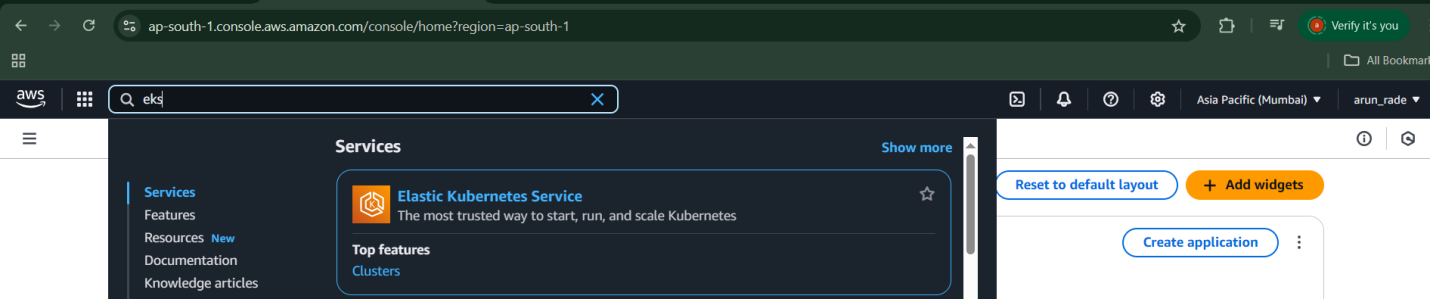
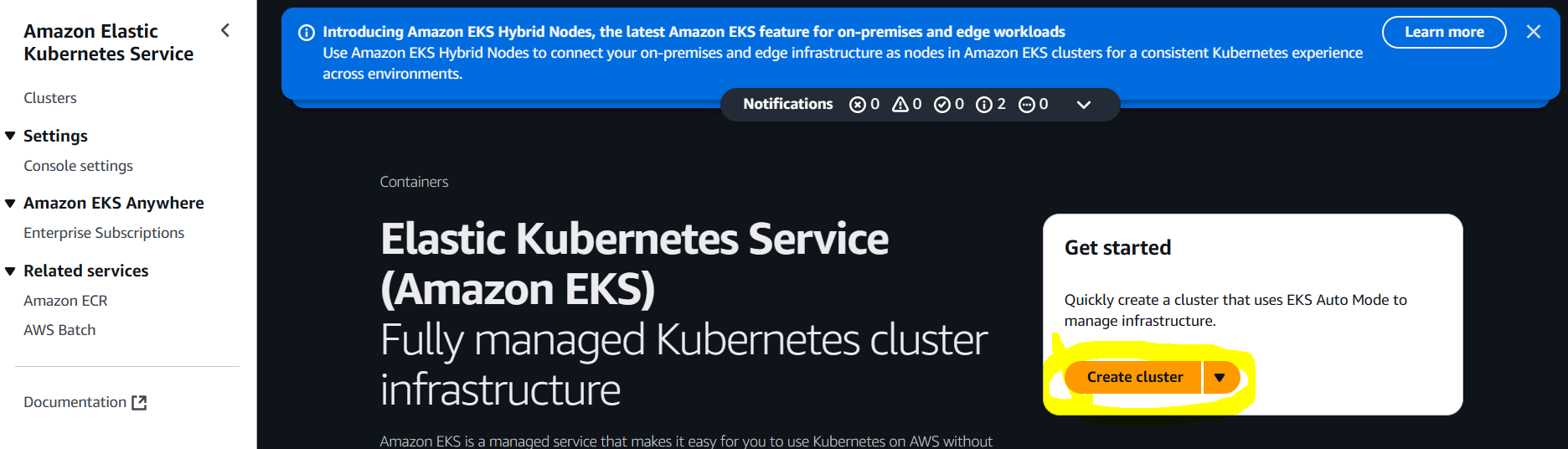
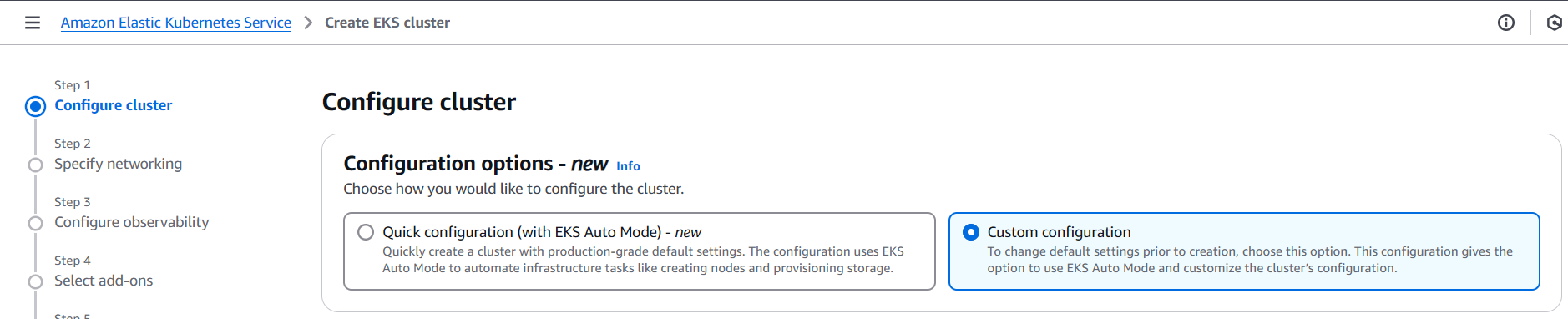
1. Login in AWS
2. Search for EKS
3. Click on “Elastic Kubernetes Services”



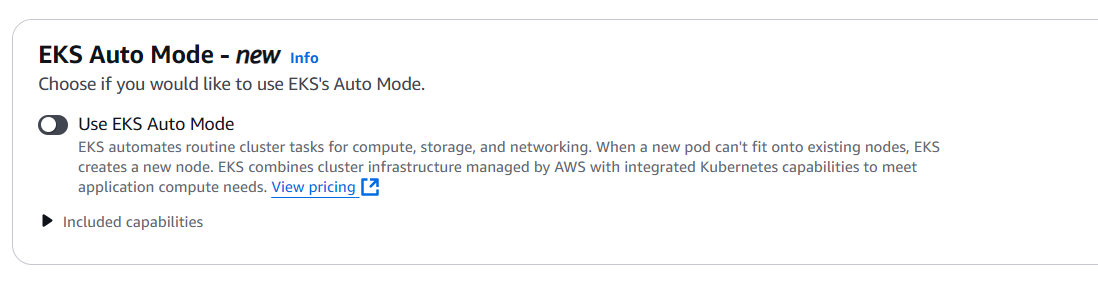
1. Click on “Create Cluster”



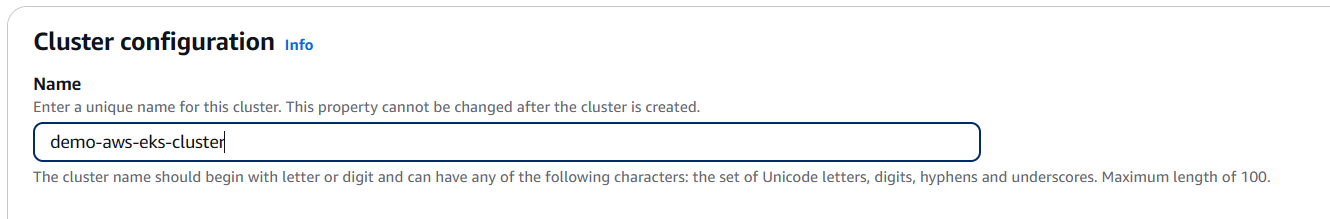
1. Click on “Custom Configuration”



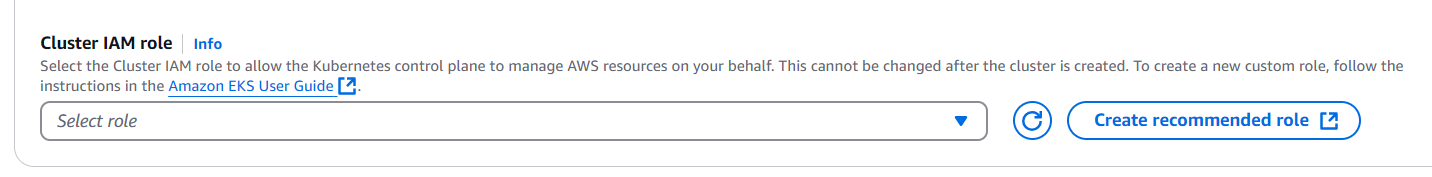
1. Do not select EKS Auto Mode – New



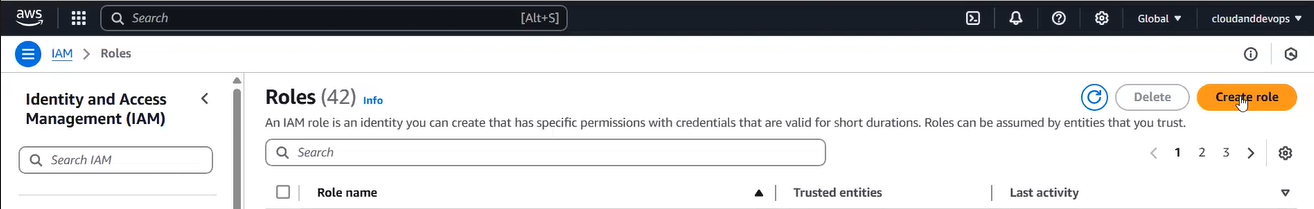
1. Provide name for Cluster:- demo-aws-eks-cluster



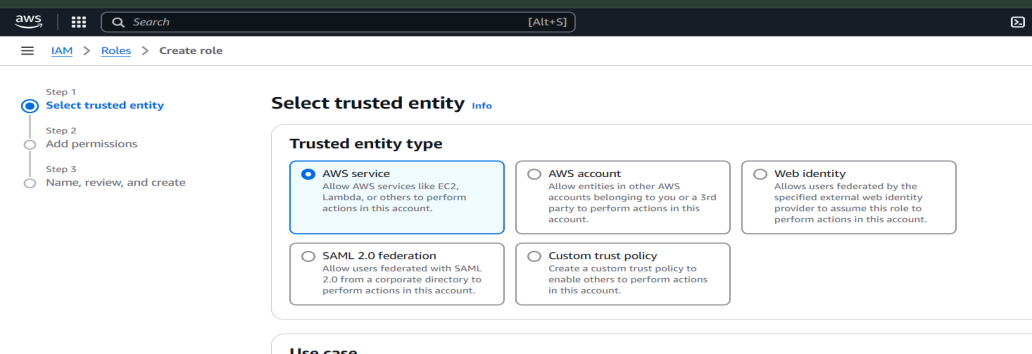
1. Create new role for cluster: click on “Create recommended role” (open in new window)



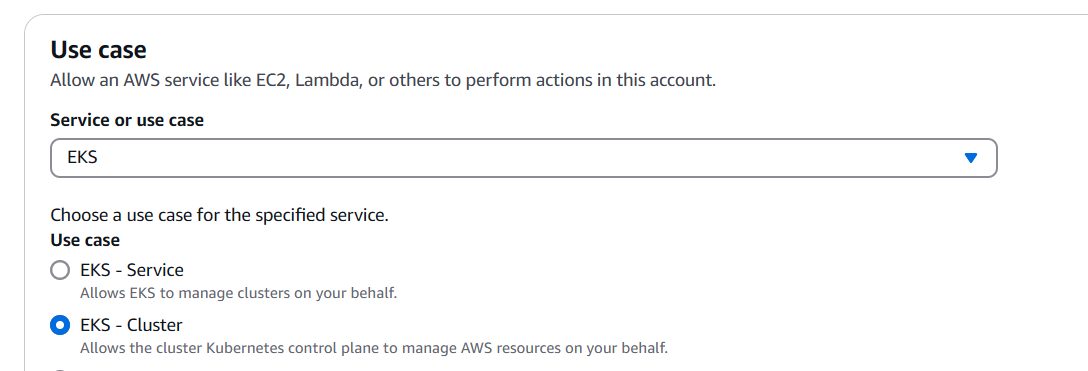
New window will open and need to click on “Create Role”



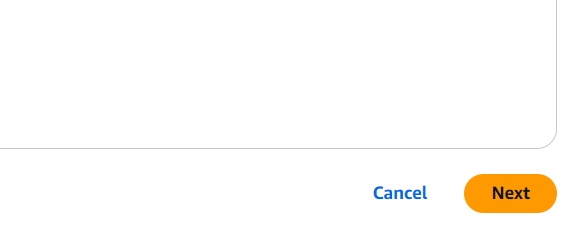
Select “AWS Service”



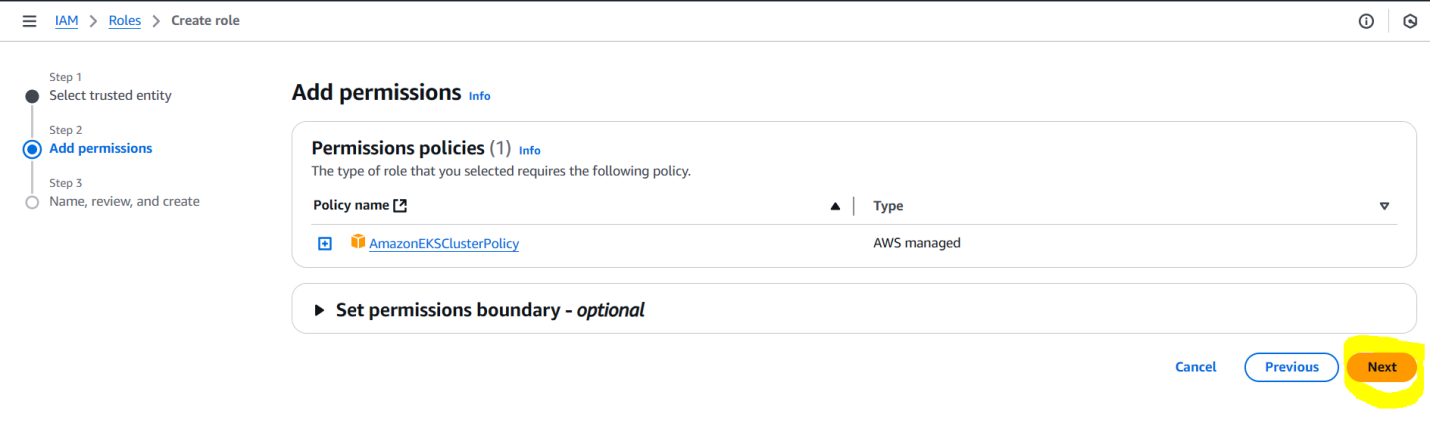
1. Select Service or use case as “EKS” and select case as “EKS-Cluster”



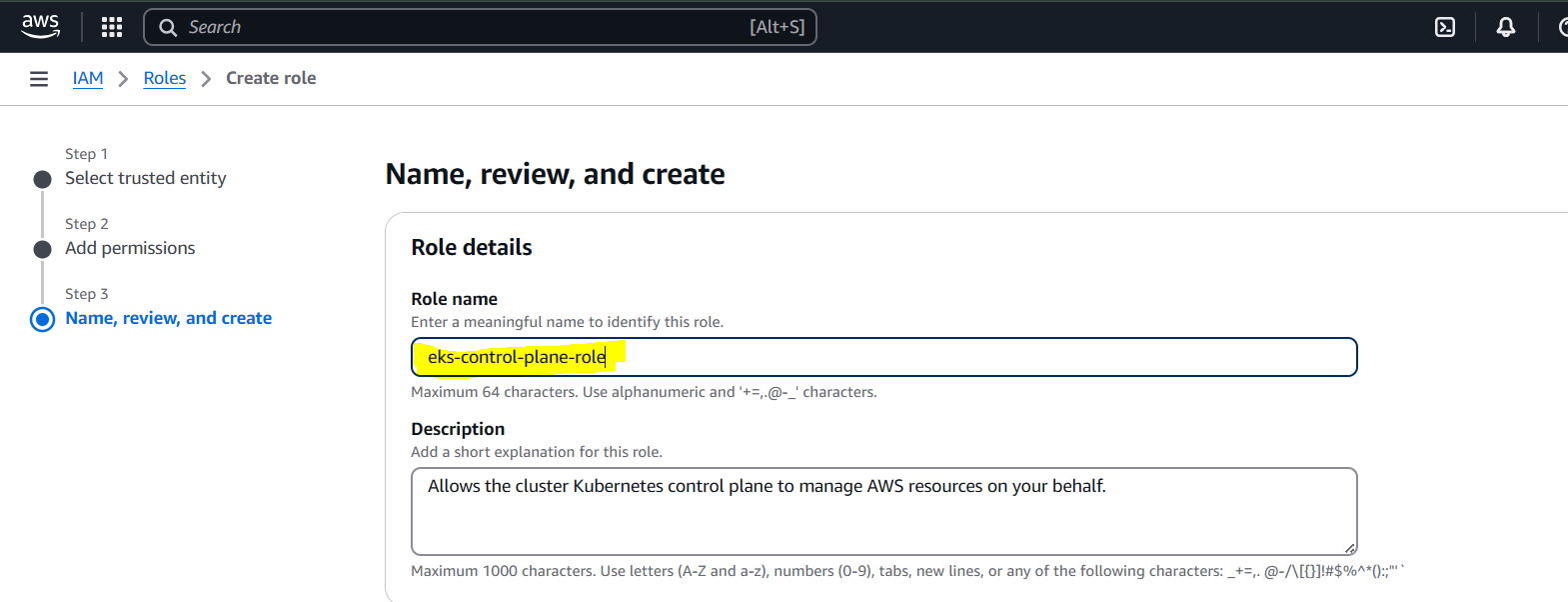
1. Click on “Next”



