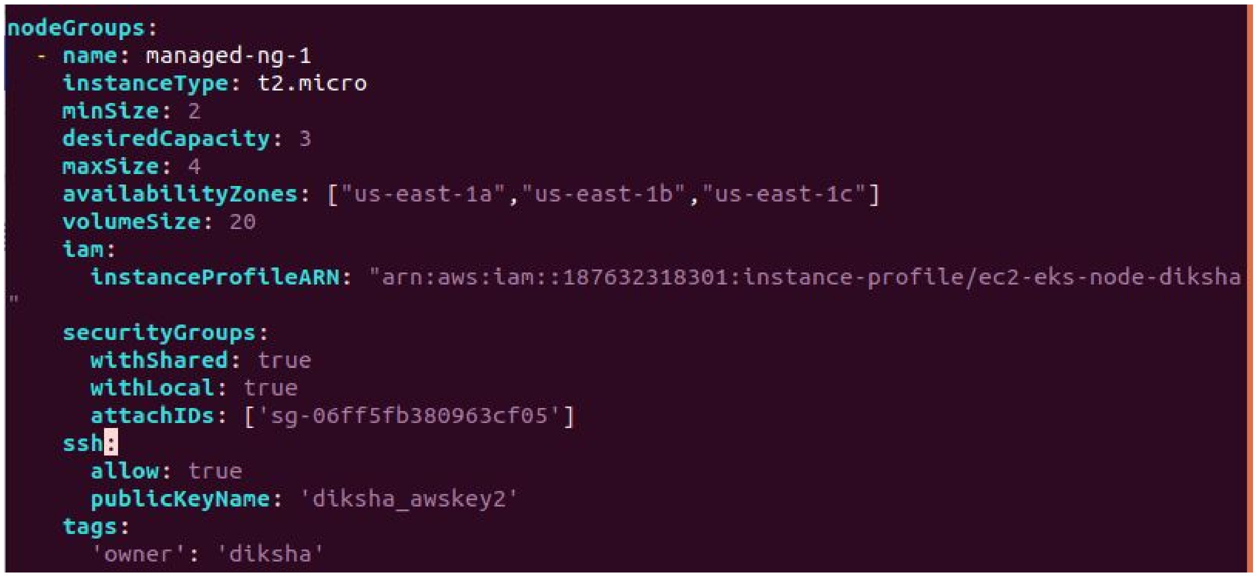


```
diksha@diksha:~/kubernetics$ eksctl version
0.14.0
diksha@diksha:~/kubernetics$ 
```

STEP 2: Create a yaml file with all the above configuration

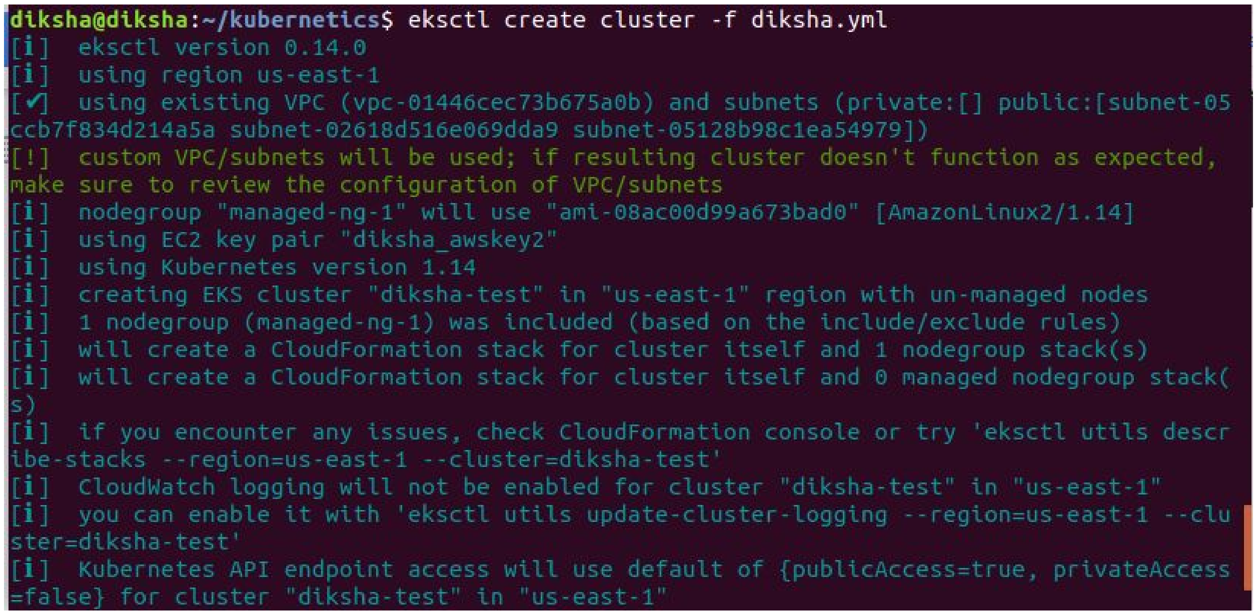


```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: diksha-test
  region: us-east-1
vpc:
  id: "vpc-01446cec73b675a0b"
  cidr: "192.168.0.0/16"
  subnets:
    public:
      us-east-1c:
          id: "subnet-05128b98c1ea54979"
          cidr: "192.168.192.0/18"
      us-east-1b:
          id: "subnet-02618d516e069dda9"
          cidr: "192.168.128.0/18"
      us-east-1a:
          id: "subnet-05ccb7f834d214a5a"
          cidr: "192.168.64.0/18"
iam:
  serviceRoleARN: "arn:aws:iam::187632318301:role/diksha_eksrole"
```

