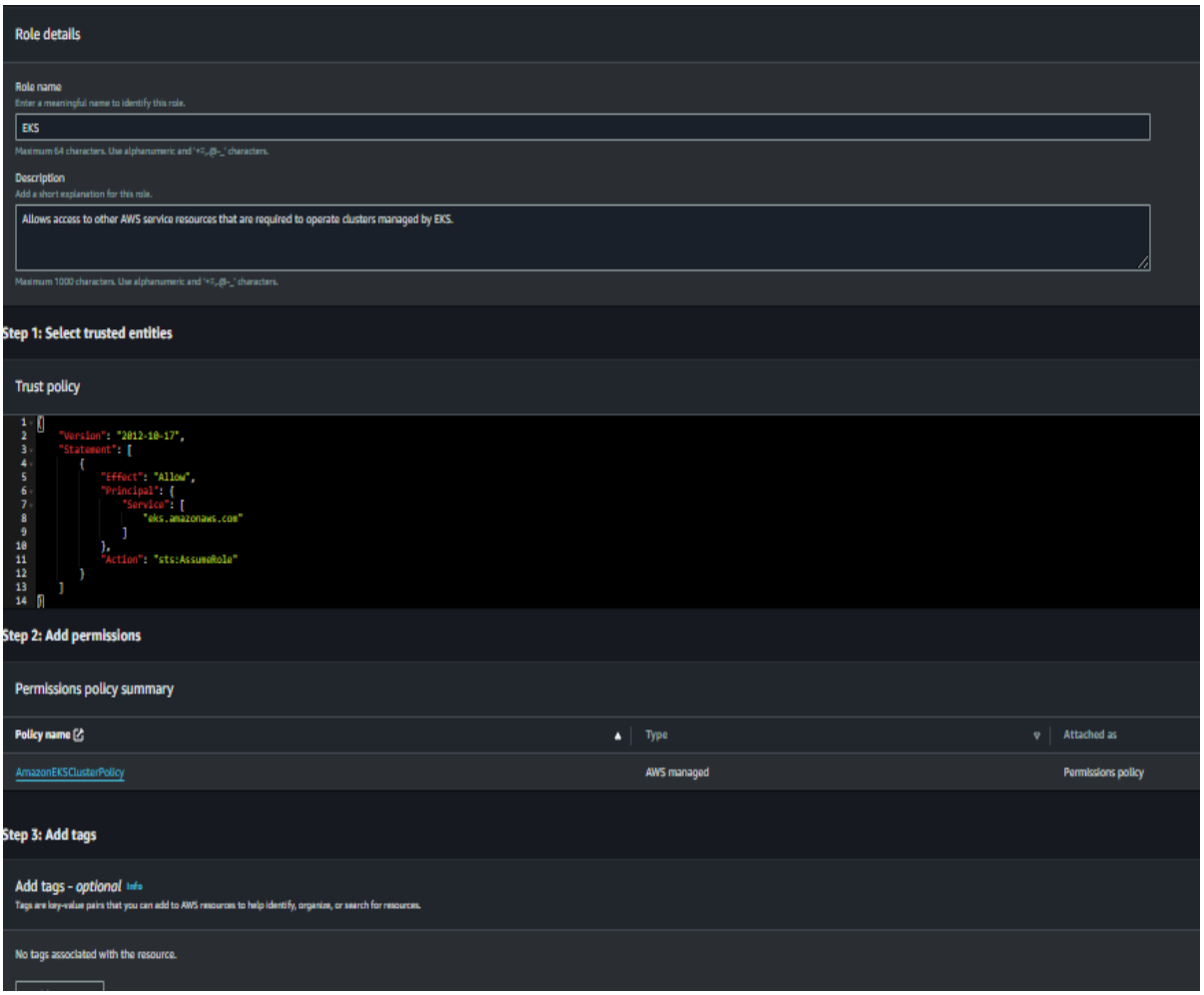
CDEC – B24