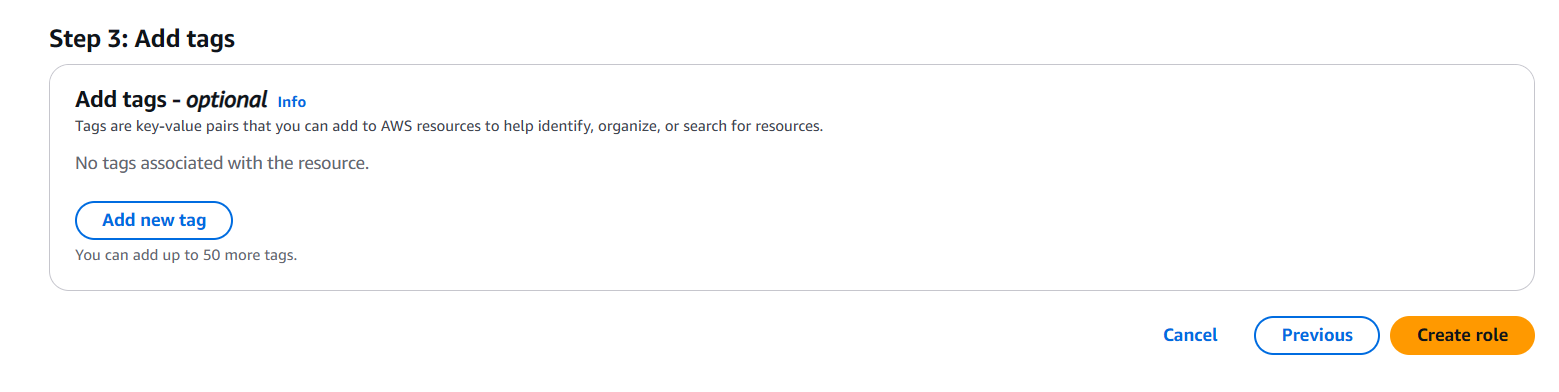
Click on “Next”



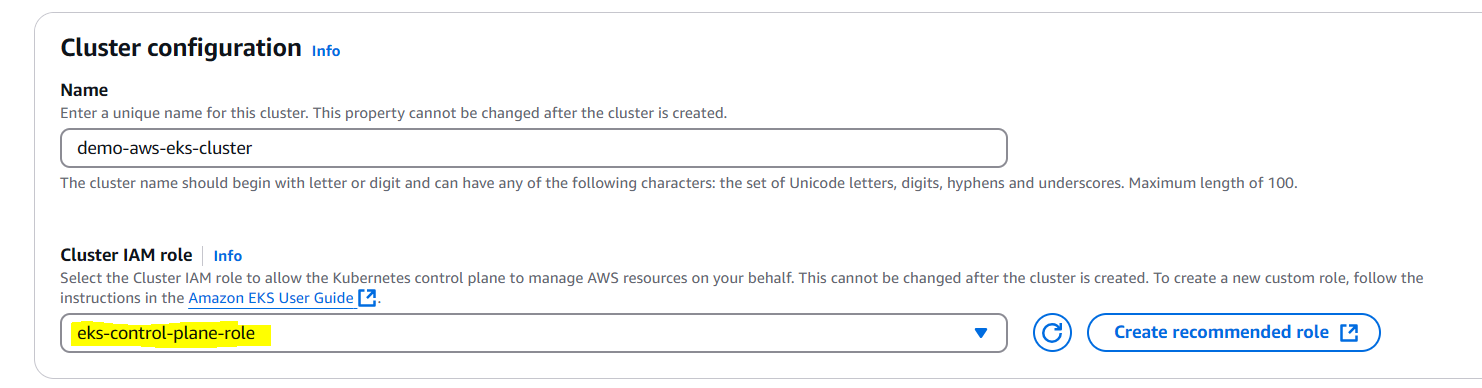
1. Provide cluster name as pr below: eks-control-plane-role



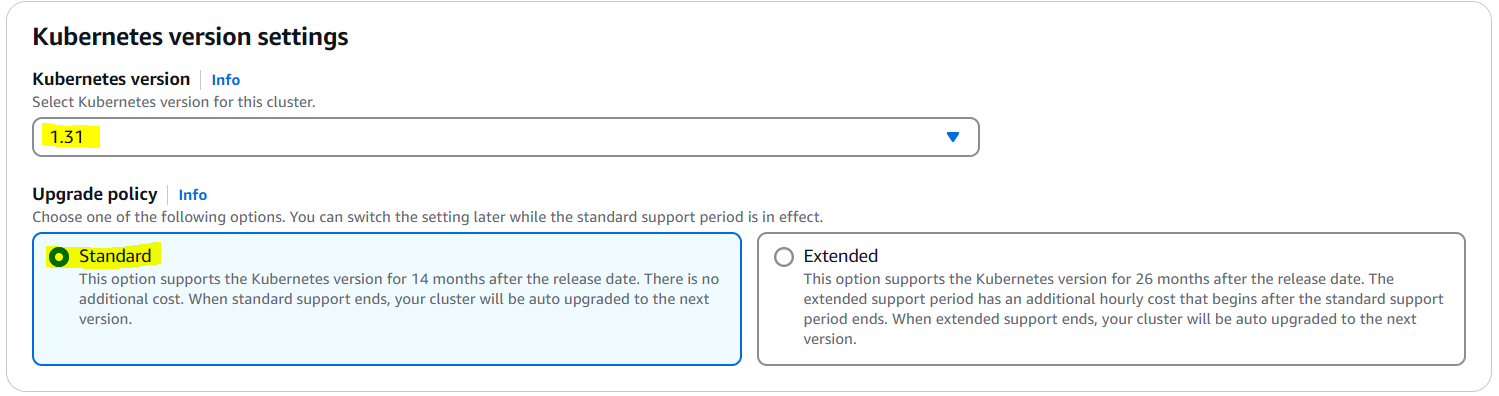
1. Click on Create role button: so new role will create and we are ready to select this role while creating cluster.



1. Goto create EKS cluster page
2. Select newly created role i.e. eks-control-plane-role



1. Select Kubernetes version e.g. 1.31 and select upgrade policy as “Standard”



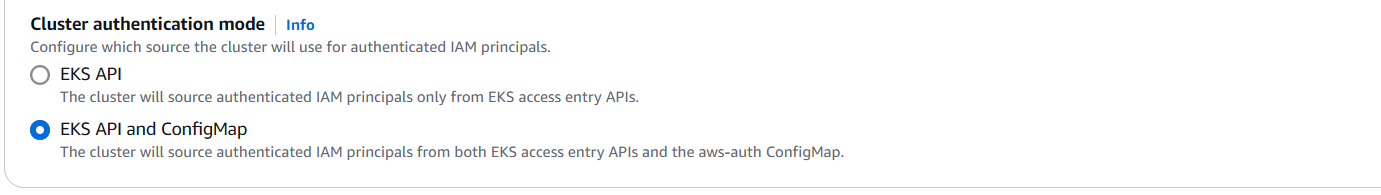
1. Don’t do anything for “Auto Mode Compute- New”
2. For Cluster Access:

17a. Bootstap cluster administration access: so select “Allow cluster administration access”

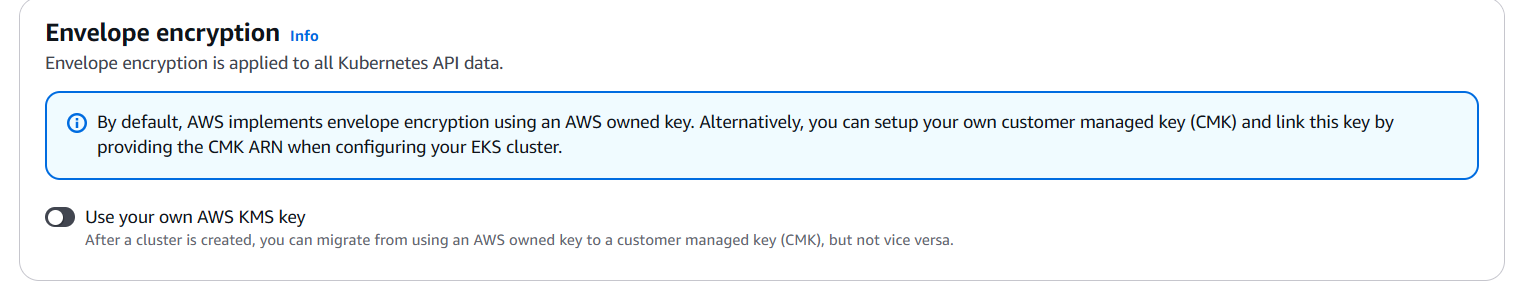


17b. Cluster Authentication Mode

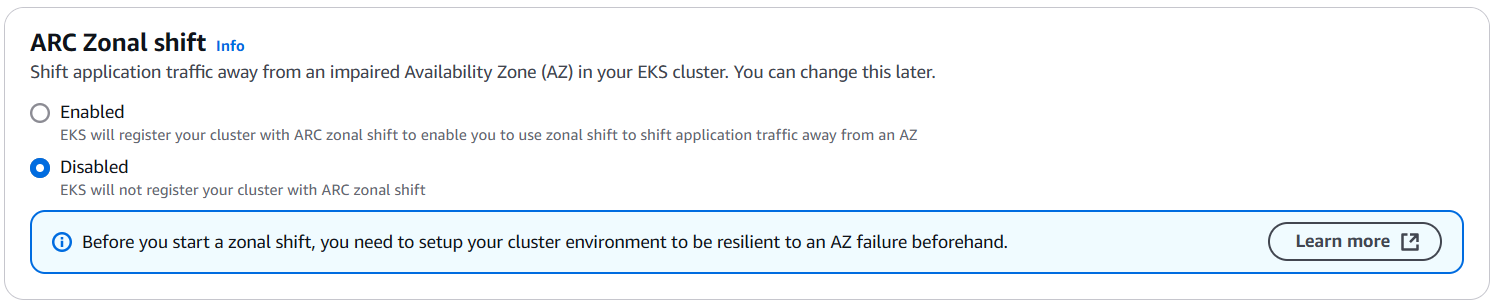
Select “EKS API and ConfigMap”



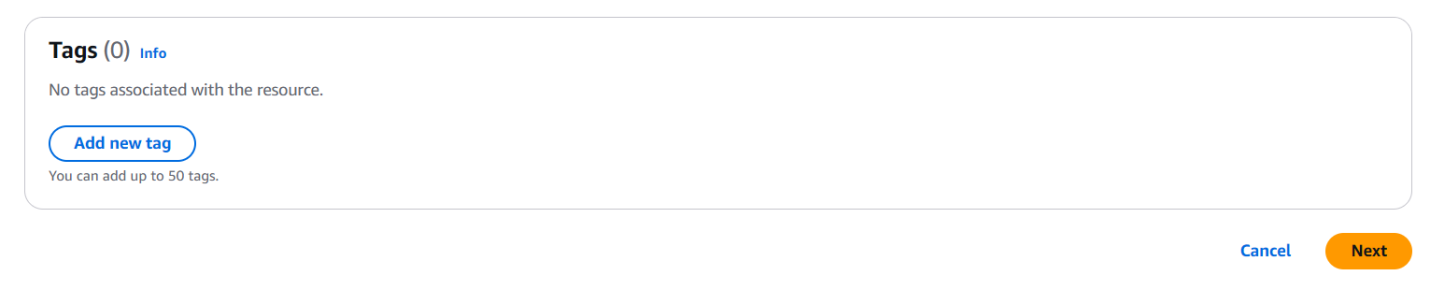
1. Don’t update anything at : Envelope Encryption



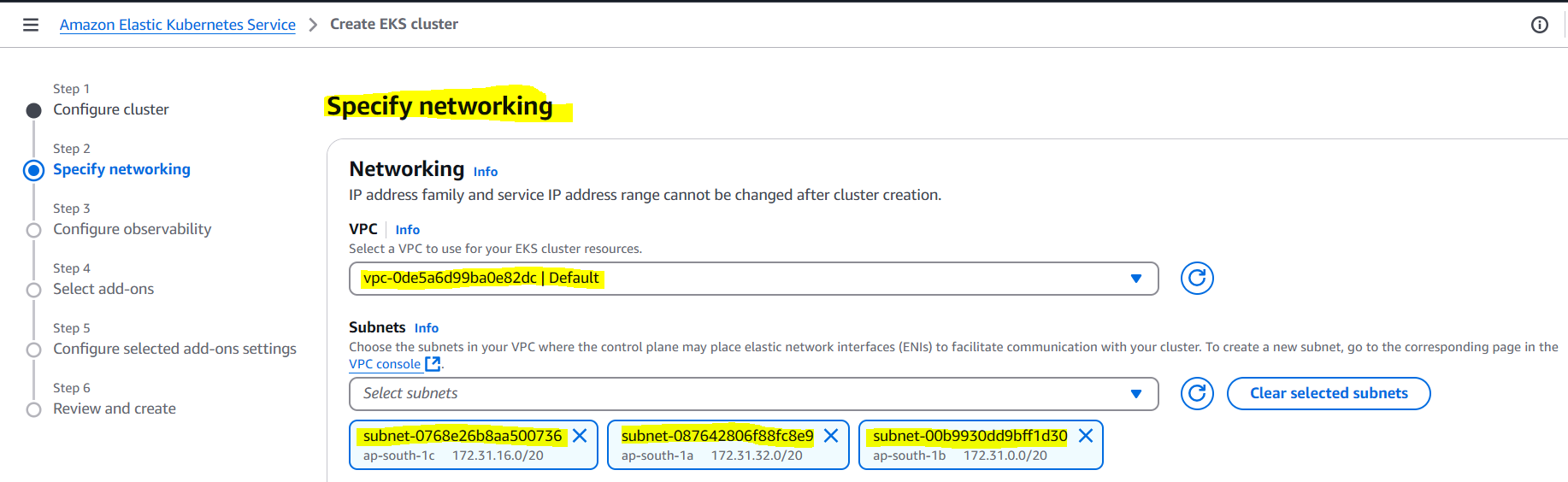
1. Don’t update anything at “ARC Zonal shift”



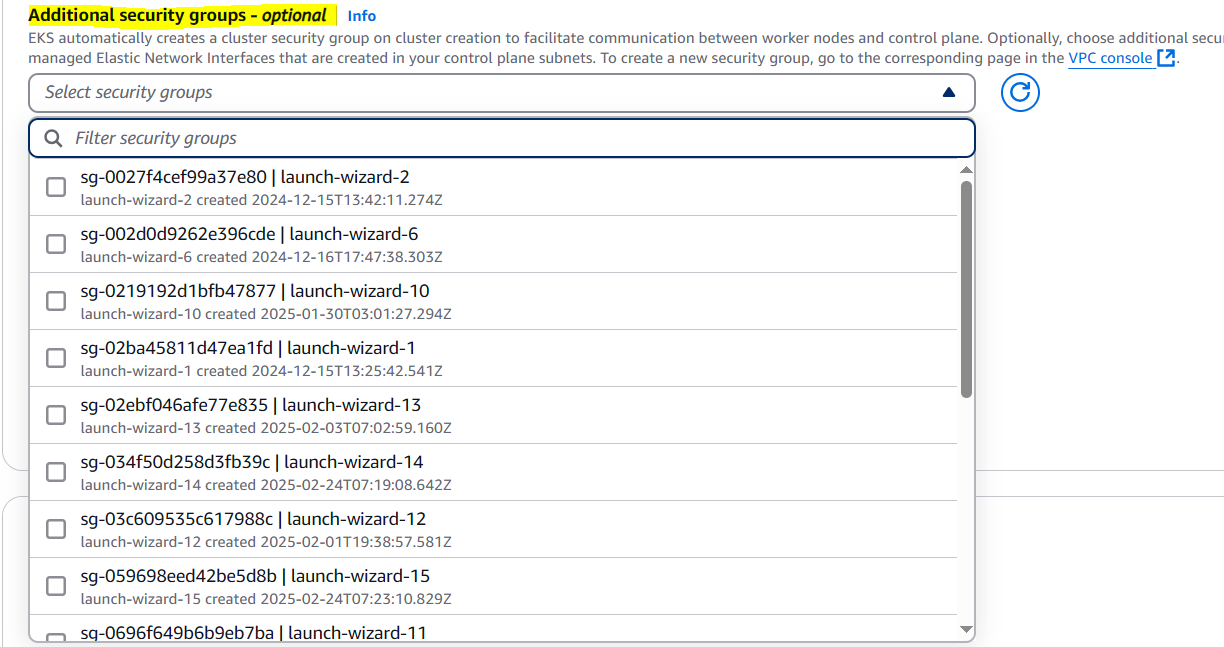
1. Don’t add any tag and Click on “Next”



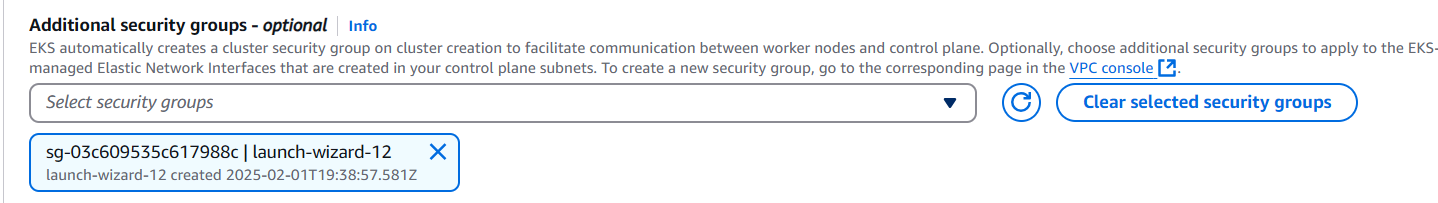
1. For Specify Network, Select VPC as “Default” as per below screen shot



1. **Additional Security Group**: If we have any security group present so please select or else we need to create with specific permission.