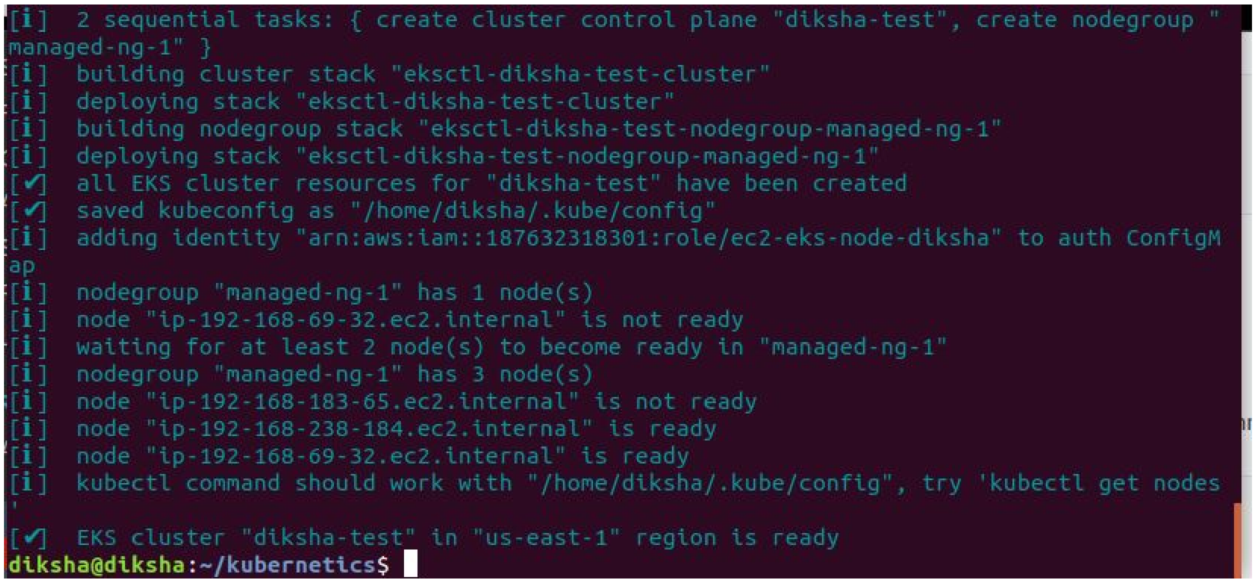


```
nodeGroups:
  - name: managed-ng-1
    instanceType: t2.micro
    minSize: 2
    desiredCapacity: 3
    maxSize: 4
    availabilityZones: ["us-east-1a","us-east-1b","us-east-1c"]
    volumeSize: 20
    iam:
      instanceProfileARN: "arn:aws:iam::187632318301:instance-profile/ec2-eks-node-diksha
"
    securityGroups:
      withShared: true
      withLocal: true
      attachIDs: ['sg-06ff5fb380963cf05']
    ssh:
      allow: true
      publicKeyName: 'diksha_awskey2'
    tags:
      'owner': 'diksha'
```