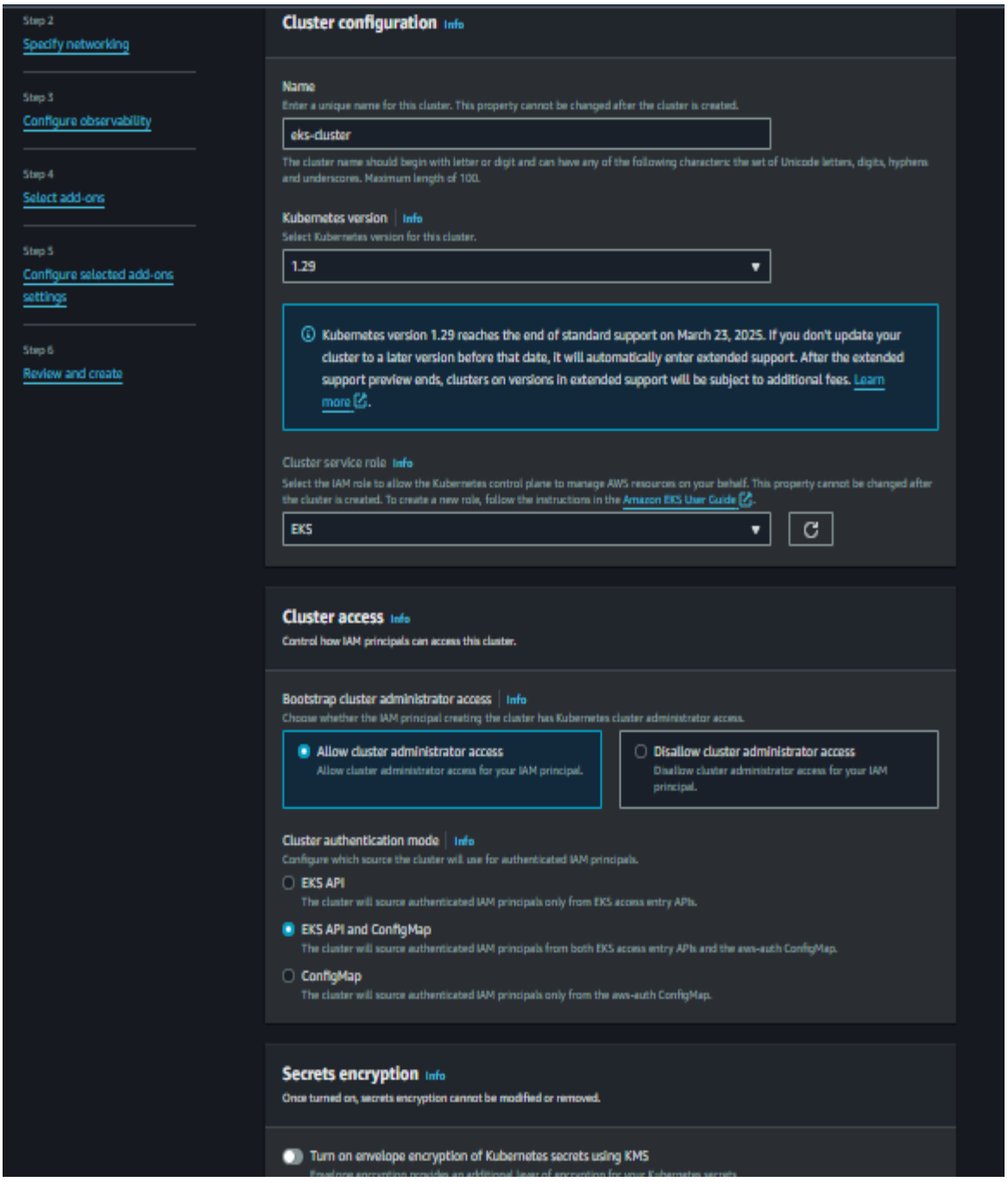
Name – Vaibhav Navneet Jorvekar