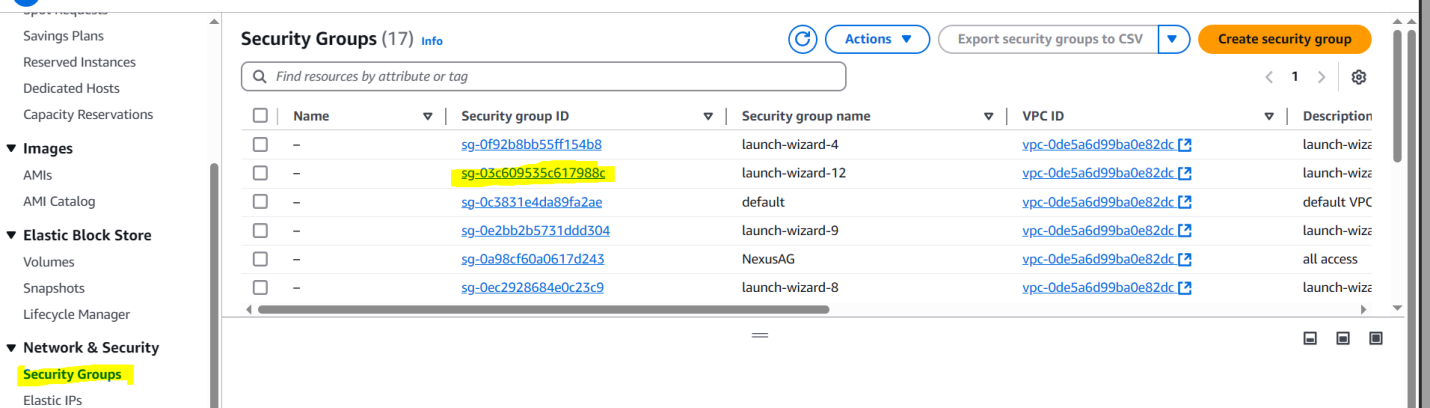


e.g. going to select “launch-wizard-12”

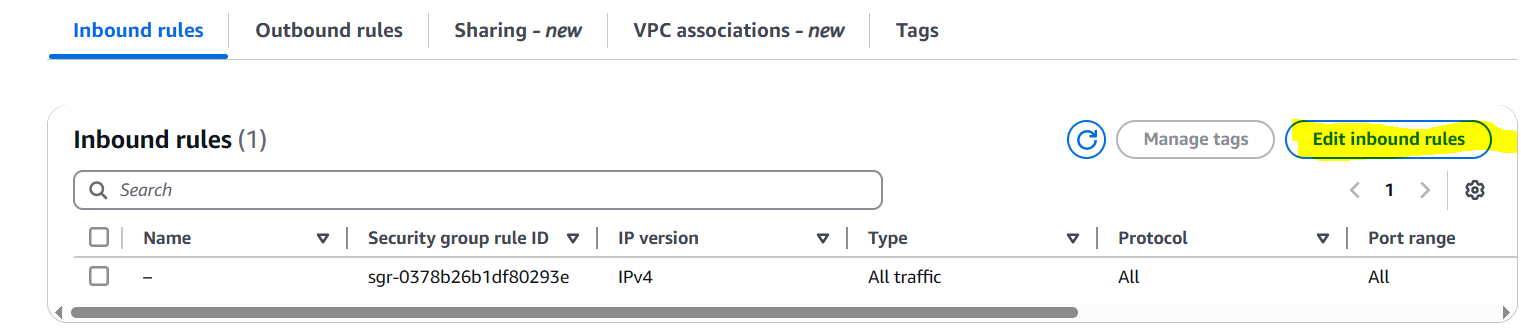


Click on selected group and check the “Inbound Rule”

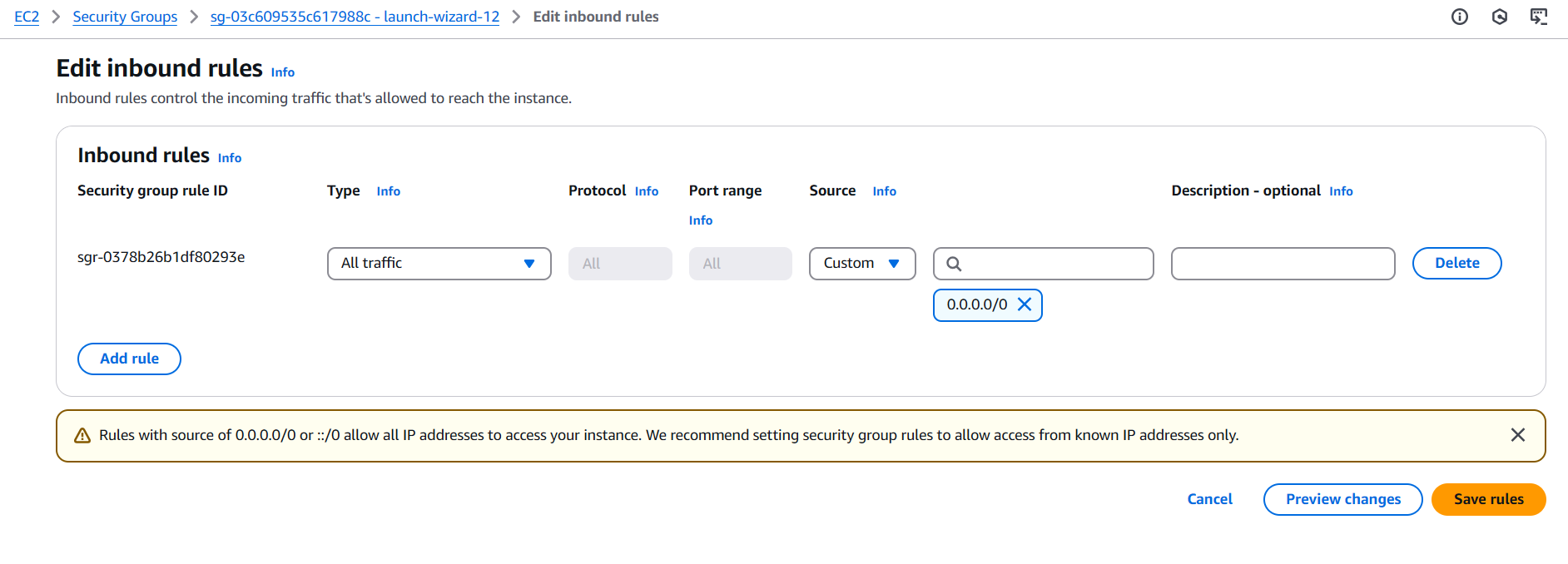
1. Goto EC2
2. Click on “Security Group”
3. Search for launch-wizard-12” and click on that



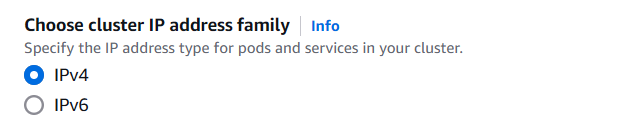
1. Once open : check for “Inbound Rule”



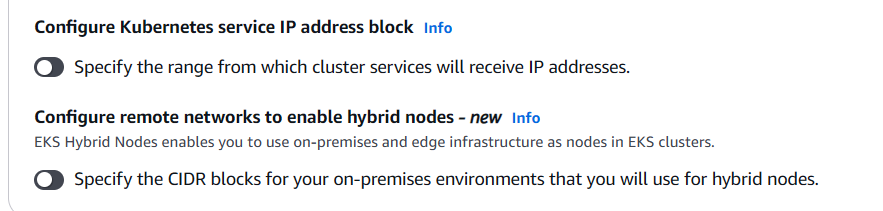
1. Select parameters as per below screen shot and save rules.



1. Choose cluster IP address family as IPv4

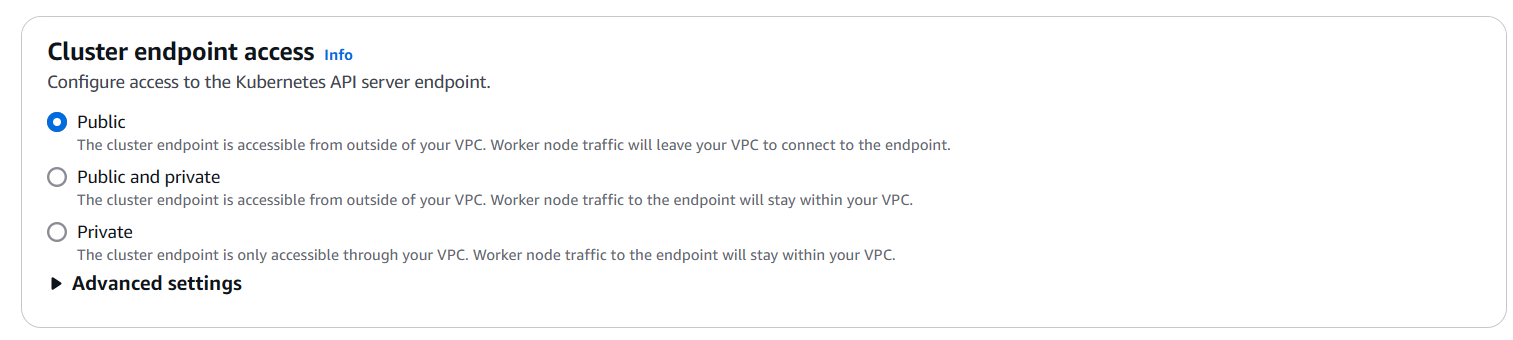


1. Don’t do anything on below



1. Cluster endpoint access:

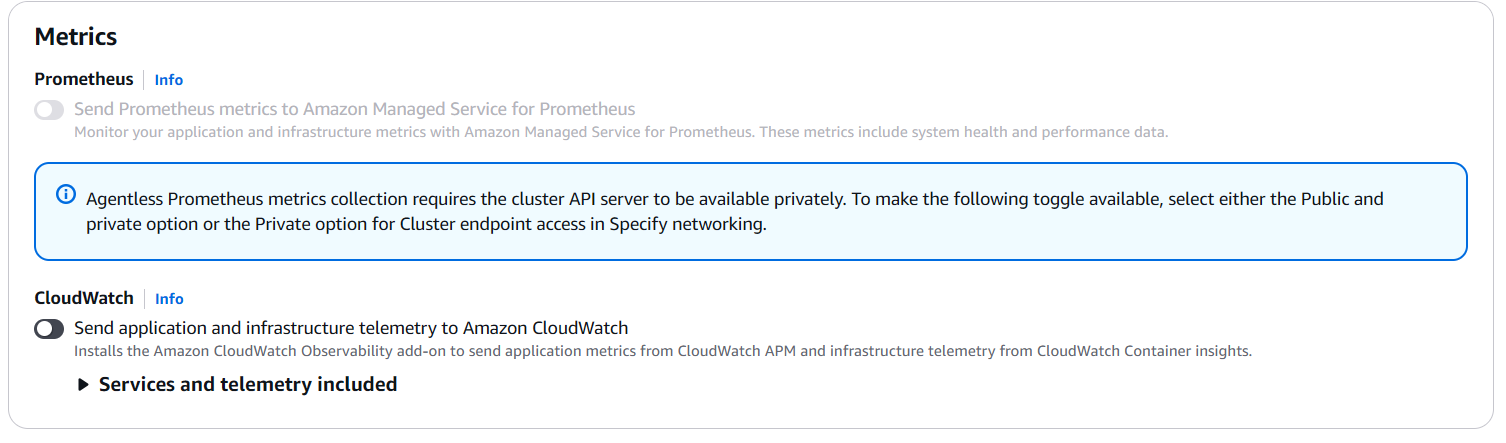
Select “Public” as cluster endpoint access.



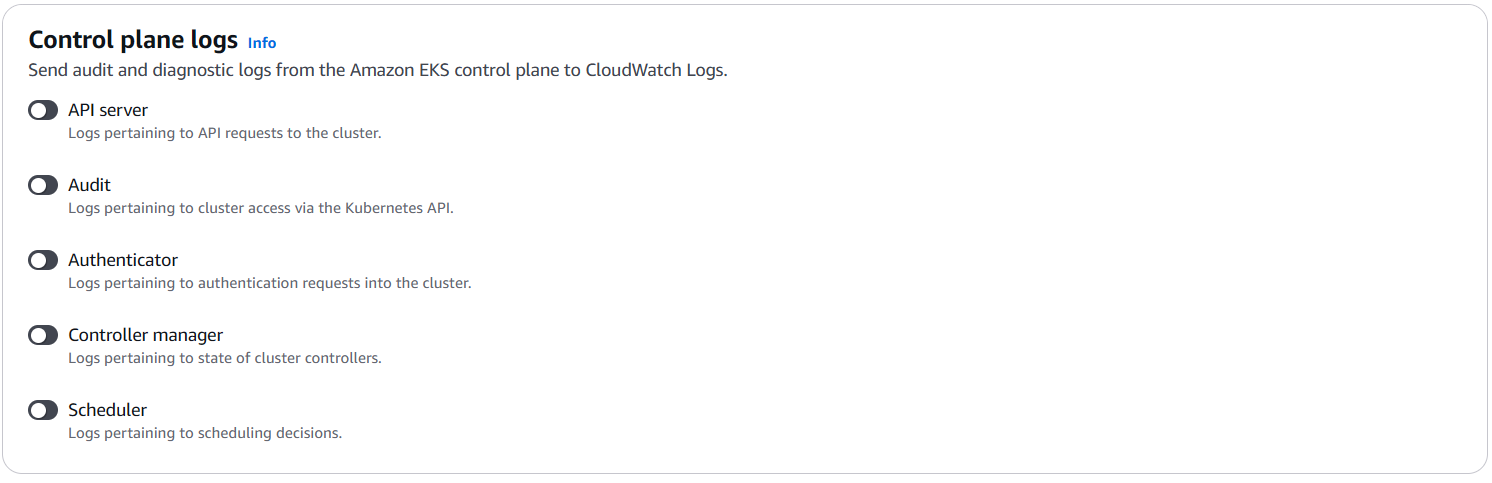
1. Click on “Next”



1. Don’t active any metrics for Promethus and cloudwatch



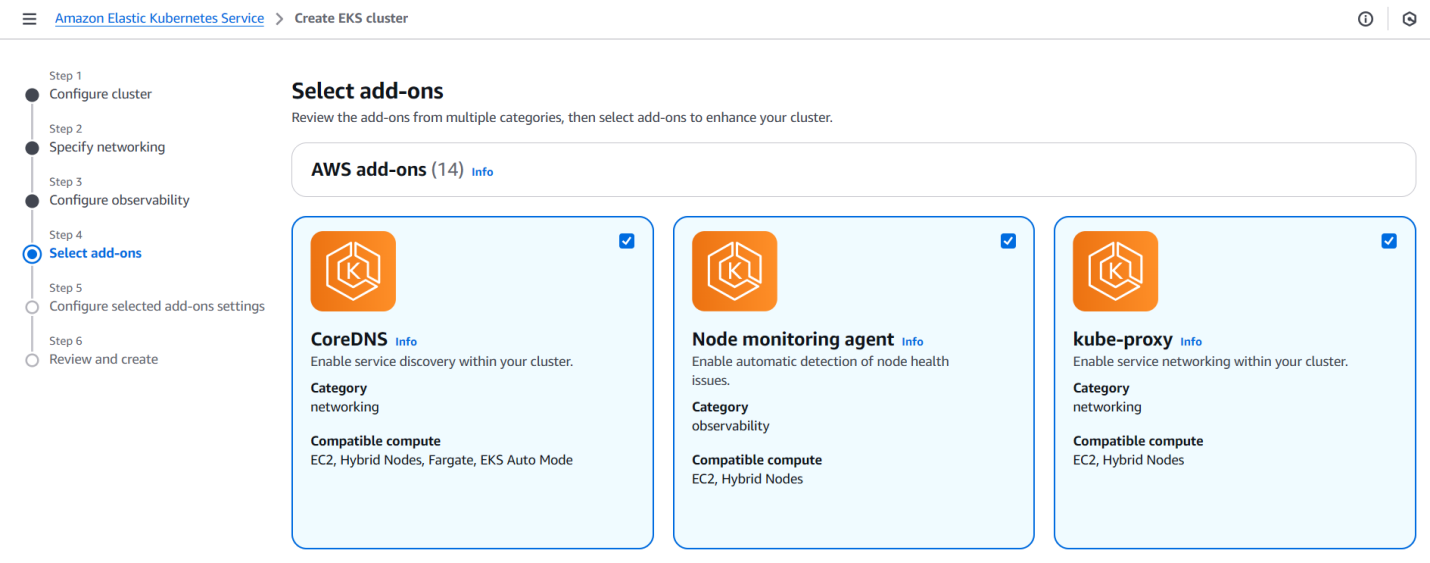
1. Don’t select or active any parameter for “Control plane log”



1. Click on “Next”

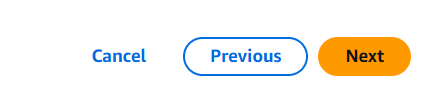


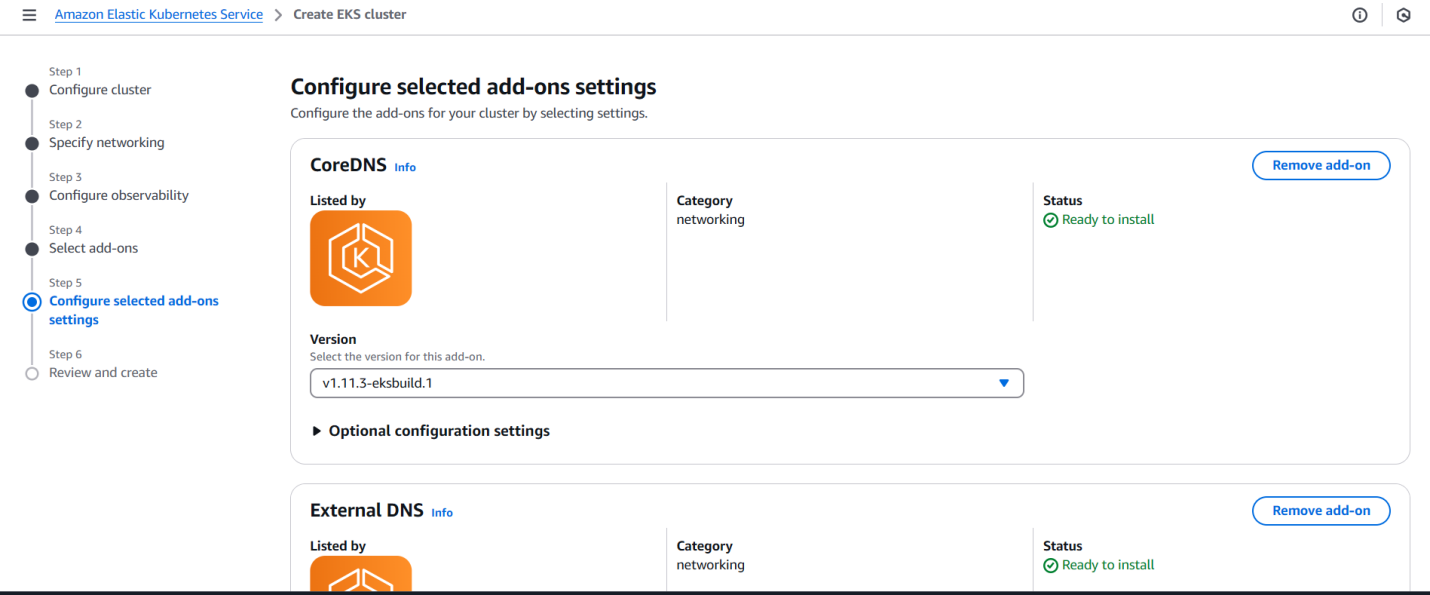
1. Select required “Add Ons”



Select additionally Cert Manager, **Amazon EFS CSI Driver, Amazon EBS CSI Driver**