STEP 3: Run the below command to create the cluster

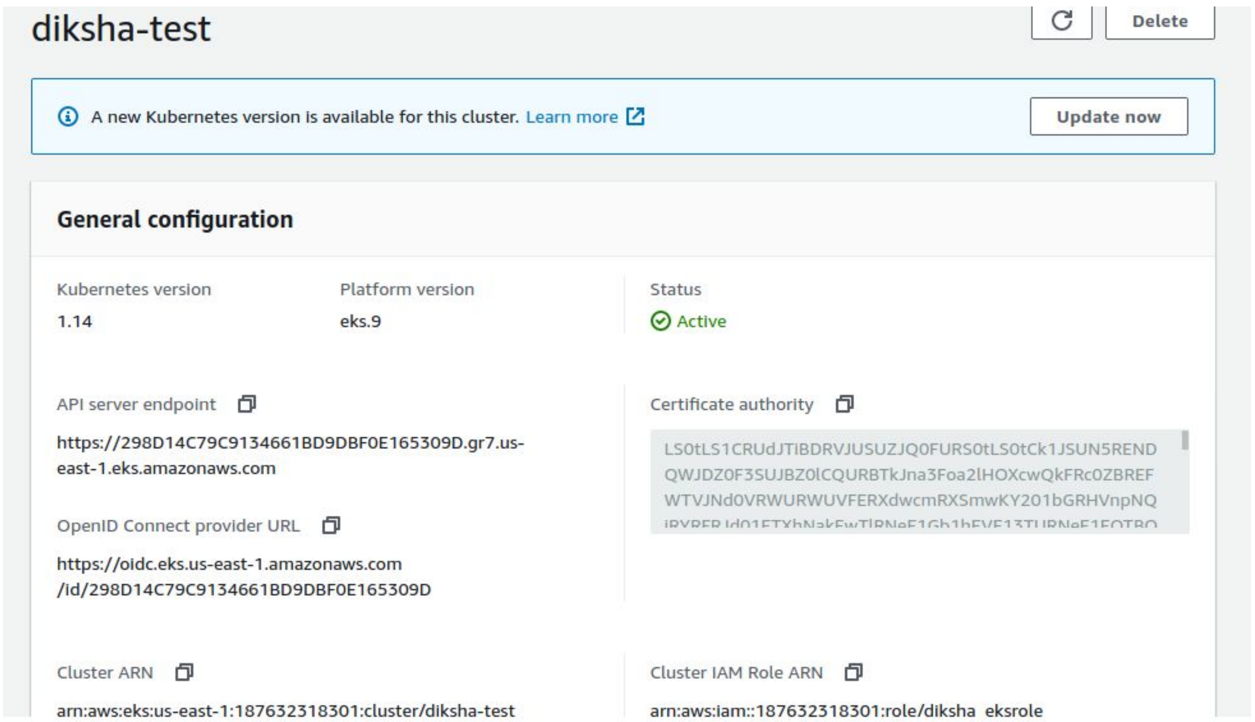


```
diksha@diksha:~/kubernetics$ eksctl create cluster -f diksha.yml
[ℹ]  eksctl version 0.14.0
[ℹ]  using region us-east-1
[✔]  using existing VPC (vpc-01446cec73b675a0b) and subnets (private:[] public:[subnet-05
ccb7f834d214a5a subnet-02618d516e069dda9 subnet-05128b98c1ea54979])
[!]  custom VPC/subnets will be used; if resulting cluster doesn't function as expected,
make sure to review the configuration of VPC/subnets
[ℹ]  nodegroup "managed-ng-1" will use "ami-08ac00d99a673bad0" [AmazonLinux2/1.14]
[ℹ]  using EC2 key pair "diksha_awskey2"
[ℹ]  using Kubernetes version 1.14
[ℹ]  creating EKS cluster "diksha-test" in "us-east-1" region with un-managed nodes
[ℹ]  1 nodegroup (managed-ng-1) was included (based on the include/exclude rules)
[ℹ]  will create a CloudFormation stack for cluster itself and 1 nodegroup stack(s)
[ℹ]  will create a CloudFormation stack for cluster itself and 0 managed nodegroup stack(
s)
[ℹ]  if you encounter any issues, check CloudFormation console or try 'eksctl utils descr
ibe-stacks --region=us-east-1 --cluster=diksha-test'
[ℹ]  CloudWatch logging will not be enabled for cluster "diksha-test" in "us-east-1"
[ℹ]  you can enable it with 'eksctl utils update-cluster-logging --region=us-east-1 --clu
ster=diksha-test'
[ℹ]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess
=false} for cluster "diksha-test" in "us-east-1"
```



```
[ℹ]  2 sequential tasks: { create cluster control plane "diksha-test", create nodegroup "
managed-ng-1" }
[ℹ]  building cluster stack "eksctl-diksha-test-cluster"
[ℹ]  deploying stack "eksctl-diksha-test-cluster"
[ℹ]  building nodegroup stack "eksctl-diksha-test-nodegroup-managed-ng-1"
[ℹ]  deploying stack "eksctl-diksha-test-nodegroup-managed-ng-1"
[✔]  all EKS cluster resources for "diksha-test" have been created
[✔]  saved kubeconfig as "/home/diksha/.kube/config"
[ℹ]  adding identity "arn:aws:iam::187632318301:role/ec2-eks-node-diksha" to auth ConfigM
ap
[ℹ]  nodegroup "managed-ng-1" has 1 node(s)
[ℹ]  node "ip-192-168-69-32.ec2.internal" is not ready
[ℹ]  waiting for at least 2 node(s) to become ready in "managed-ng-1"
[ℹ]  nodegroup "managed-ng-1" has 3 node(s)
[ℹ]  node "ip-192-168-183-65.ec2.internal" is not ready
[ℹ]  node "ip-192-168-238-184.ec2.internal" is ready
[ℹ]  node "ip-192-168-69-32.ec2.internal" is ready
[ℹ]  kubectl command should work with "/home/diksha/.kube/config", try 'kubectl get nodes
'
[✔]  EKS cluster "diksha-test" in "us-east-1" region is ready
diksha@diksha:~/kubernetics$
```

STEP 4: Cluster is being created


diksha-test

A new Kubernetes version is available for this cluster. Learn more

General configuration

Kubernetes version: 1.14

Platform version: eks.9

Status: Active

API server endpoint: https://298D14C79C9134661BD9DBF0E165309D.gr7.us-east-1.eks.amazonaws.com

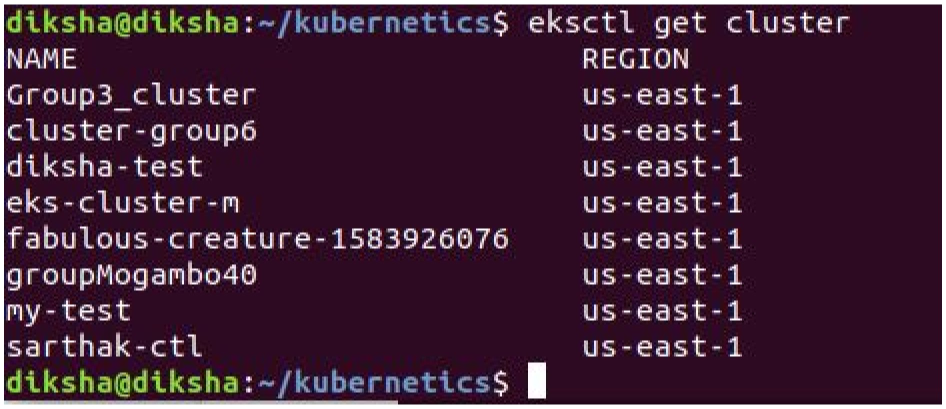
OpenID Connect provider URL: https://oidc.eks.us-east-1.amazonaws.com/id/298D14C79C9134661BD9DBF0E165309D

Certificate authority:

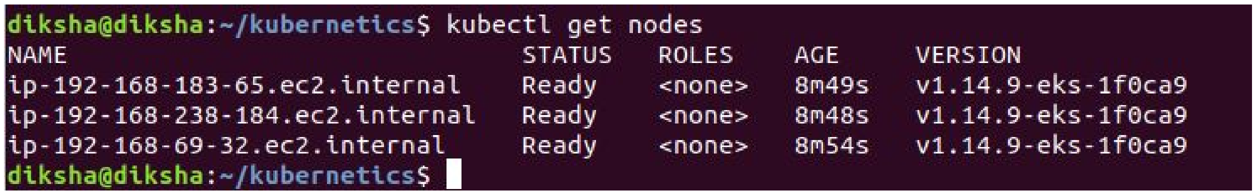
```
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN5REND
QWJDZ0F3SUJBZ0lCQURBTkJna3Foa2lHOXcwQkFRc0ZBREF
WTVJNd0VRWURWUVFERXdwcmRXSmwKY201bGRHVnpNQ
jRYREBJd01FTXhNakEwTlRNeE1Gb1hFVE13TURNeE1FOTBQ
```

Cluster ARN: arn:aws:eks:us-east-1:187632318301:cluster/diksha-test

Cluster IAM Role ARN: arn:aws:iam::187632318301:role/diksha eksrole



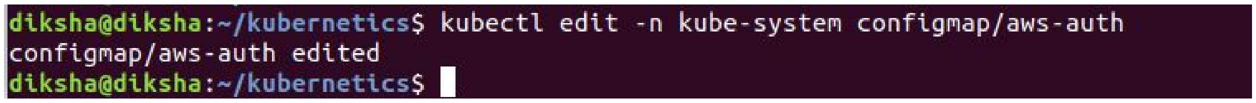
```
diksha@diksha:~/kubernetics$ eksctl get cluster
NAME                            REGION
Group3_cluster                  us-east-1
cluster-group6                  us-east-1
diksha-test                     us-east-1
eks-cluster-m                   us-east-1
fabulous-creature-1583926076    us-east-1
groupMogambo40                  us-east-1
my-test                         us-east-1
sarthak-ctl                     us-east-1
diksha@diksha:~/kubernetics$
```



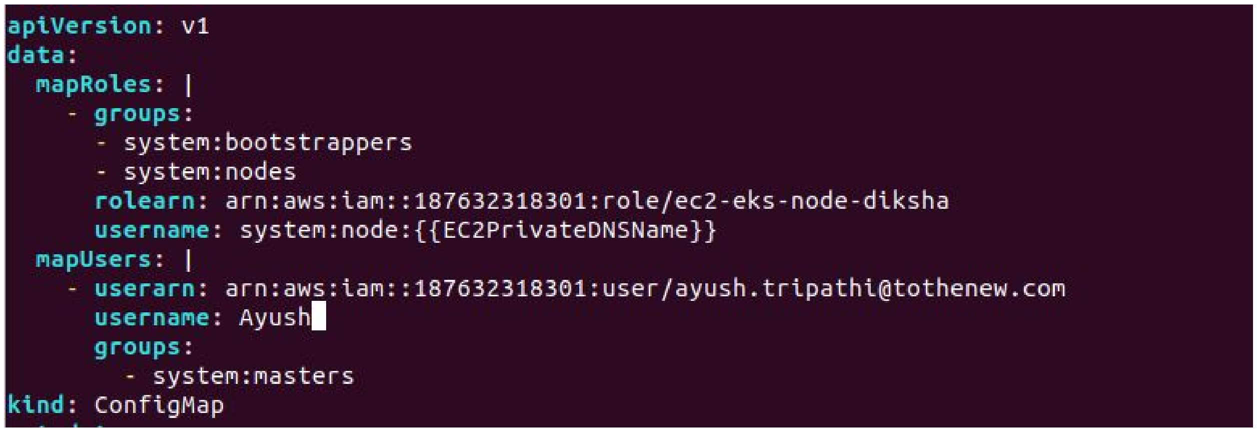
```
diksha@diksha:~/kubernetics$ kubectl get nodes
NAME                              STATUS   ROLES    AGE     VERSION
ip-192-168-183-65.ec2.internal    Ready    <none>   8m49s   v1.14.9-eks-1f0ca9
ip-192-168-238-184.ec2.internal   Ready    <none>   8m48s   v1.14.9-eks-1f0ca9
ip-192-168-69-32.ec2.internal     Ready    <none>   8m54s   v1.14.9-eks-1f0ca9
diksha@diksha:~/kubernetics$
```