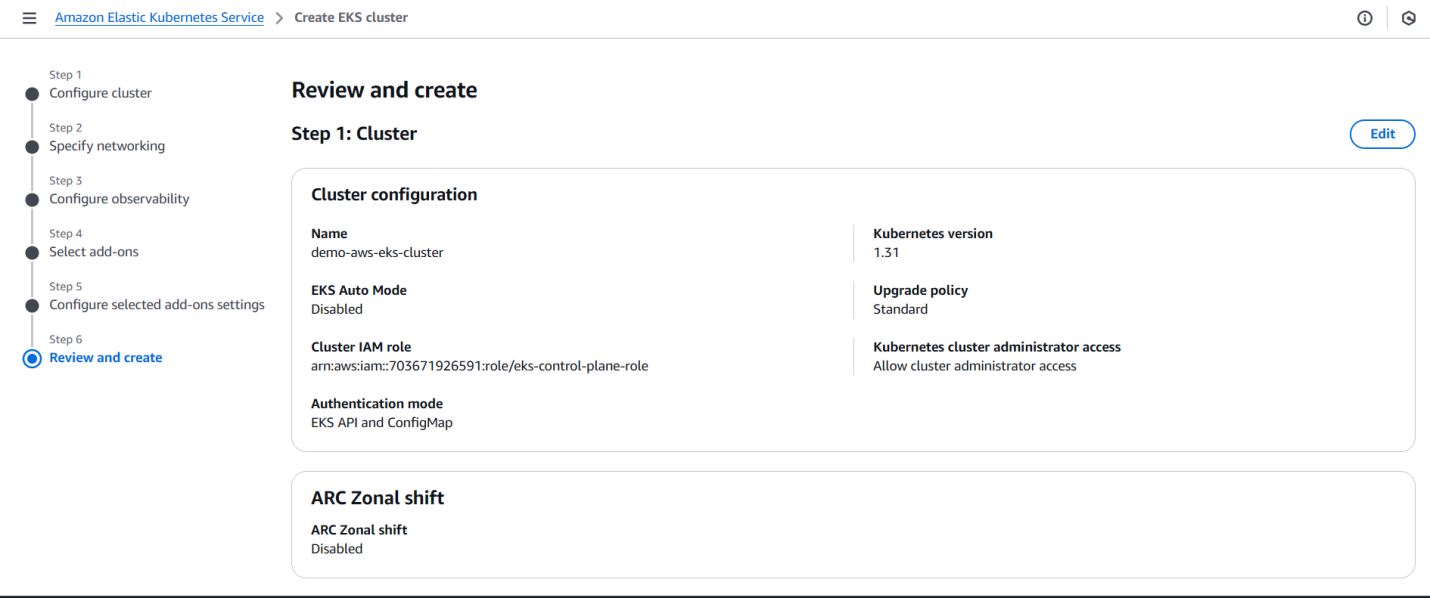
**click on Next**



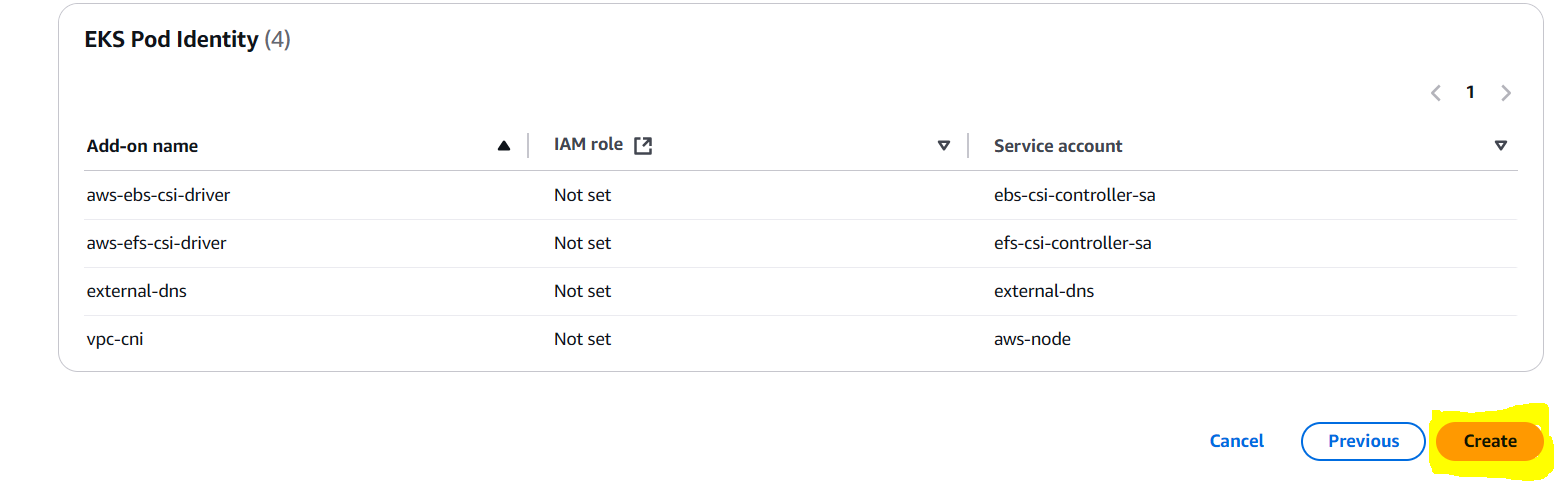


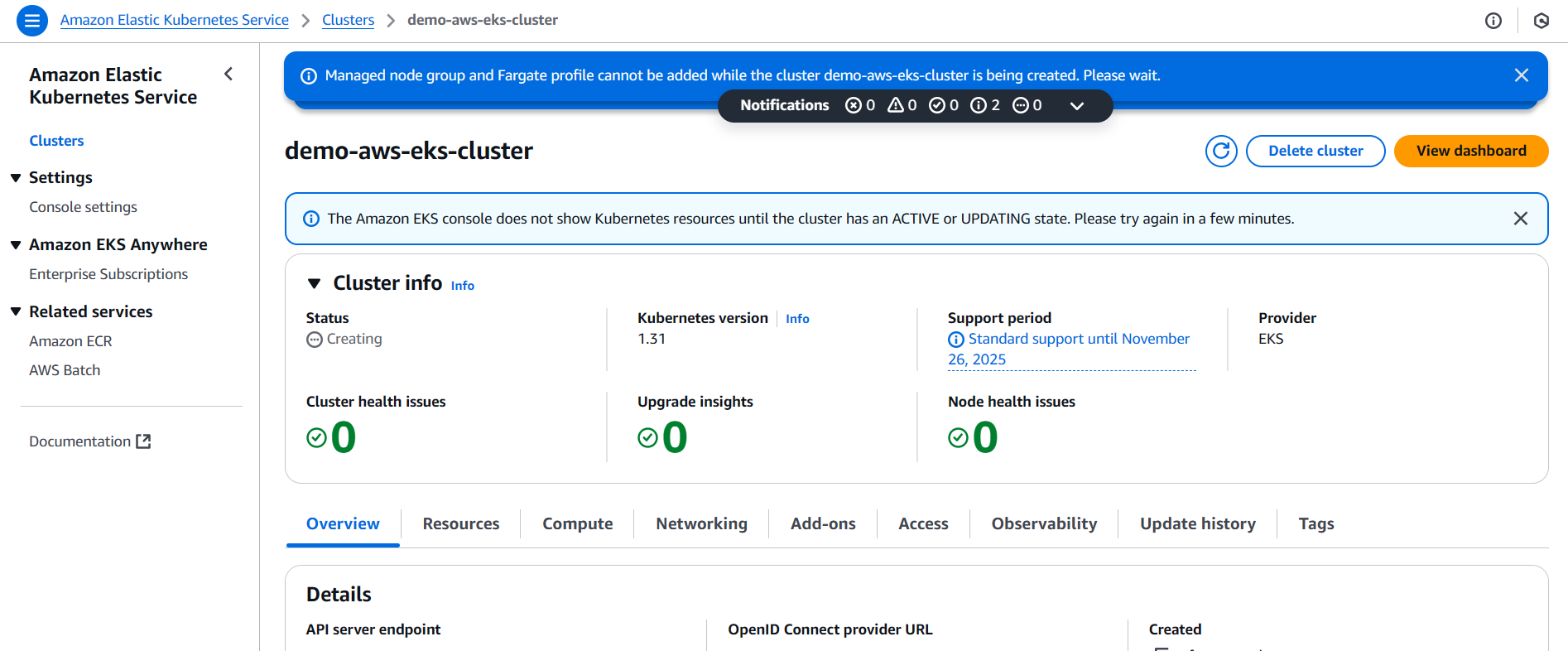
**Click on Next**





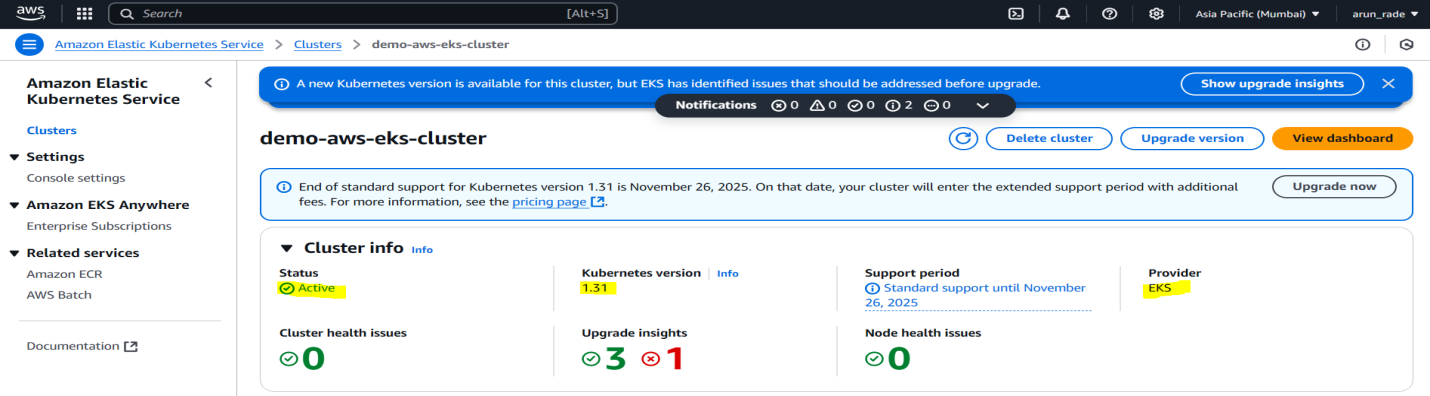
1. **Click on “Create”**





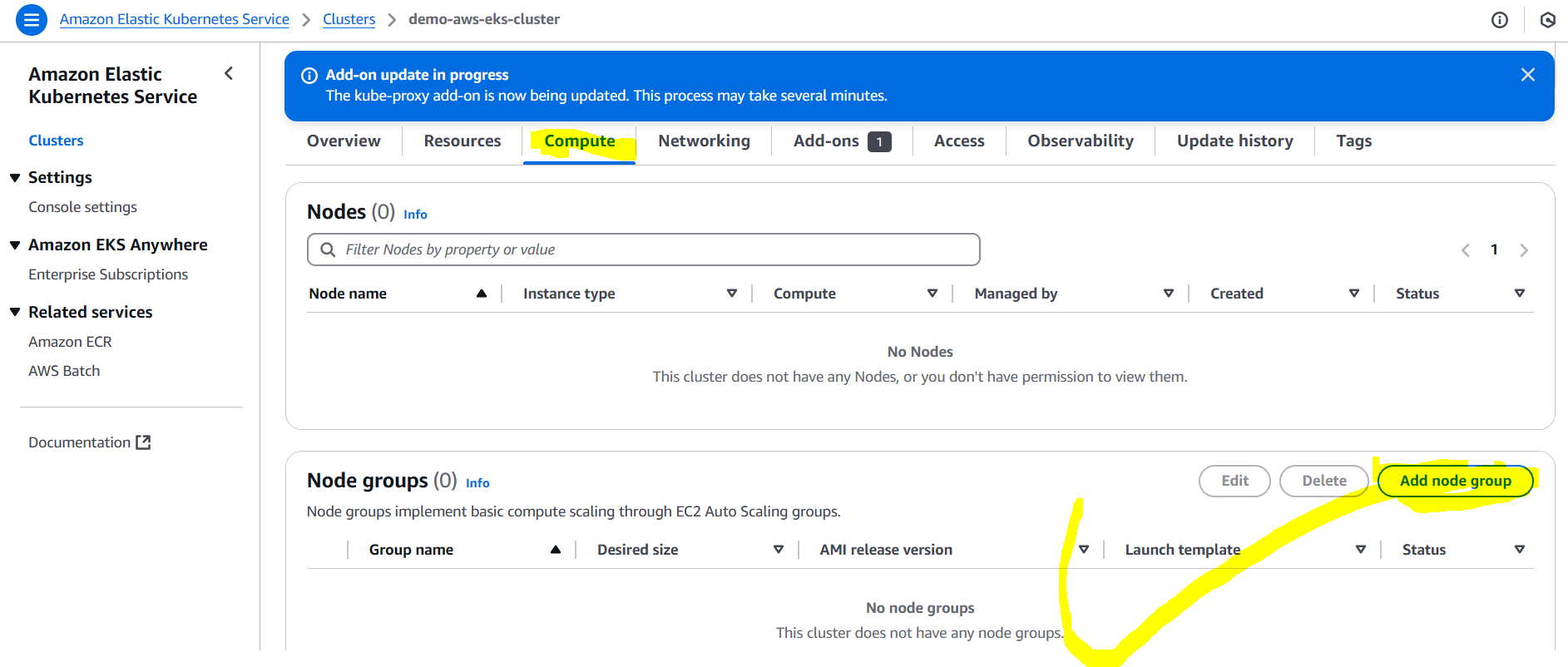
**Control plane created successfully**

**So, we can see status is “Active” after 5 to 10 min.**

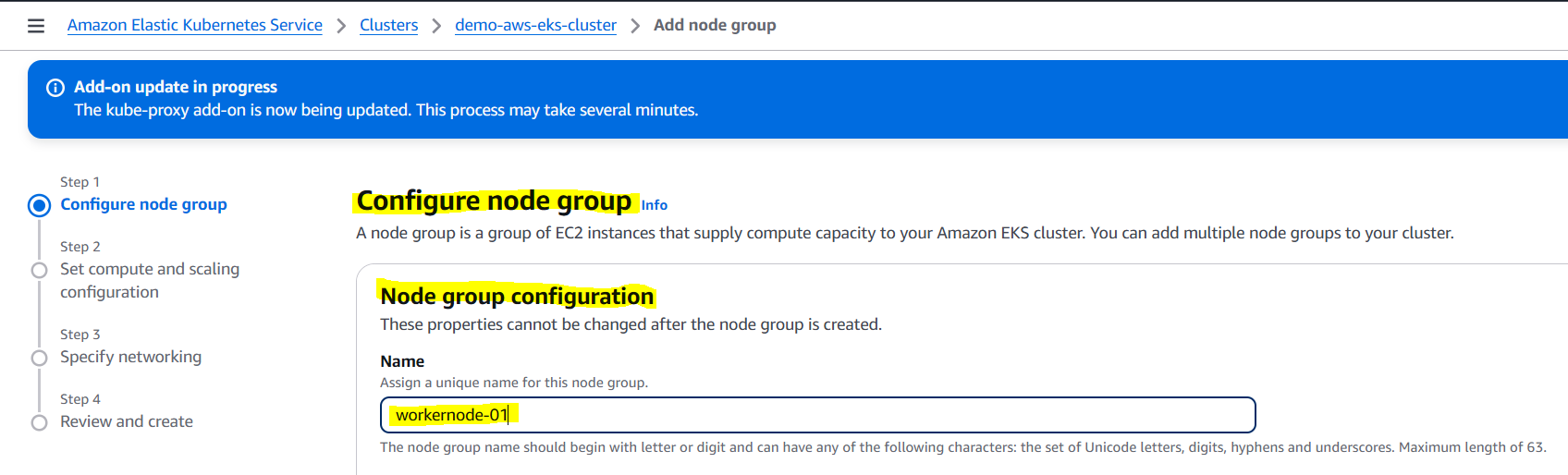


1. Now we need to create **Worker Node**

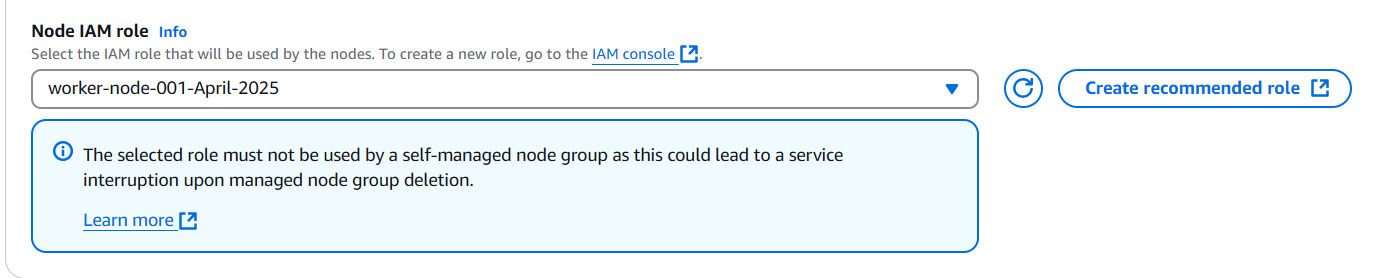
So that we need to click on “Compute” tab/button once your cluster is ready/active.



Click on “Add Node Group” to create new work node. So can see “Configure node group” window. And provide name of workgroup.