2.Authentication Management

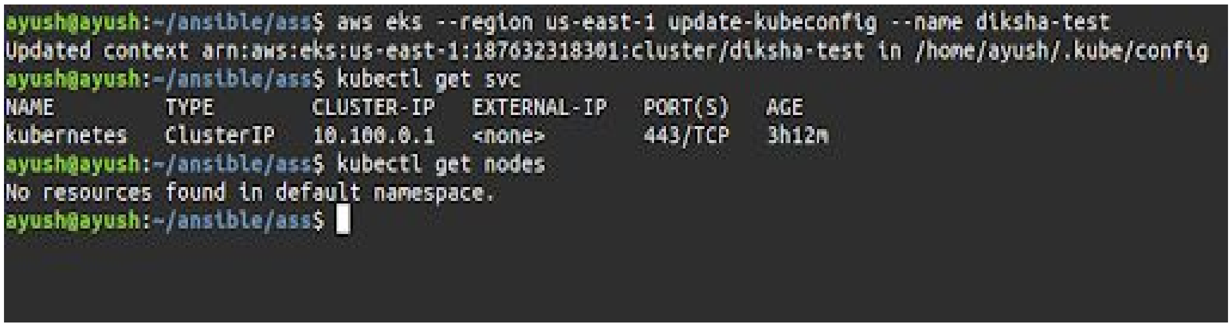
a. **Add new 2 IAM user into the cluster**



```
diksha@diksha:~/kubernetics$ kubectl edit -n kube-system configmap/aws-auth
configmap/aws-auth edited
diksha@diksha:~/kubernetics$
```



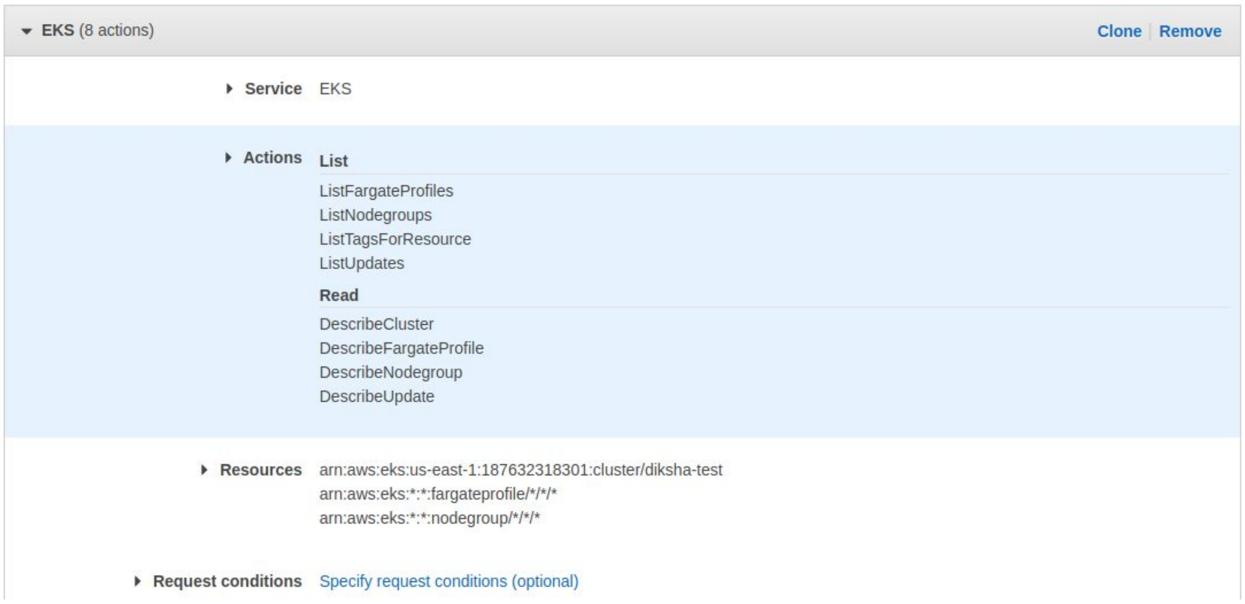
```
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::187632318301:role/ec2-eks-node-diksha
      username: system:node:{{EC2PrivateDNSName}}
  mapUsers: |
    - userarn: arn:aws:iam::187632318301:user/ayush.tripathi@tothenew.com
      username: Ayush
      groups:
        - system:masters
kind: ConfigMap
```



```
ayush@ayush:~/ansible/ass$ aws eks --region us-east-1 update-kubeconfig --name diksha-test
Updated context arn:aws:eks:us-east-1:187632318301:cluster/diksha-test in /home/ayush/.kube/config
ayush@ayush:~/ansible/ass$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   3h12m
ayush@ayush:~/ansible/ass$ kubectl get nodes
No resources found in default namespace.
ayush@ayush:~/ansible/ass$
```

b. **Enable an EC2 server to access Cluster master API without using access/secret key**

STEP 1: Create a new policy and specify service=EKS



EKS (8 actions) Clone | Remove

Service: EKS

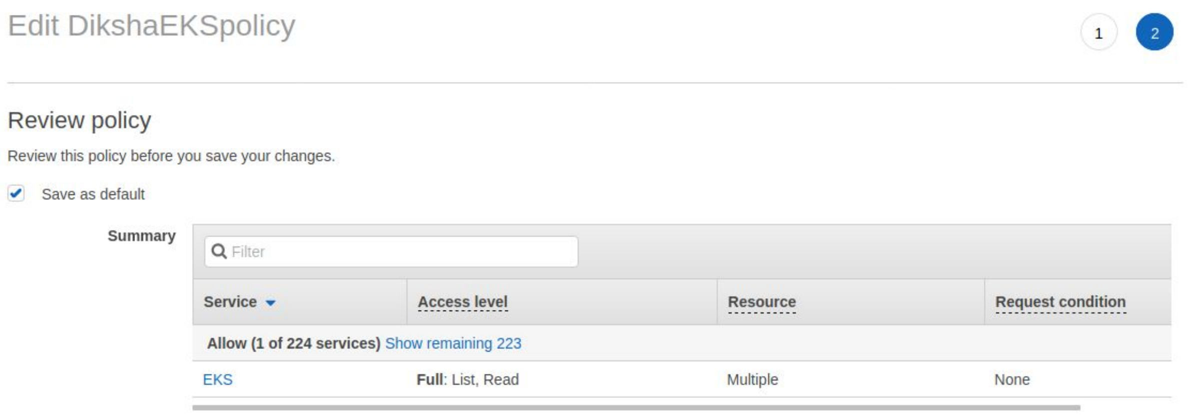
Actions:

List
- ListFargateProfiles
- ListNodegroups
- ListTagsForResource
- ListUpdates

Read
- DescribeCluster
- DescribeFargateProfile
- DescribeNodegroup
- DescribeUpdate

Resources:
arn:aws:eks:us-east-1:187632318301:cluster/diksha-test
arn:aws:eks:*:*:fargateprofile/*/*/*
arn:aws:eks:*:*:nodegroup/*/*/*

Request conditions: Specify request conditions (optional)



Edit DikshaEKSpolicy

Review policy

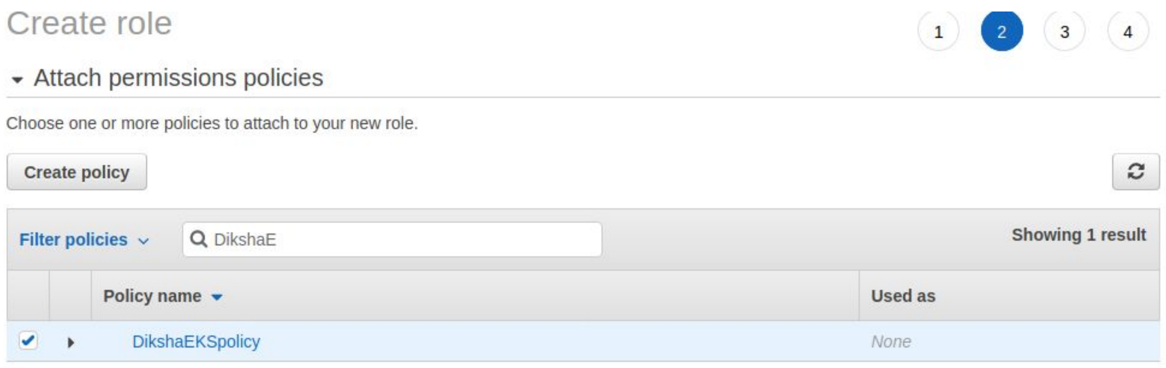
Review this policy before you save your changes.

☑ Save as default

Summary

Service	Access level	Resource	Request condition
Allow (1 of 224 services) Show remaining 223			
EKS	**Full**: List, Read	Multiple	None

STEP 2: Create a new role and attach the above policy



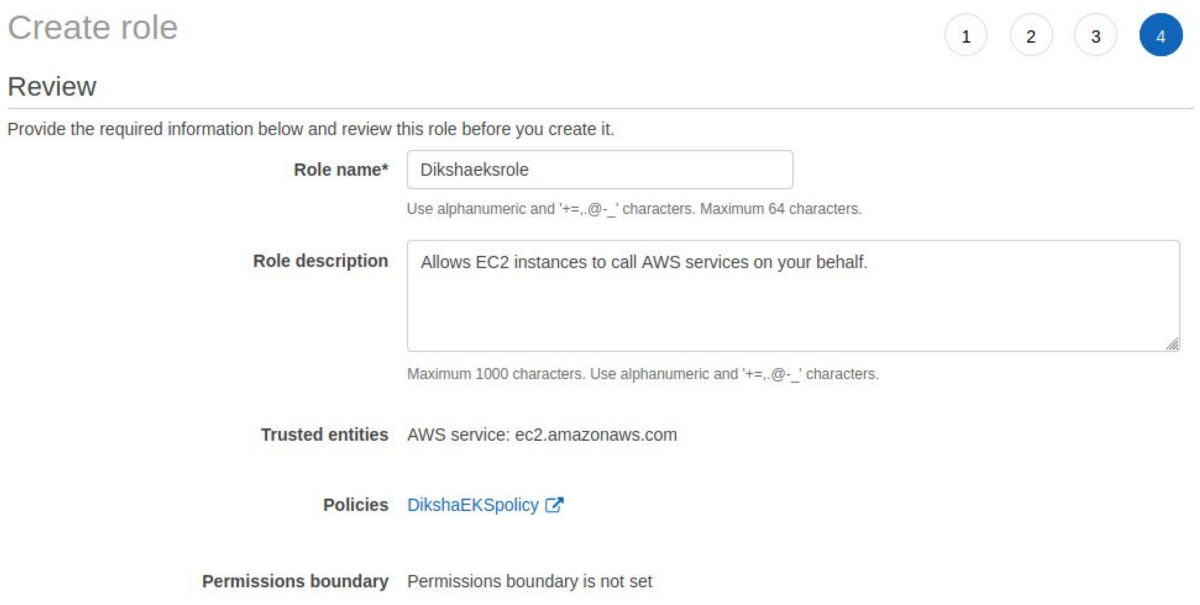
Create role

Attach permissions policies

Choose one or more policies to attach to your new role.