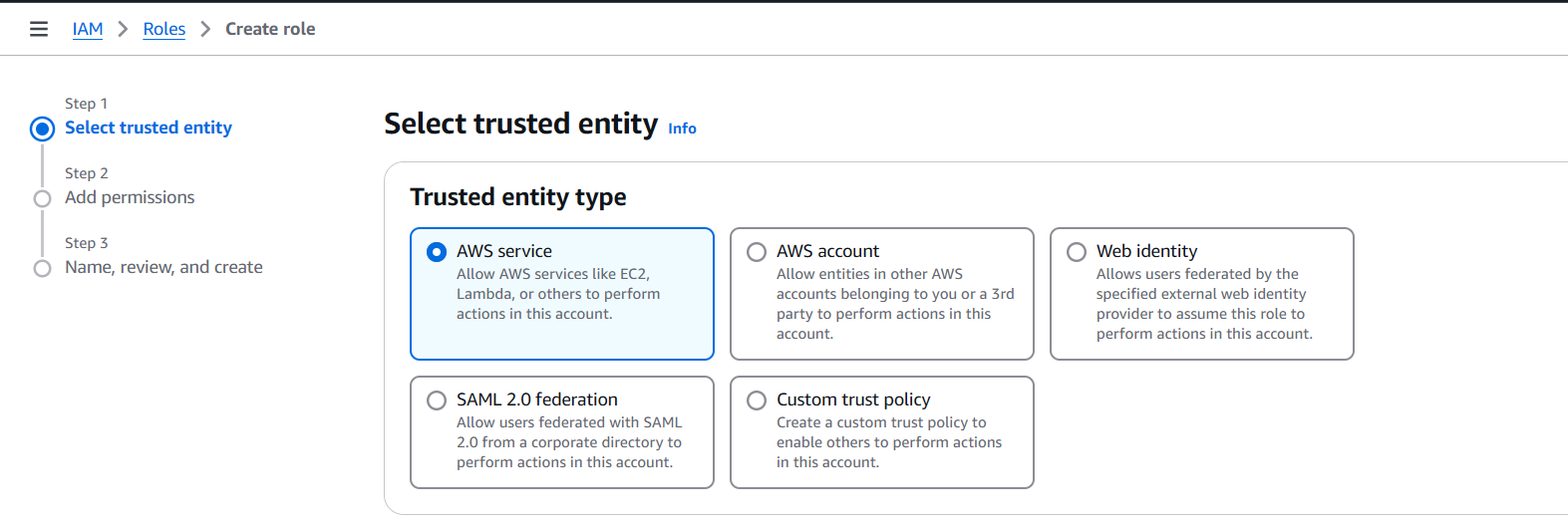


1. Need to create one more role for worker node

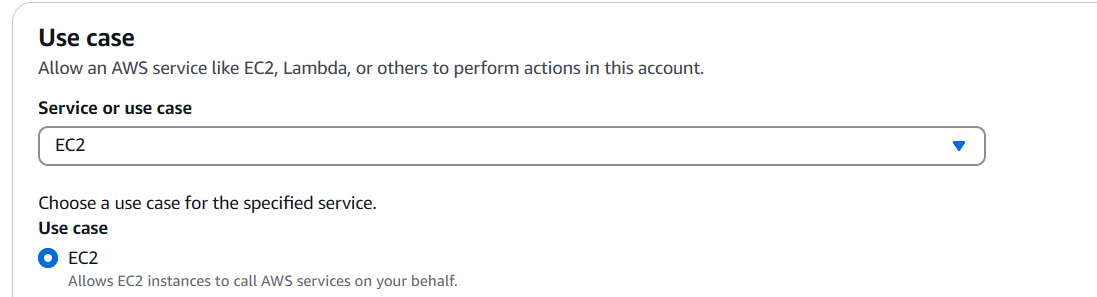


33a. Click on “Create recommended role” to create new role.

33b. Select “Trusted entity type” as “AWS Service.



33c. Use Case: Service or Use Case: EC2 and click on Next



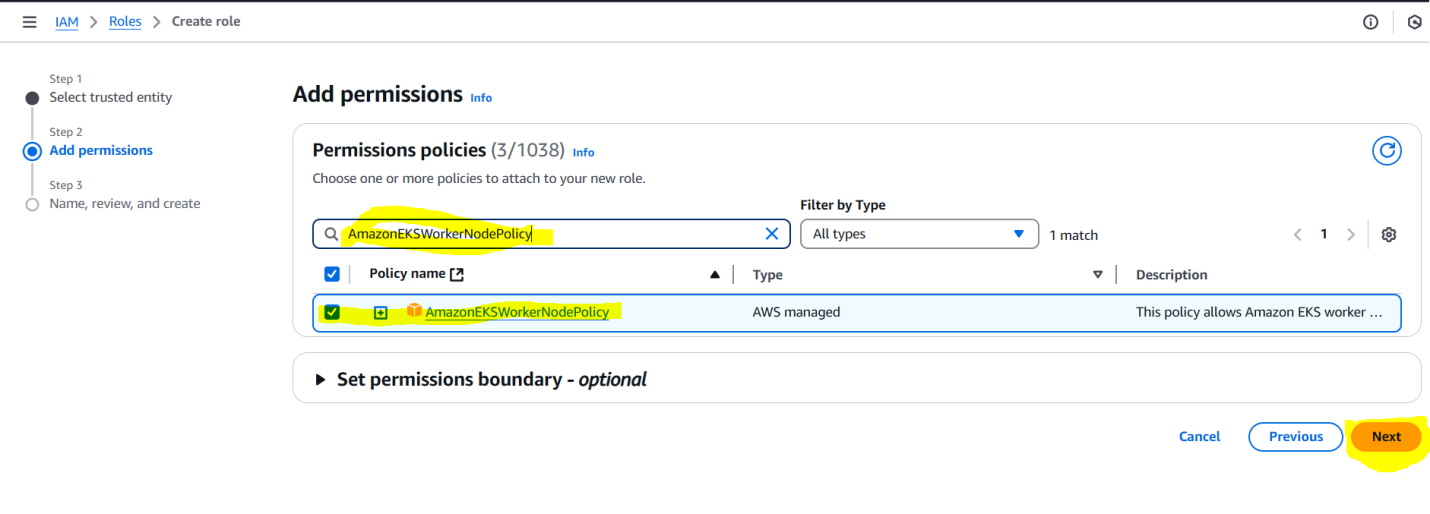


33d. now we need to permissions to create role for worker node

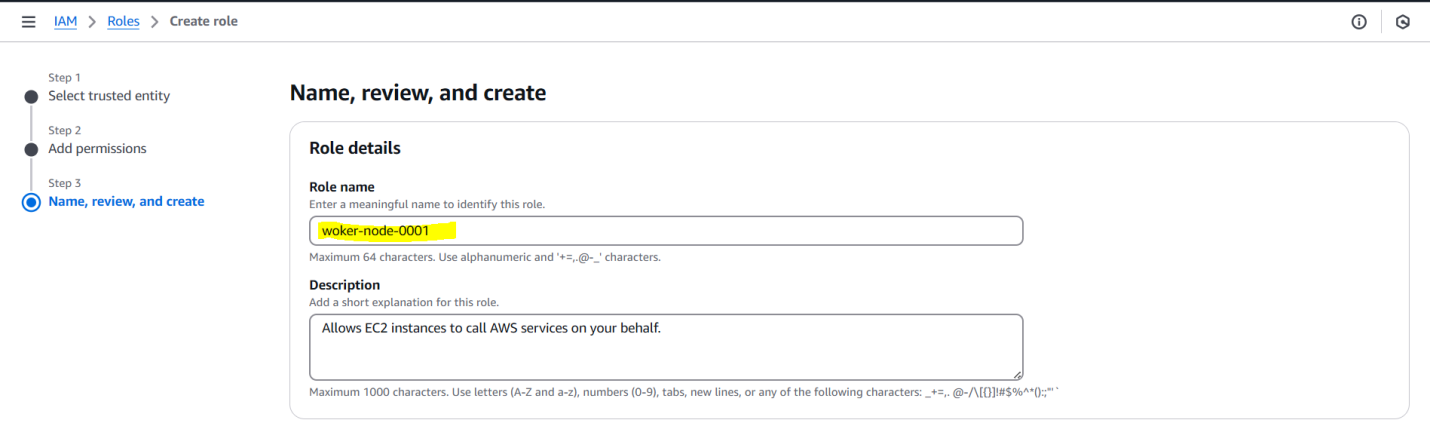
For that we need to add 3 permissions

1. AmazonEKS\_CNI\_Policy
2. AmazonEC2ContainerRegistryReadOnly
3. AmazonEKSWorkerNodePolicy

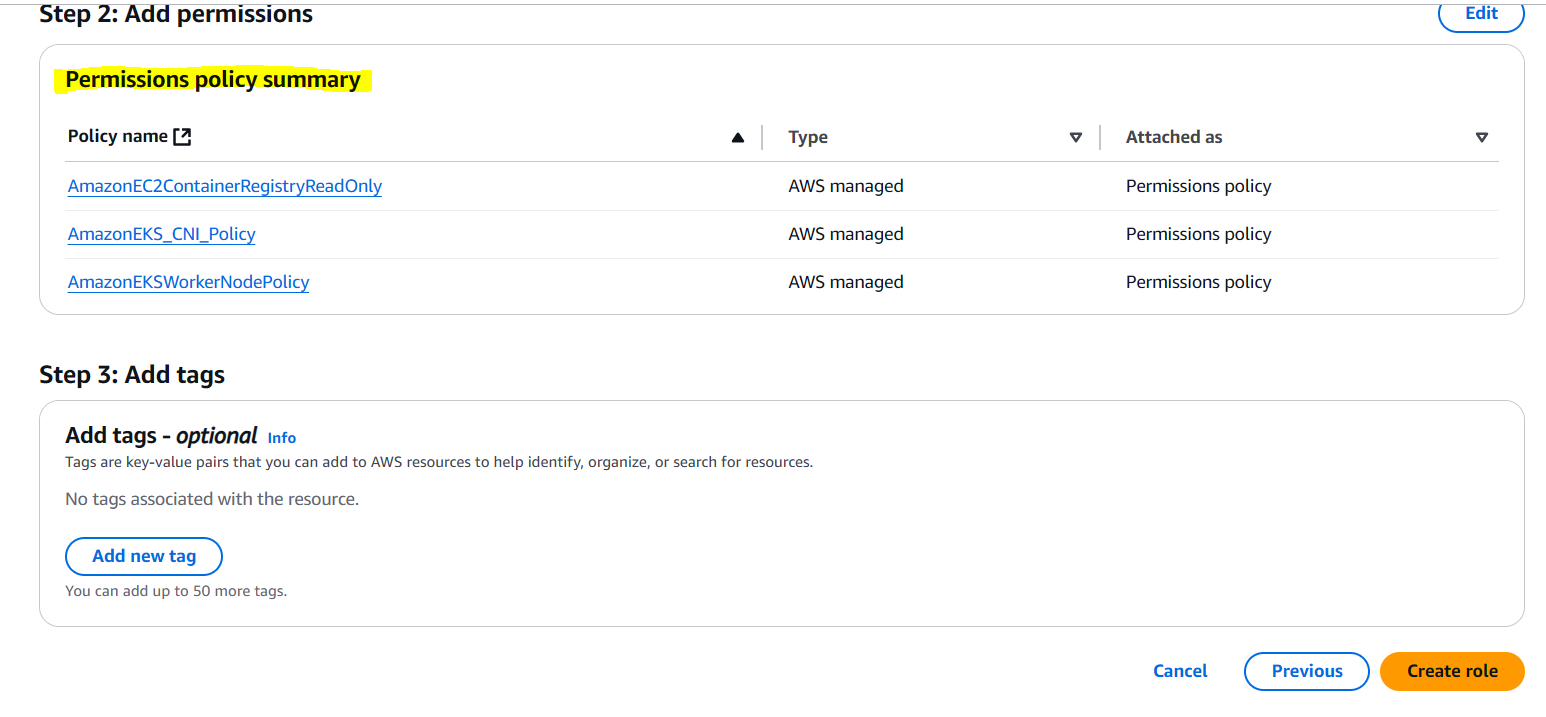
Search and add tick mark these policies and click on “Next”



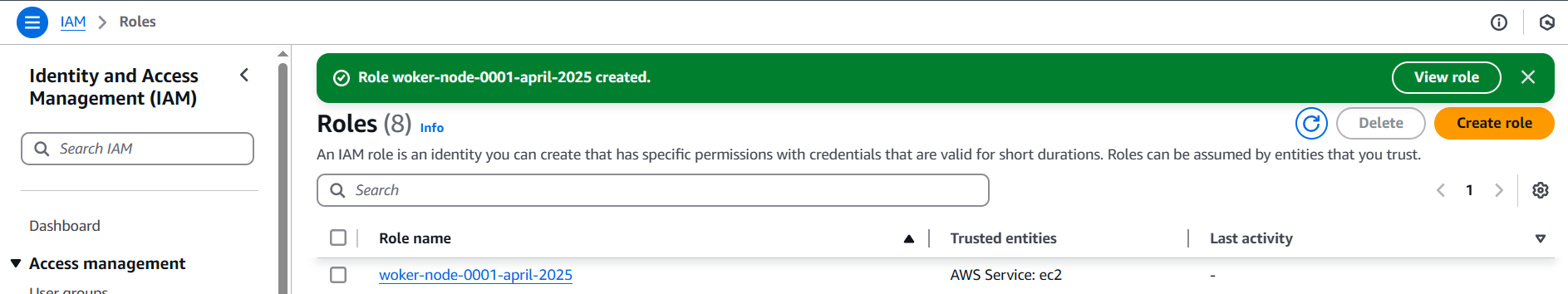
33e. Give name of worker\_Node:



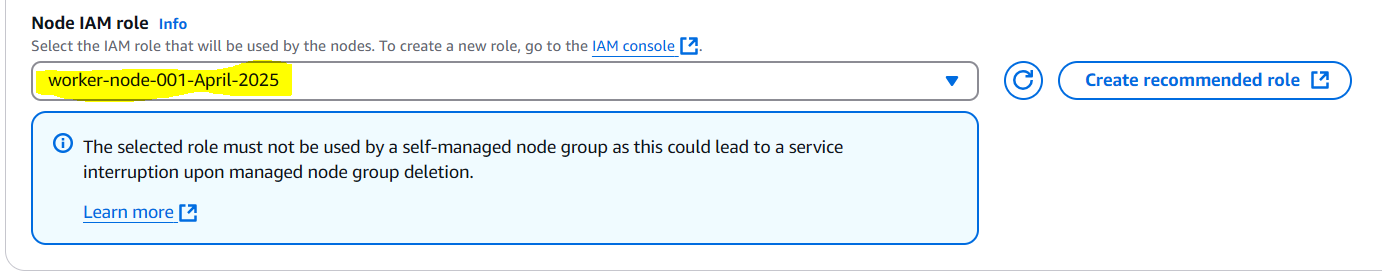
33f. You can see attached permissions under Add Permissions and click on “Create Role”



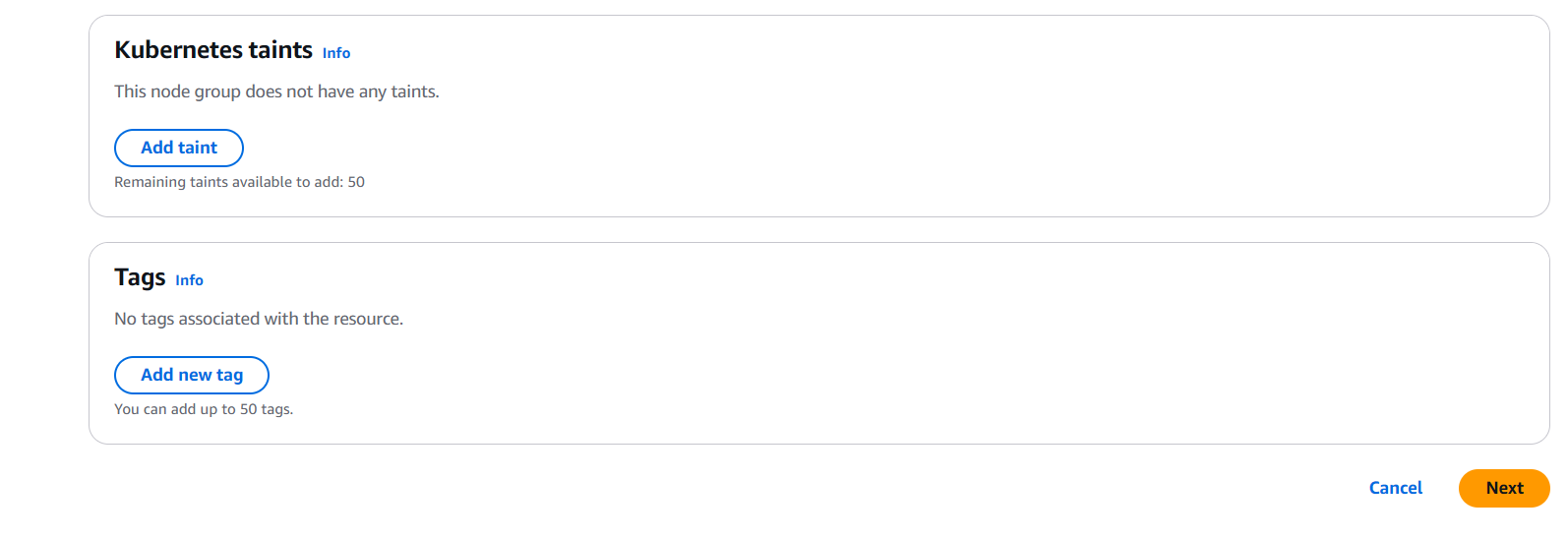
33g. You can see newly created role.