Filter policies: DikshaE — Showing 1 result

		Policy name	Used as
☑	▸	DikshaEKSpolicy	None



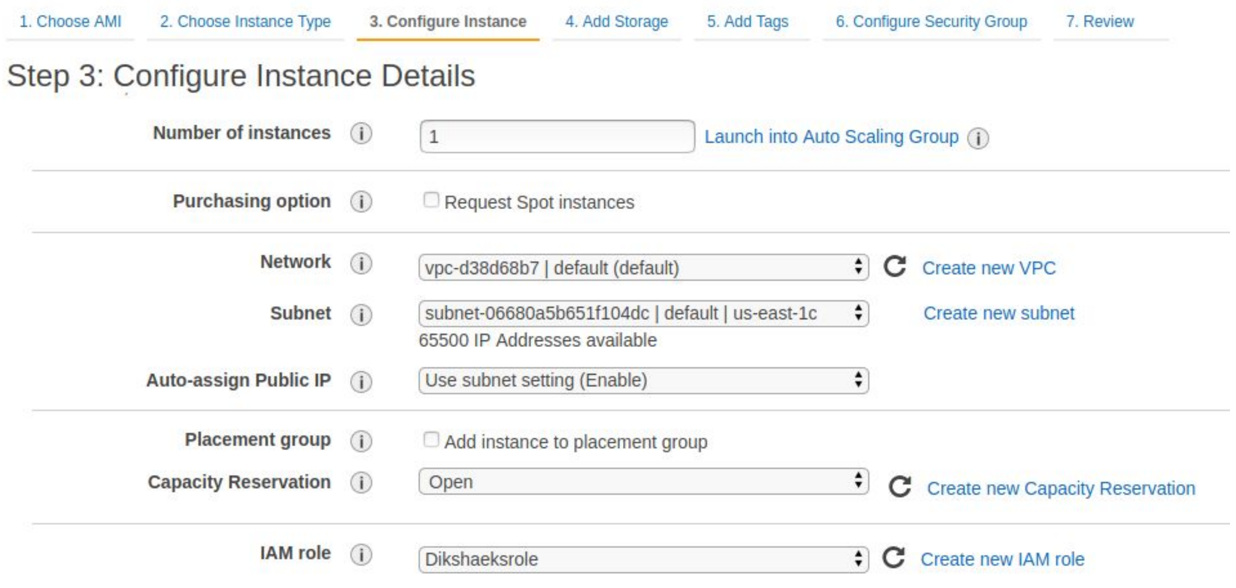
Create role

Review

Provide the required information below and review this role before you create it.

Role name*	Dikshaeksrole
	Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.
Role description	Allows EC2 instances to call AWS services on your behalf.
	Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.
Trusted entities	AWS service: ec2.amazonaws.com
Policies	DikshaEKSpolicy
Permissions boundary	Permissions boundary is not set

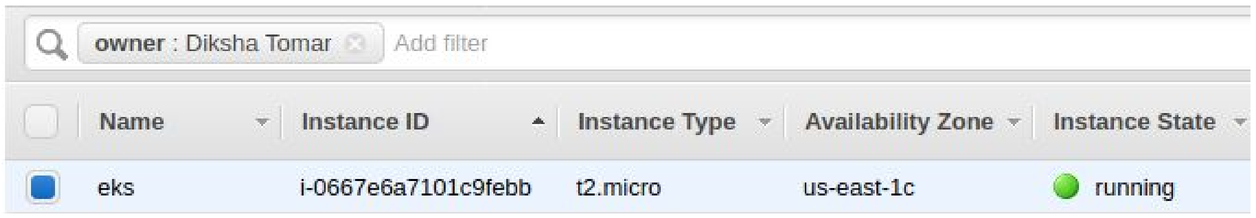
STEP 3: Now launch an instance and attach this role to that instance.



Step 3: Configure Instance Details

Number of instances	1
Purchasing option	Request Spot instances
Network	vpc-d38d68b7 | default (default)
Subnet	subnet-06680a5b651f104dc | default | us-east-1c 65500 IP Addresses available
Auto-assign Public IP	Use subnet setting (Enable)
Placement group	Add instance to placement group
Capacity Reservation	Open
IAM role	Dikshaeksrole

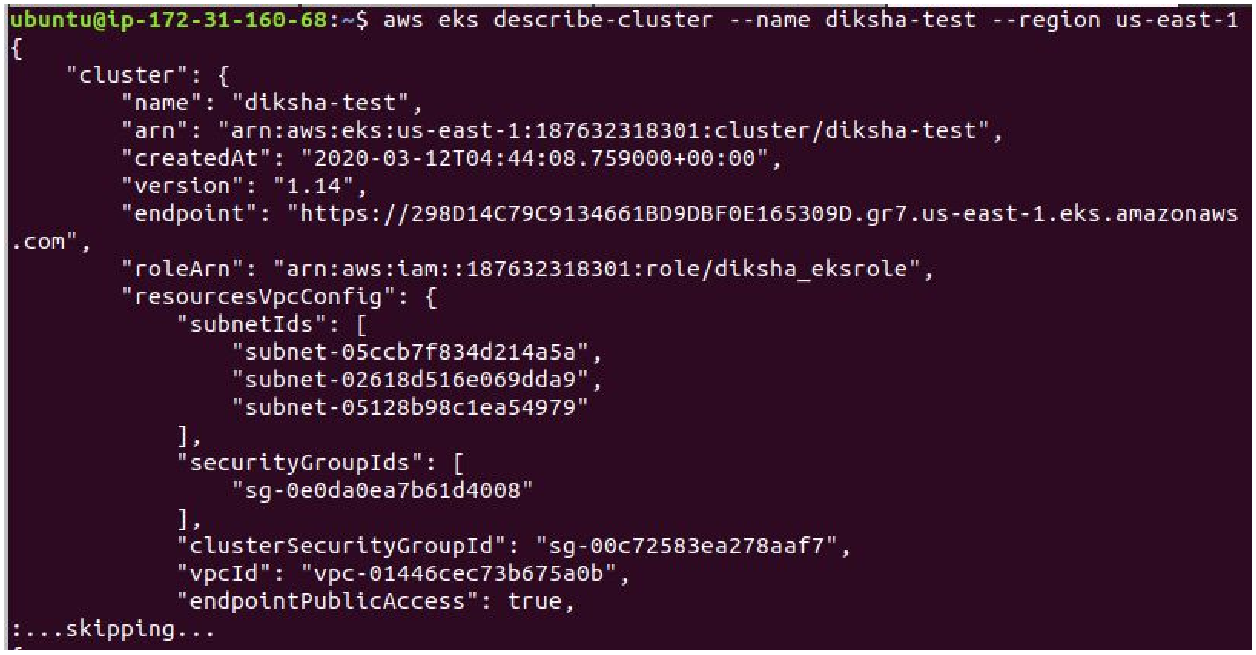
Launch into Auto Scaling Group
Create new VPC
Create new subnet
Create new Capacity Reservation
Create new IAM role