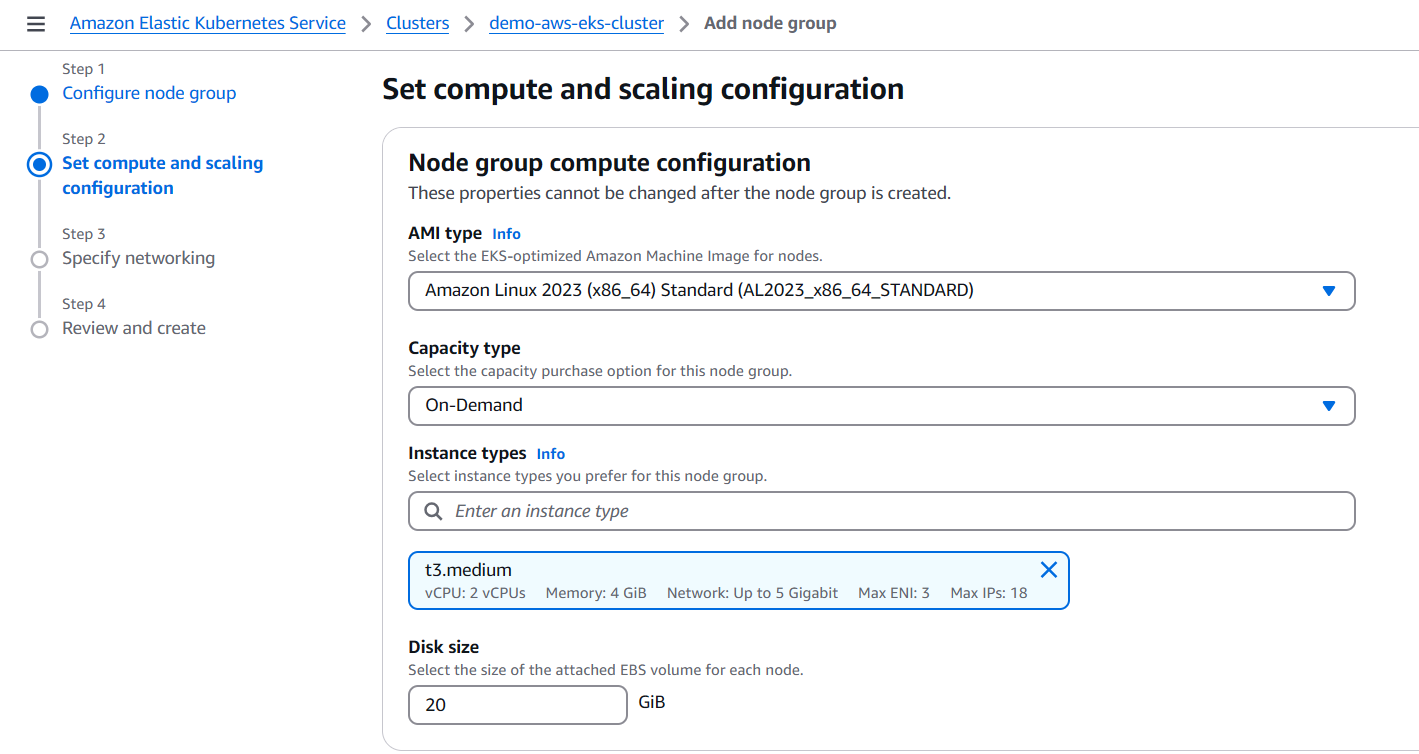
1. Select newly created role under Node IAM role



1. Don’t select any thing after that and directly click on “Next” button.



1. Set Compute and scaling Configuration: don’t amend anything under this.

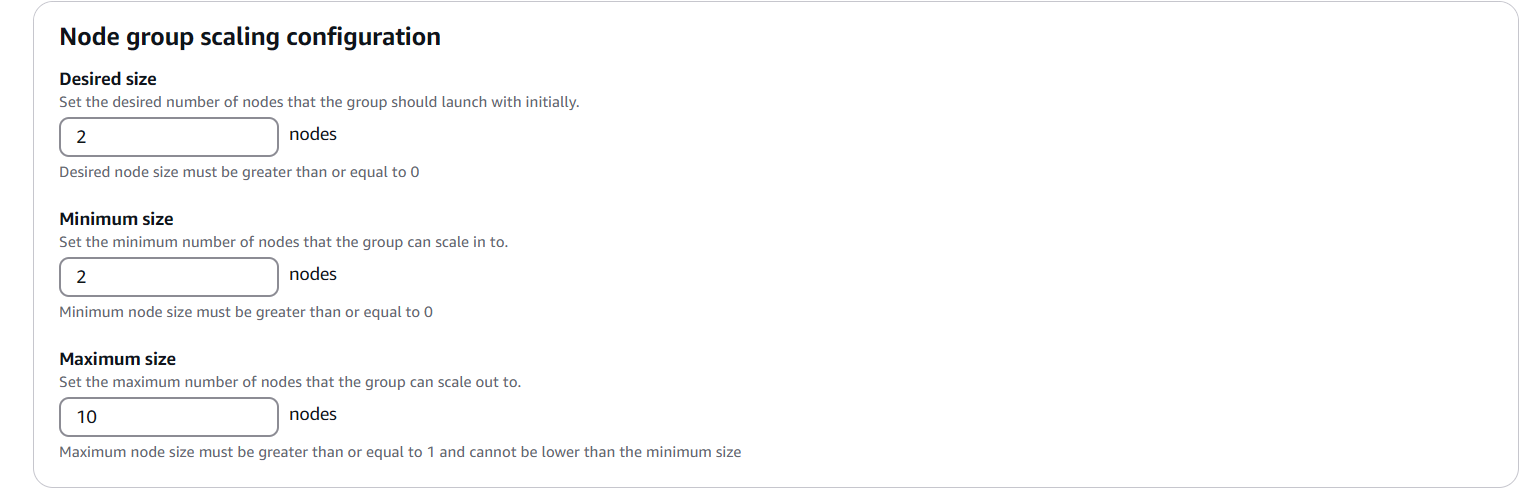


1. Node configuration

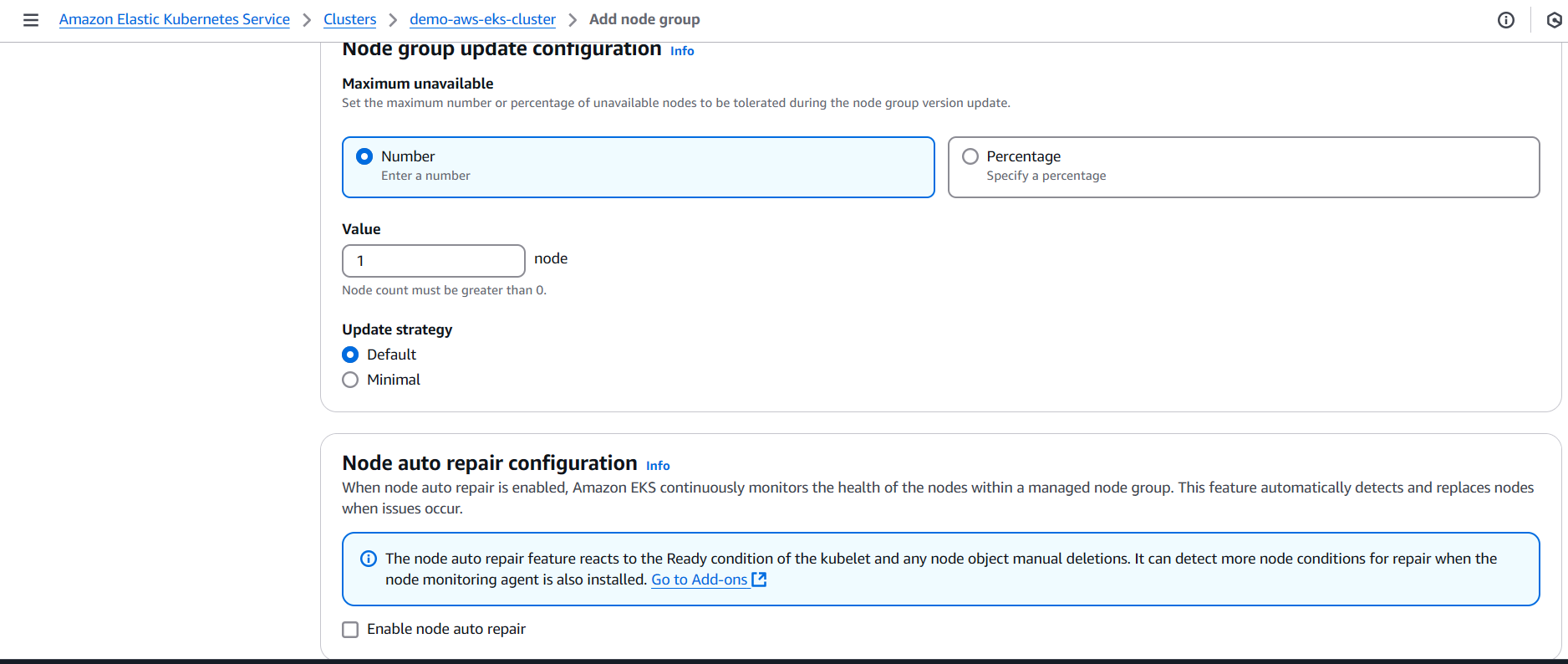
Desired size: 2

Minimum size: 2

Maximum size : 10



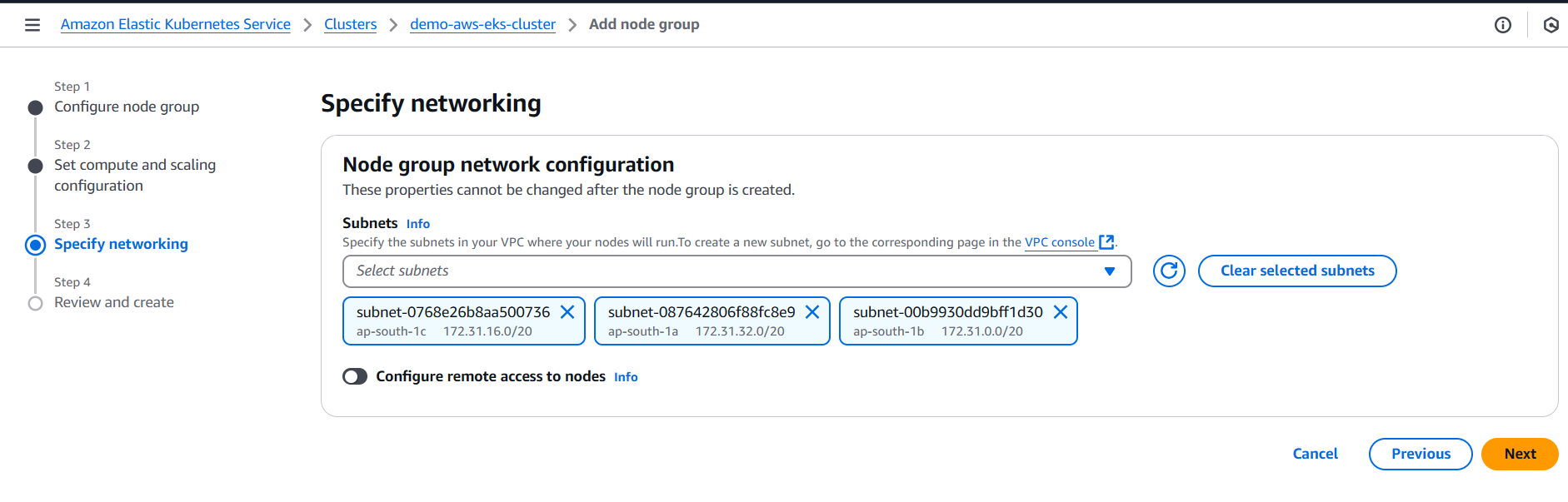
1. Node Group update Configuration: Don’t change anything

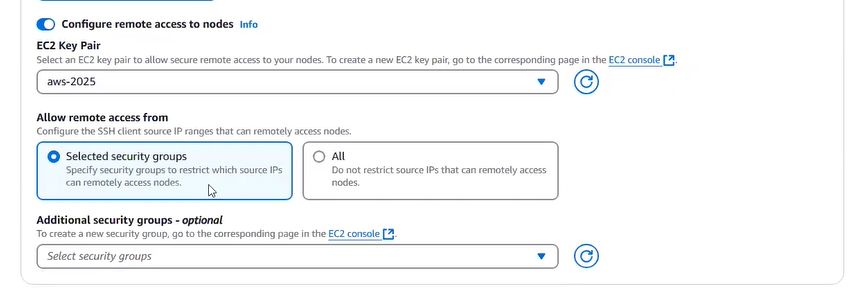


1. Click on Next



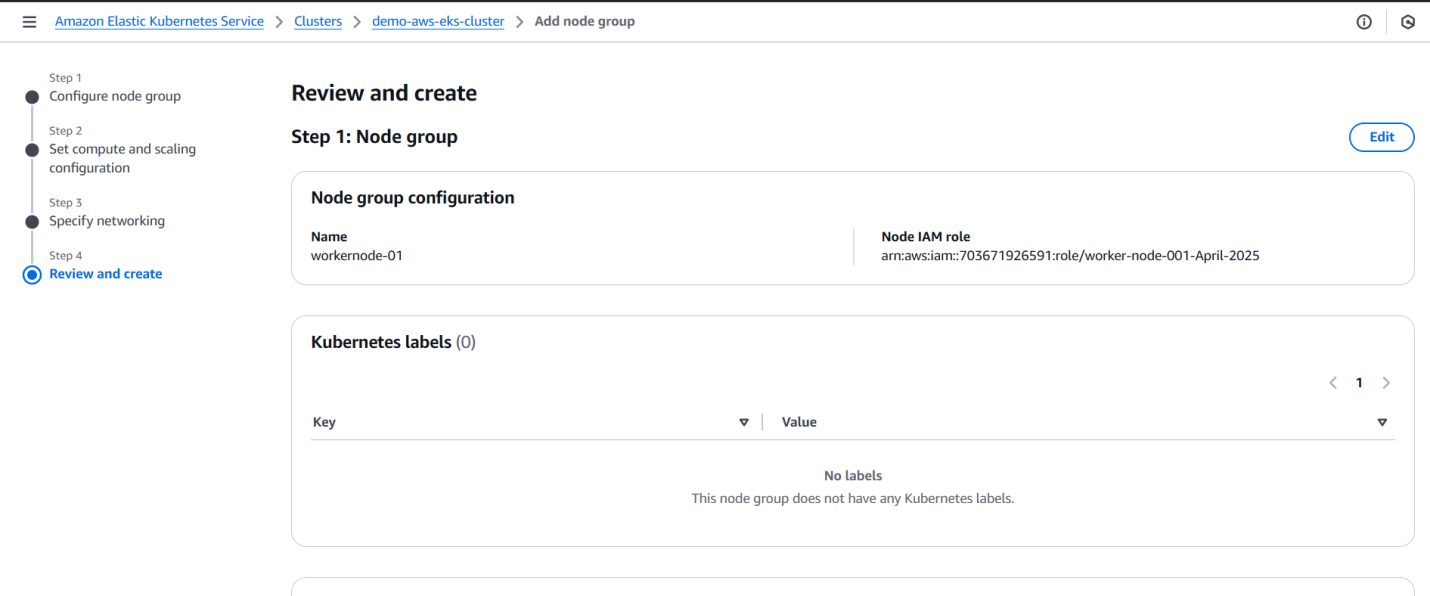
1. Specify Neworking: Don’t change anything and click on Next.

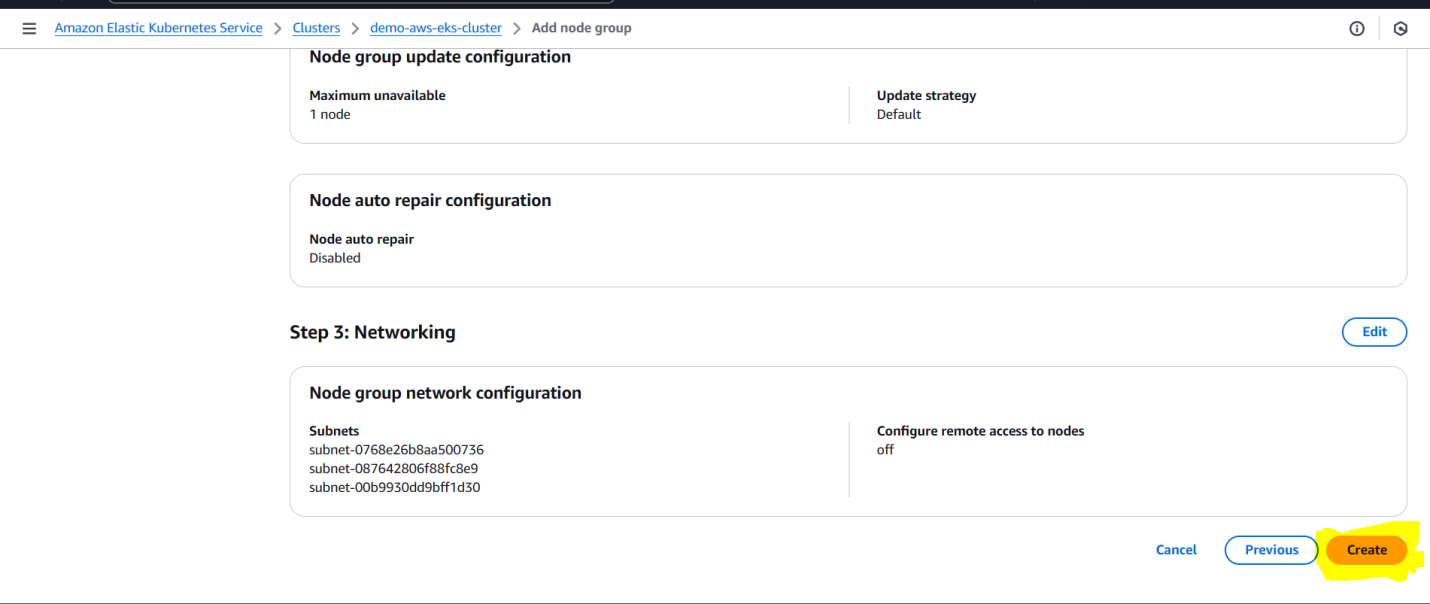


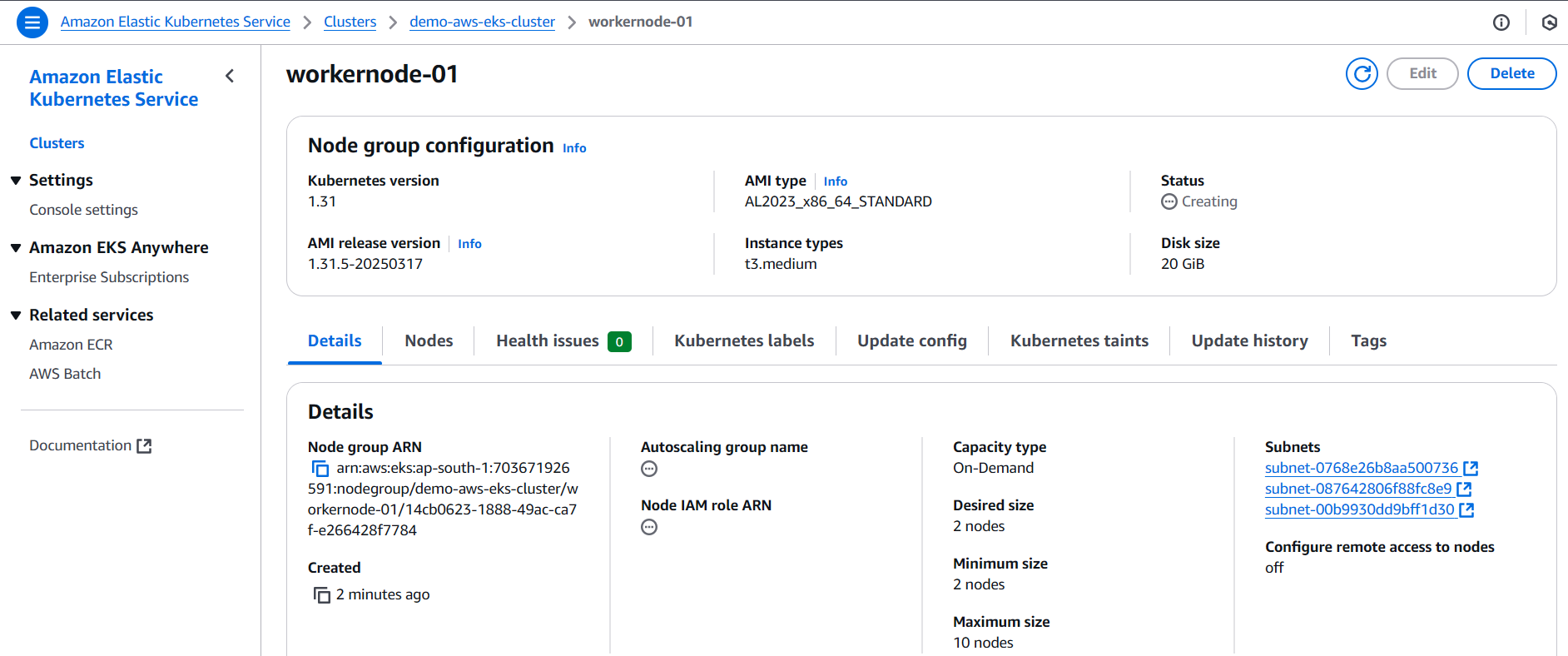


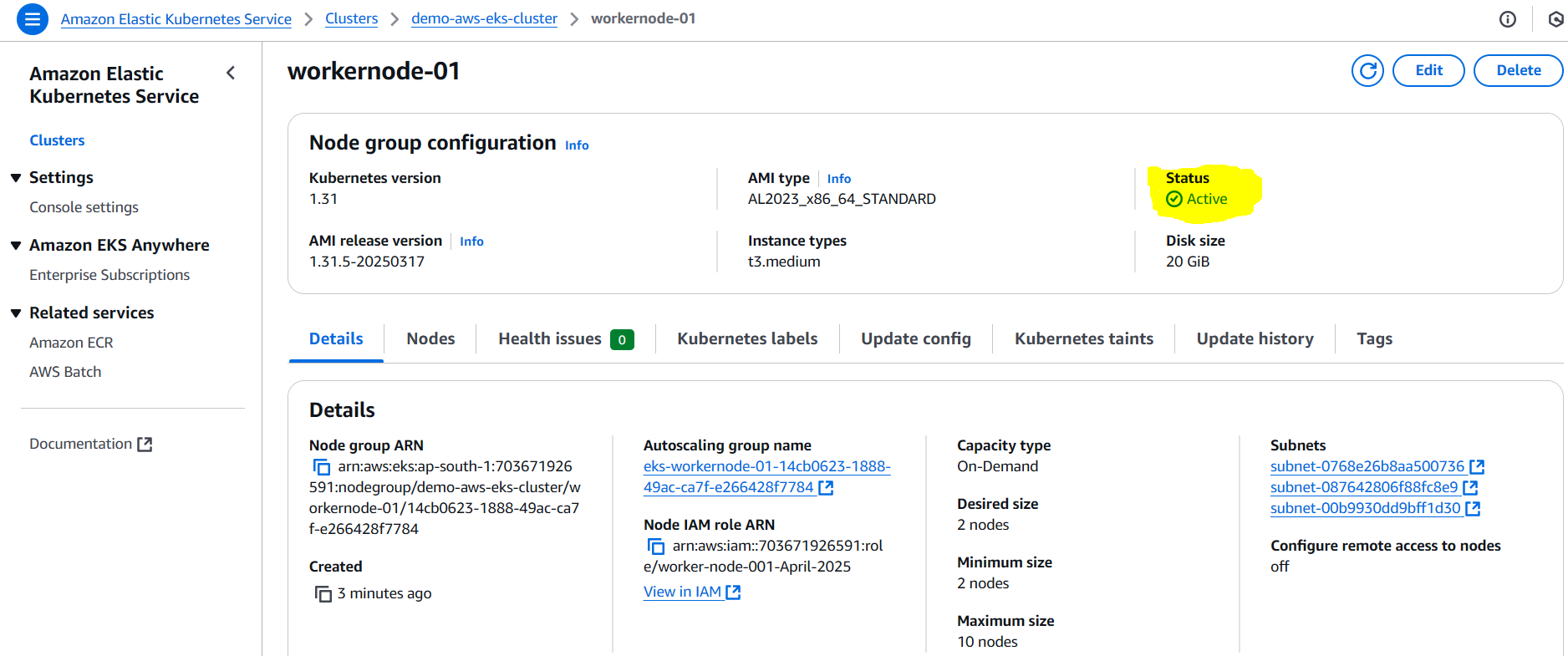
Select additional security group for above step.

1. Review and create

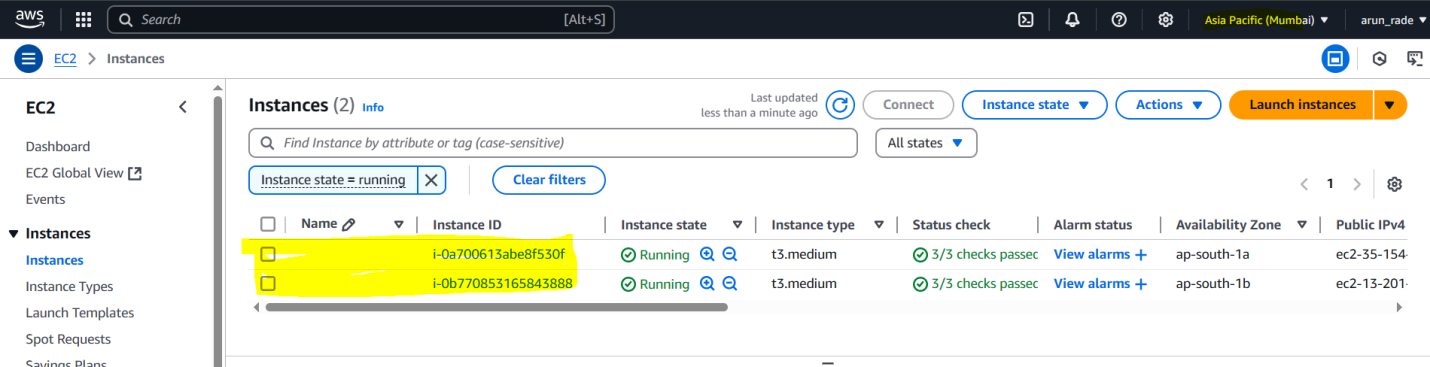








Once Worker Node gets created we can see 2 EC2 instances are created with t3.medium instance type.



Create one EC2 instance to connect Kubernetes cluster. t2.micro is ok.

Now we need to install Kubectl on newly created EC2 instance.

Login with root user i.e. sudo su –

1. Create a Linux VM to connect aws eks cluster .

login to cluster and turn root #sudo su -

2. Install kubectl

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

curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

chmod +x kubectl

mv kubectl /usr/local/bin/

check kubectl version with command kubectl version

[root@ip-172-31-15-11 ~]# kubectl version

Client Version: v1.32.3

Kustomize Version: v5.5.0

The connection to the server localhost:8080 was refused - did you specify the right host or port?

3. Install aws-iam-authenticator

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

curl -o aws-iam-authenticator https://amazon-eks.s3.us-west-2.amazonaws.com/1.15.10/2020-02-22/bin/linux/amd64/aws-iam-authenticator

chmod +x ./aws-iam-authenticator

sudo mv ./aws-iam-authenticator /usr/local/bin

aws-iam-authenticator version

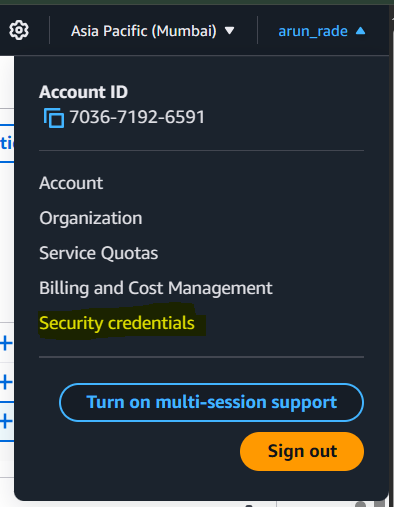
4. upgrade aws-cli

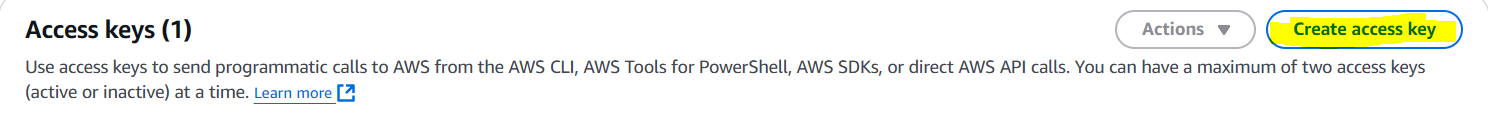
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

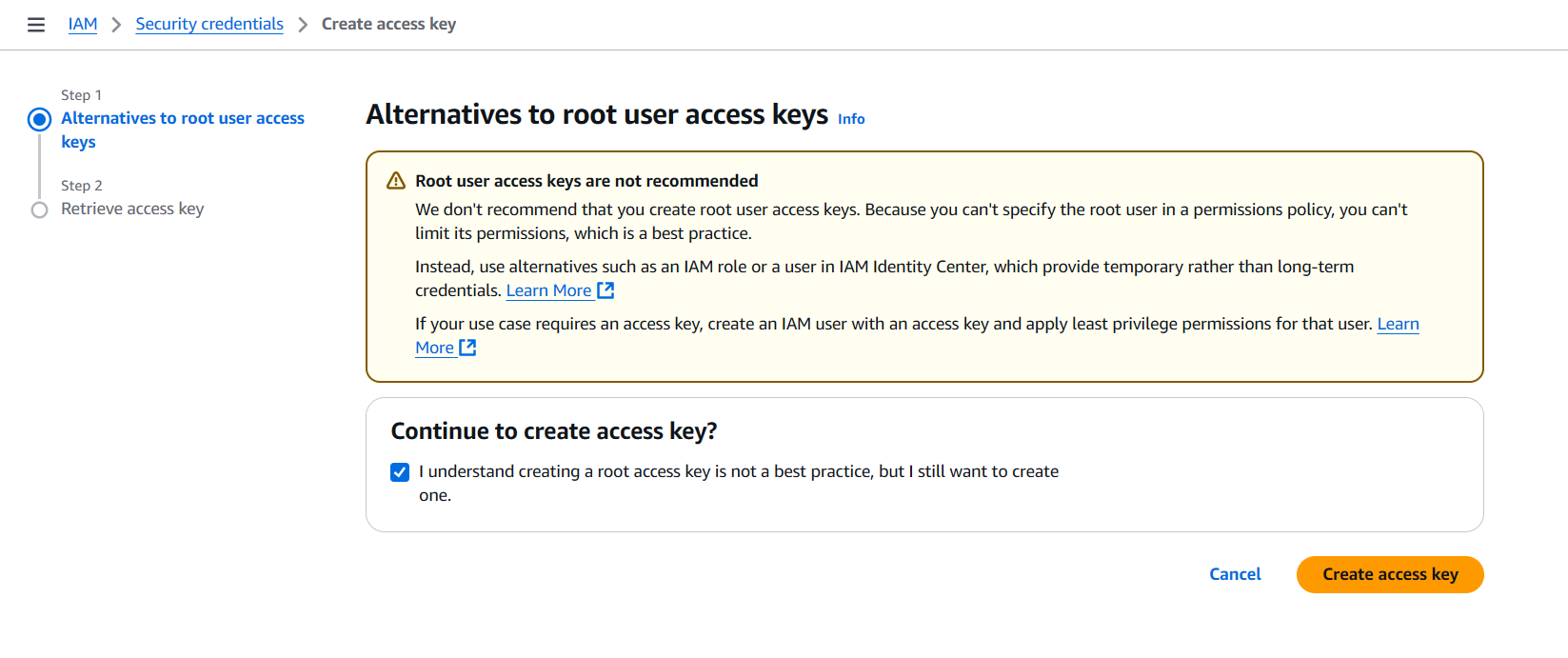
unzip awscliv2.zip

sudo ./aws/install

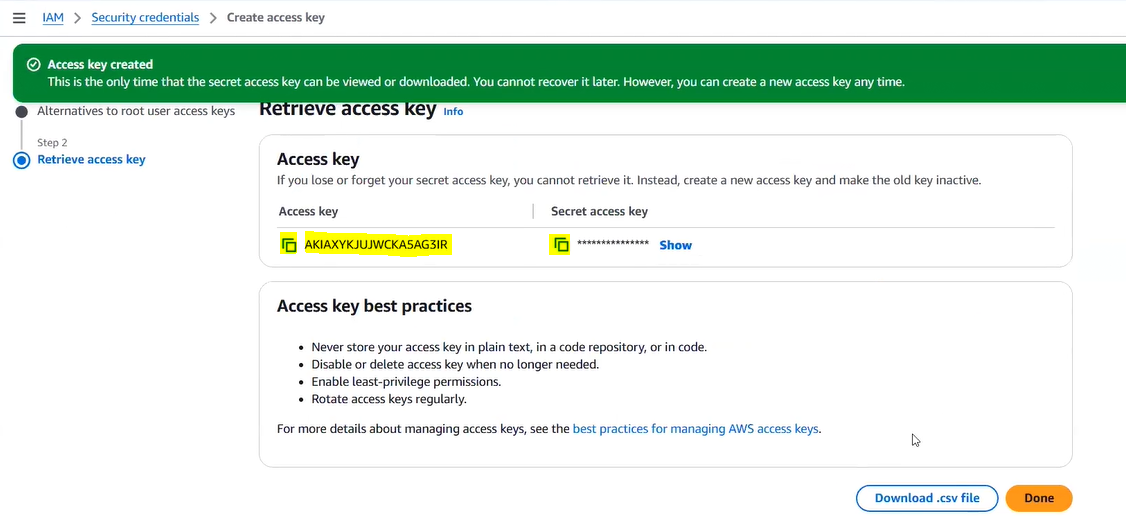
1. create root used access key and secret accesskey and configure using #aws configure







Click on “Create Access Key” and copy both keys and provide in below command.



[root@ip-172-31-15-11 ~]# **aws configure**

AWS Access Key ID [None]: AKIA2HVQ5P474LLNETDS

AWS Secret Access Key [None]: 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

Default region name [None]: ap-south-1

Default output format [None]: json

6. #aws sts get-caller-identity

[root@ip-172-31-15-11 ~]# aws sts get-caller-identity

{

"UserId": "703671926591",

"Account": "703671926591",

"Arn": "arn:aws:iam::703671926591:root"

}

7 #aws eks update-kubeconfig --region ap-south-1 --name fusion-k8s-cluster

**[root@ip-172-31-15-11 ~]# aws eks update-kubeconfig --region ap-south-1 --name demo-aws-eks-cluster**

**Added new context arn:aws:eks:ap-south-1:703671926591:cluster/demo-aws-eks-cluster to /root/.kube/config**

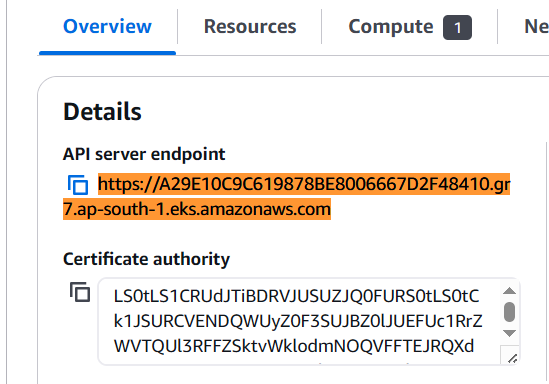
8 open /root/.kube/config and update #apiVersion: client.authentication.k8s.io/v1 and #interactiveMode: IfAvailable --- **for troubleshoot purpose only**

Kubeconfig.yaml

Get below certificate x.509 from cat /root/.kube/config file

<https://A29E10C9C619878BE8006667D2F48410.gr7.ap-south-1.eks.amazonaws.com>

and match with cluster certificate/API Server endpoint.



9. **kubectl get node**.

[root@ip-172-31-15-11 ~]# kubectl get node

NAME STATUS ROLES AGE VERSION

ip-172-31-29-59.ap-south-1.compute.internal Ready <none> 40m v1.31.5-eks-5d632ec

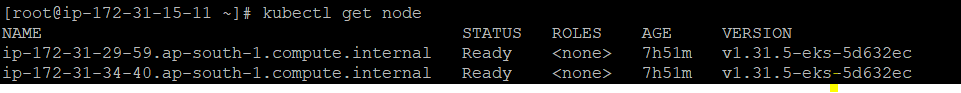
ip-172-31-34-40.ap-south-1.compute.internal Ready <none> 40m v1.31.5-eks-5d632ec

[root@ip-172-31-15-11 ~]#

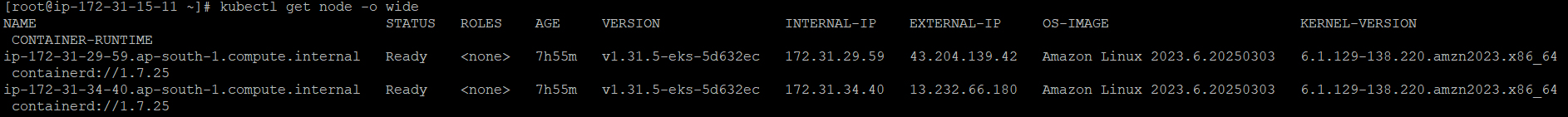
Kubernetes commands:

* + - 1. to get the list of nodes present on cluster

command: kubectl get node

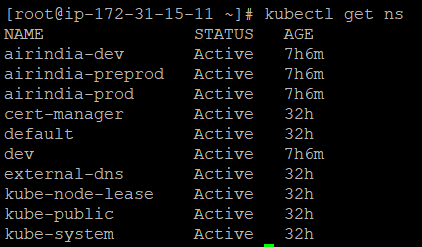


* + - 1. To get the list of nodes with all details



* + - 1. To get the list on Namespace on cluster

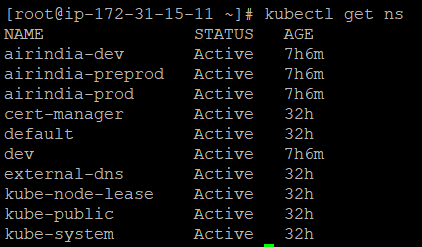
Kubectl get ns



* + - 1. How to create the namespaces

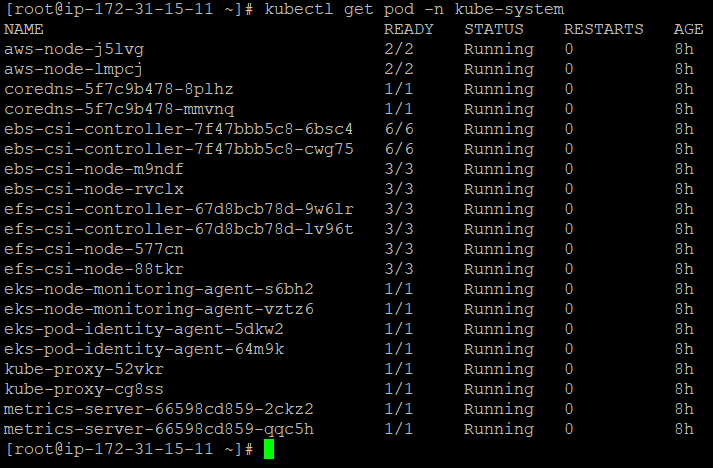
Kubectl create ns <namespce\_name>





1. To get the list of pod’s in kube-system namespace

Kubectl get pod -n kube-system



There are two types of creating Kubernetes objects.