owner : Diksha Tomar

Name	Instance ID	Instance Type	Availability Zone	Instance State
eks	i-0667e6a7101c9febb	t2.micro	us-east-1c	running

STEP 4: Run the below command and you will observe that your Instance is able to access your cluster.

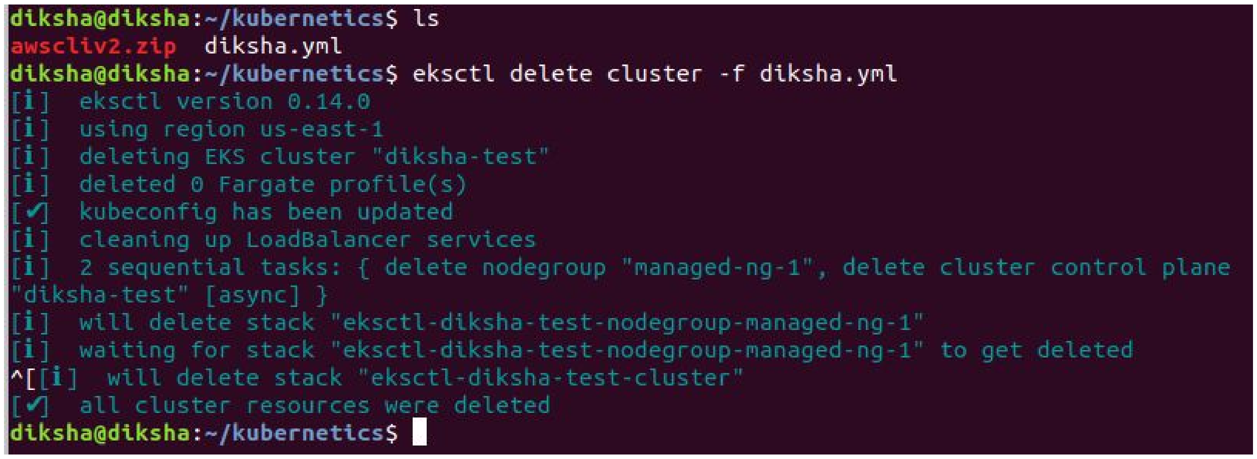


```
ubuntu@ip-172-31-160-68:~$ aws eks describe-cluster --name diksha-test --region us-east-1
{
    "cluster": {
        "name": "diksha-test",
        "arn": "arn:aws:eks:us-east-1:187632318301:cluster/diksha-test",
        "createdAt": "2020-03-12T04:44:08.759000+00:00",
        "version": "1.14",
        "endpoint": "https://298D14C79C9134661BD9DBF0E165309D.gr7.us-east-1.eks.amazonaws
.com",
        "roleArn": "arn:aws:iam::187632318301:role/diksha_eksrole",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-05ccb7f834d214a5a",
                "subnet-02618d516e069dda9",
                "subnet-05128b98c1ea54979"
            ],
            "securityGroupIds": [
                "sg-0e0da0ea7b61d4008"
            ],
            "clusterSecurityGroupId": "sg-00c72583ea278aaf7",
            "vpcId": "vpc-01446cec73b675a0b",
            "endpointPublicAccess": true,
:...skipping...
```

3. Eksctl command to terminate the stack

STEP 1: Run the following command to delete the cluster:

$ eksctl delete cluster -f <YAML file>



```
diksha@diksha:~/kubernetics$ ls
awscliv2.zip  diksha.yml
diksha@diksha:~/kubernetics$ eksctl delete cluster -f diksha.yml
[i]  eksctl version 0.14.0
[i]  using region us-east-1
[i]  deleting EKS cluster "diksha-test"
[i]  deleted 0 Fargate profile(s)
[✔]  kubeconfig has been updated
[i]  cleaning up LoadBalancer services
[i]  2 sequential tasks: { delete nodegroup "managed-ng-1", delete cluster control plane
"diksha-test" [async] }
[i]  will delete stack "eksctl-diksha-test-nodegroup-managed-ng-1"
[i]  waiting for stack "eksctl-diksha-test-nodegroup-managed-ng-1" to get deleted
^[[i]  will delete stack "eksctl-diksha-test-cluster"
[✔]  all cluster resources were deleted
diksha@diksha:~/kubernetics$ 
```