* + - 1. Imperative

=========Imperative Method=============

#kubectl create ns prod

#kubectl create ns dev1

* + - 1. Declarative

=============Declarative=================

apiVersion: v1

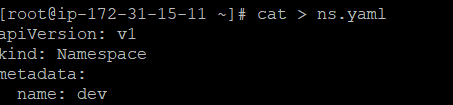
kind: Namespace

metadata:

name: dev

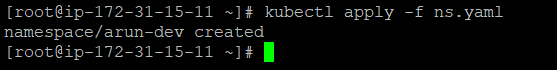
For Declarative method, we need to create first .yaml file and execute that file

e.g.



Run below command to create the namespace

kubectl apply -f ns.yaml



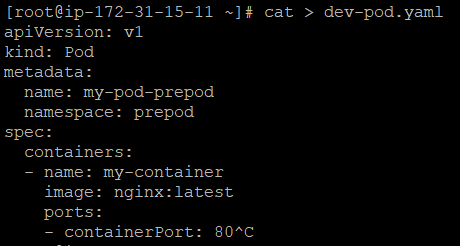
How to create POD’s:

Imperative Method:

kubectl run mypod-dev --image=nginx --namespace=dev

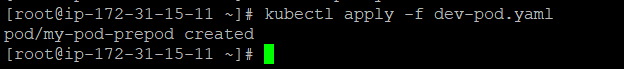


Declarative Method:



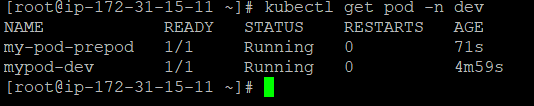
To run the file. Execute below command.

kubectl apply -f dev-pod.yaml



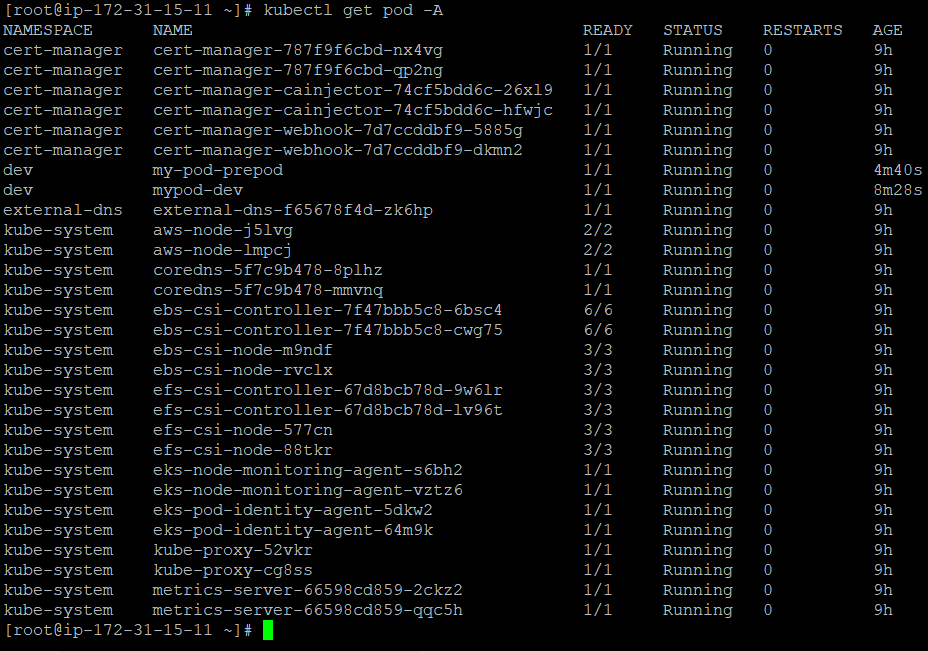
To check how many pod’s present under specific namespace.

kubectl get pod -n dev



Check all the pod’s on cluster

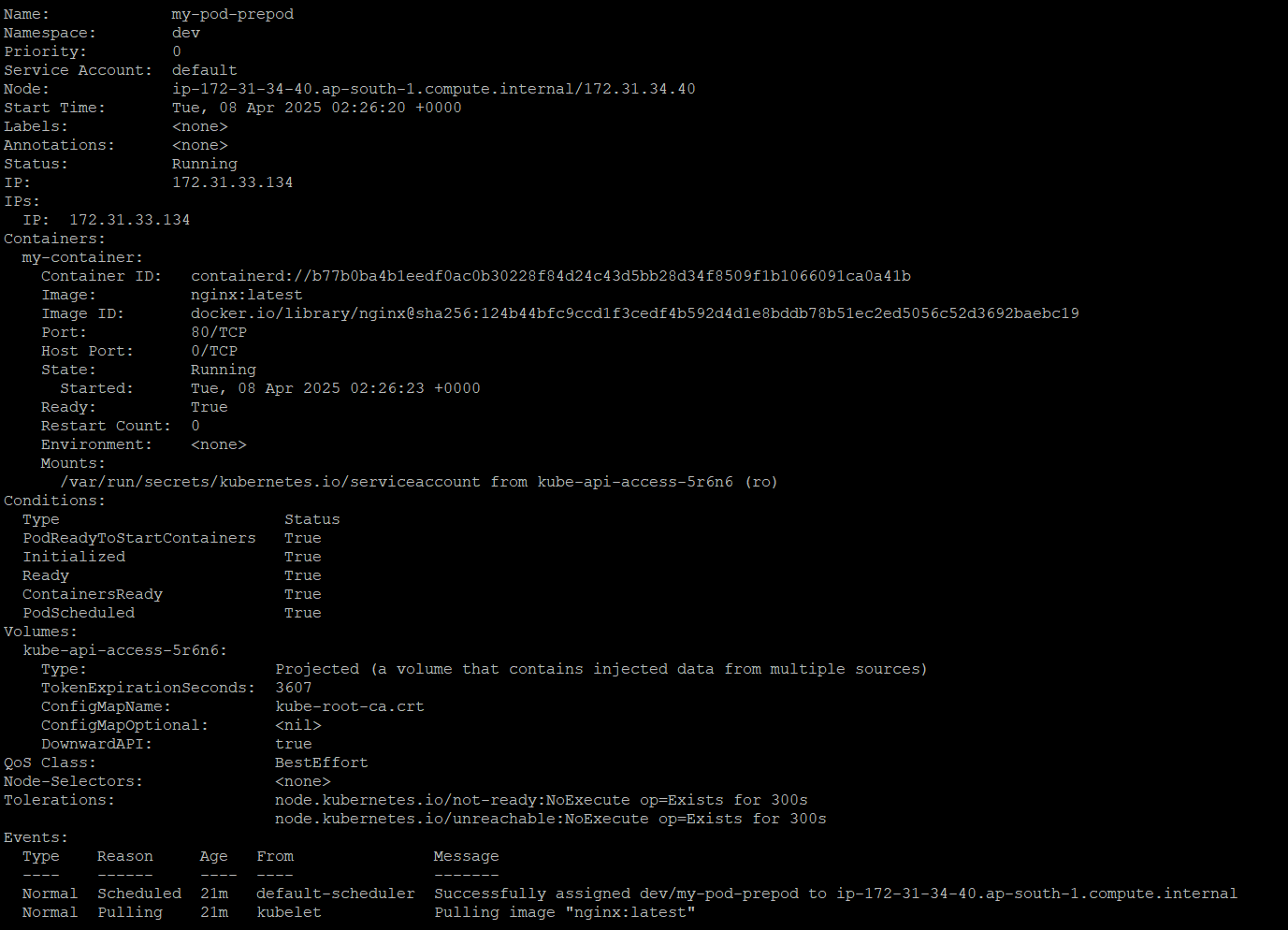
kubectl get pod -A

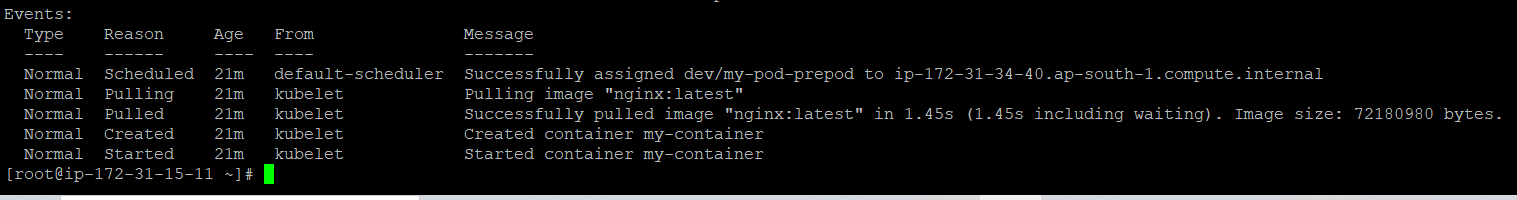


How to get all details for any pod.

kubectl describe pod ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system

kubectl describe pod my-pod-prepod -n dev

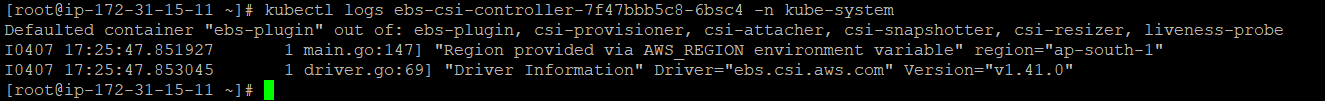




How to check logs for pod

kubectl logs ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system

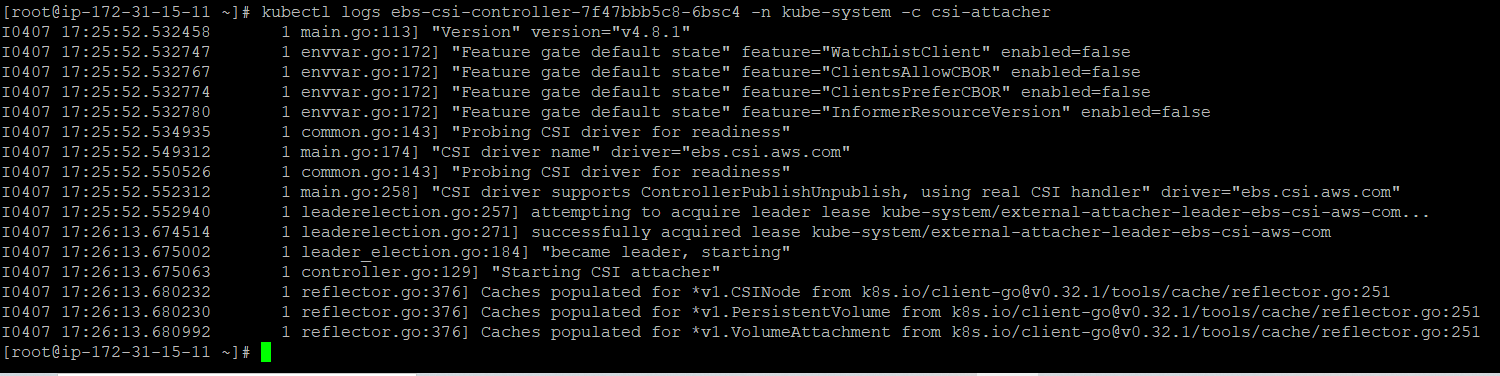
this command retrives log for default container



How to check logs for pod with specific container

kubectl logs ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system -c <container\_name>

kubectl logs ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system -c csi-attacher



aws eks update-kubeconfig --region ap-south-1 --name demo-aws-eks-cluster

20 kubectl get node

21 kubectl get node -o wide

22 clear

23 kubectl get ns dev

24 kubectl create ns dev

25 kubectl create ns airindia-dev

26 kubectl create ns airindia-preprod

27 kubectl create ns airindia-prod

28 kubectl get ns

29 sudo su -

30 cat /root/.kube/config

31 kubectl get node

32 kubectl get node -o wide

33 kubectl get ns

34 kubectl create ns indigo-dev

35 kubectl get pod -n kube-system

36 pwd

37 cat > ns.yaml

38 nore ns.yaml

39 more ns.yaml

40 kubectl apply -f ns.yaml

41 vi ns.yaml

42 kubectl apply -f ns.yaml

43 clear

44 kubectl run mypod-dev --image=nginx --namespace=dev

45 cat > dev-pod.yaml

46 more dev-pod.yaml

47 kubectl apply -f dev-pod.yaml

48 vi dev-pod.yaml

49 kubectl apply -f dev-pod.yaml

50 kubectl get pod -n dev

51 kubectl get pod -A

52 kubectl describe pod ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system

53 kubectl get pod -A

54 kubectl get pod

55 kubectl get pod dev

56 kubectl get pod -n dev

57 kubectl describe pod my-pod-prepod -n dev

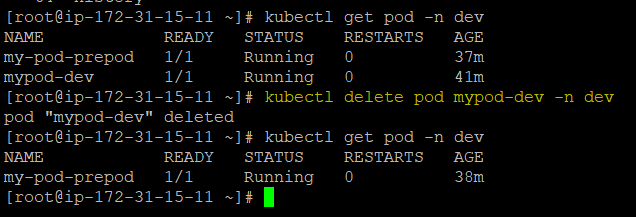
58 kubectl logs my-pod-prepod -n dev

59 kubectl logs ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system

63 kubectl logs ebs-csi-controller-7f47bbb5c8-6bsc4 -n kube-system -c csi-attacher

How to delete a pod

kubectl delete pod mypod-dev -n dev



helm install nginx-ingress ingress-nginx/ingress-nginx --namespace ingress-nginx --set controller.service.type=LoadBalancer --set controller.service.annotations."service\.beta\.kubernetes\.io/aws-load-balancer-type"="nlb" --set controller.service.annotations."service\.beta\.kubernetes\.io/aws-load-balancer-schme"="internet-facing" --set controller.service.annotations."service\.beta\.kubernetes\.io/aws-load-balancer-nlb-target-type"="ip"

aws eks update-cluster-version --name demo\_cluster--kubernetes-version 1.32 --region ap-south-1

[root@ip-172-31-14-57 ~]# aws eks update-cluster-version --name demo\_cluster --kubernetes-version 1.32 --region ap-south-1

{

"update": {

"id": "24aaf3e7-068e-309a-b7e6-63a4600fbd25",

"status": "InProgress",

"type": "VersionUpdate",

"params": [

{

"type": "Version",

"value": "1.32"

},

{

"type": "PlatformVersion",

"value": "eks.6"

}

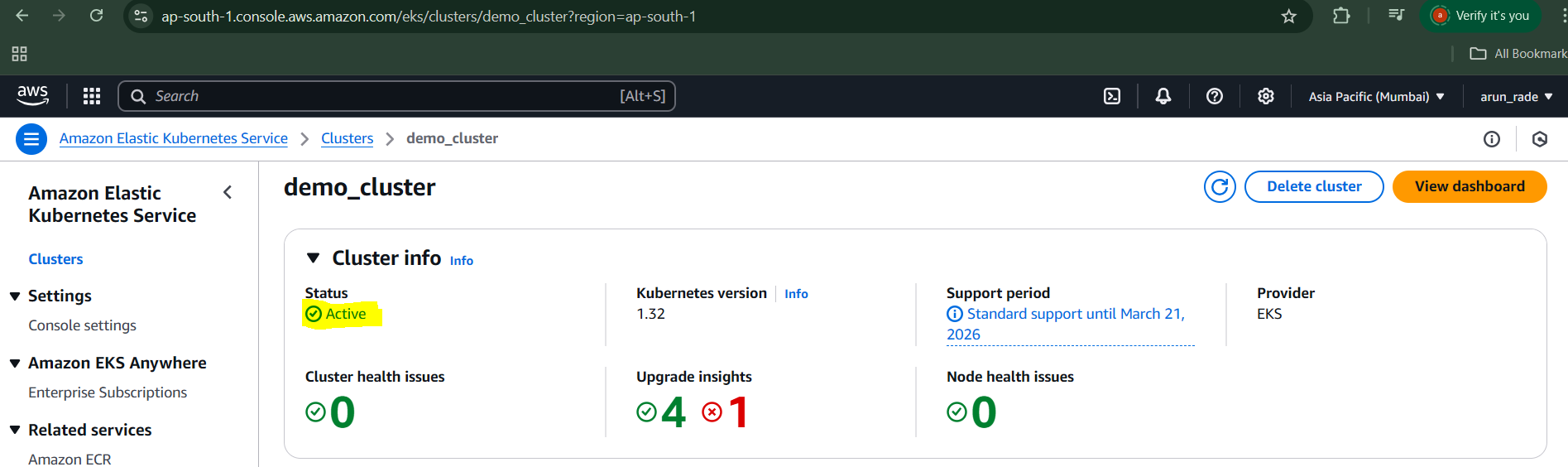
],

"createdAt": "2025-04-11T13:18:24.377000+00:00",

"errors": []

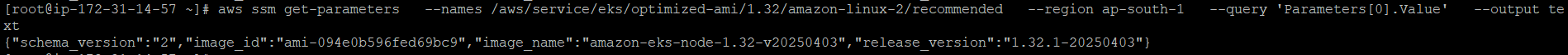
}

}



eksctl upgrade nodegroup --name=WorkerNode-01 --cluster=demo\_cluster --region=ap-south-1 --kuberenetes-version=1.32

aws ssm get-parameters --names /aws/service/eks/optimized-ami/1.32/amazon-linux-2/recommended --region ap-south-1 --query 'Parameters[0].Value' --output text



[root@ip-172-31-14-57 ~]# aws ssm get-parameters --names /aws/service/eks/optimized-ami/1.32/amazon-linux-2/recommended --region ap-south-1 --query 'Parameters[0].Value' --output text

{"schema\_version":"2","image\_id":"ami-094e0b596fed69bc9","image\_name":"amazon-eks-node-1.32-v20250403","release\_version":"1.32.1-20250403"}

update worker node

aws eks update-nodegroup-version --cluster-name demo --nodegroup-name WorkerNode01 --region ap-south-1 --release-version 1.32.1-20250403

[root@ip-172-31-14-57 ~]# aws eks update-nodegroup-version --cluster-name demo\_cluster --nodegroup-name WorkerNode-01 --region ap-south-1 --release-version 1.32.1-20250403

{

"update": {

"id": "fb68f0da-e7cc-3ef0-a333-507c95153b6c",

"status": "InProgress",

"type": "VersionUpdate",

"params": [

{

"type": "Version",

"value": "1.32"

},

{

"type": "ReleaseVersion",

"value": "1.32.1-20250403"

}

],

"createdAt": "2025-04-11T14:40:21.715000+00:00",

"errors": []

}

}

[root@ip-172-31-14-57 ~]#