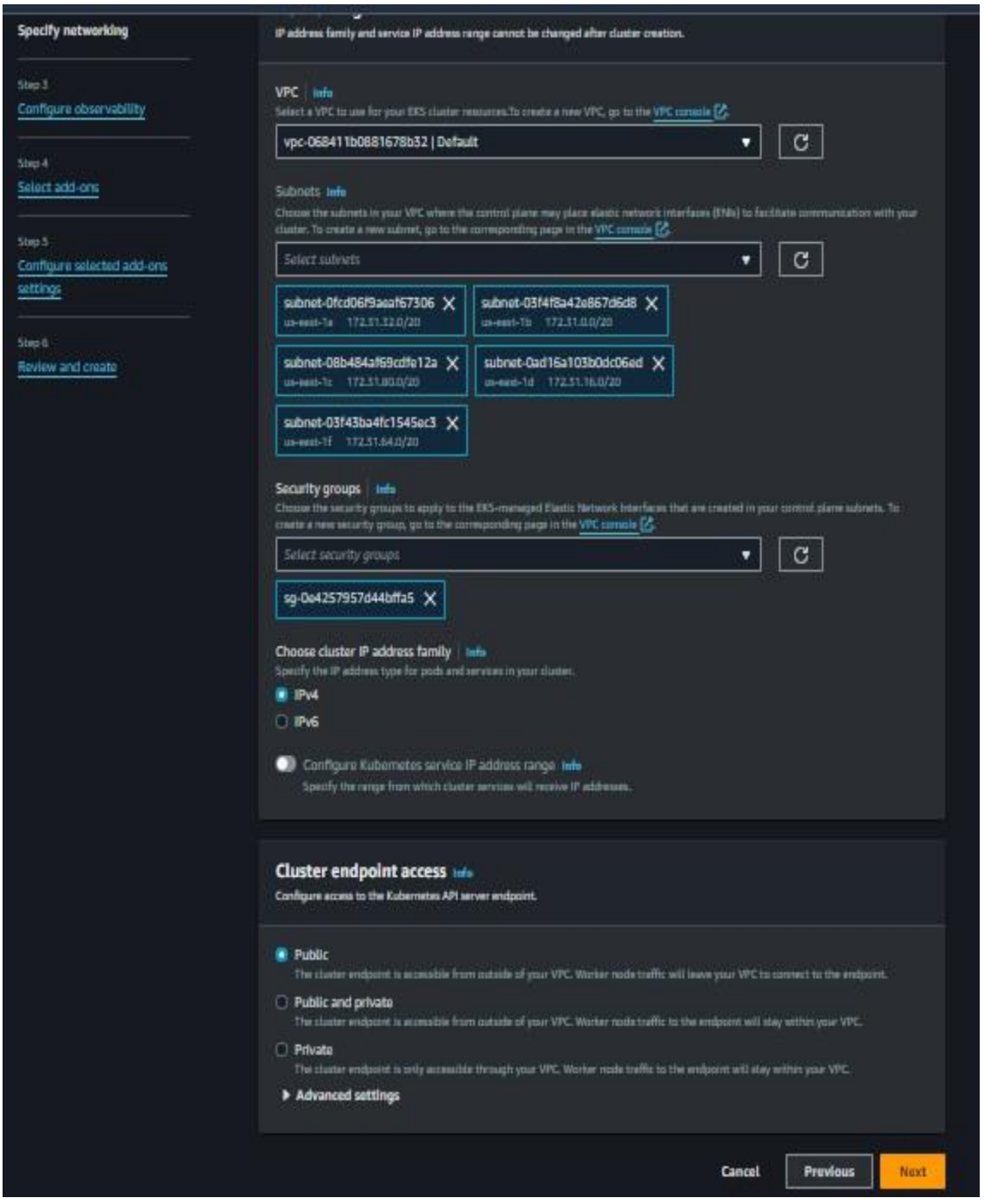
Hosting of nginx and tomcat via manifest file

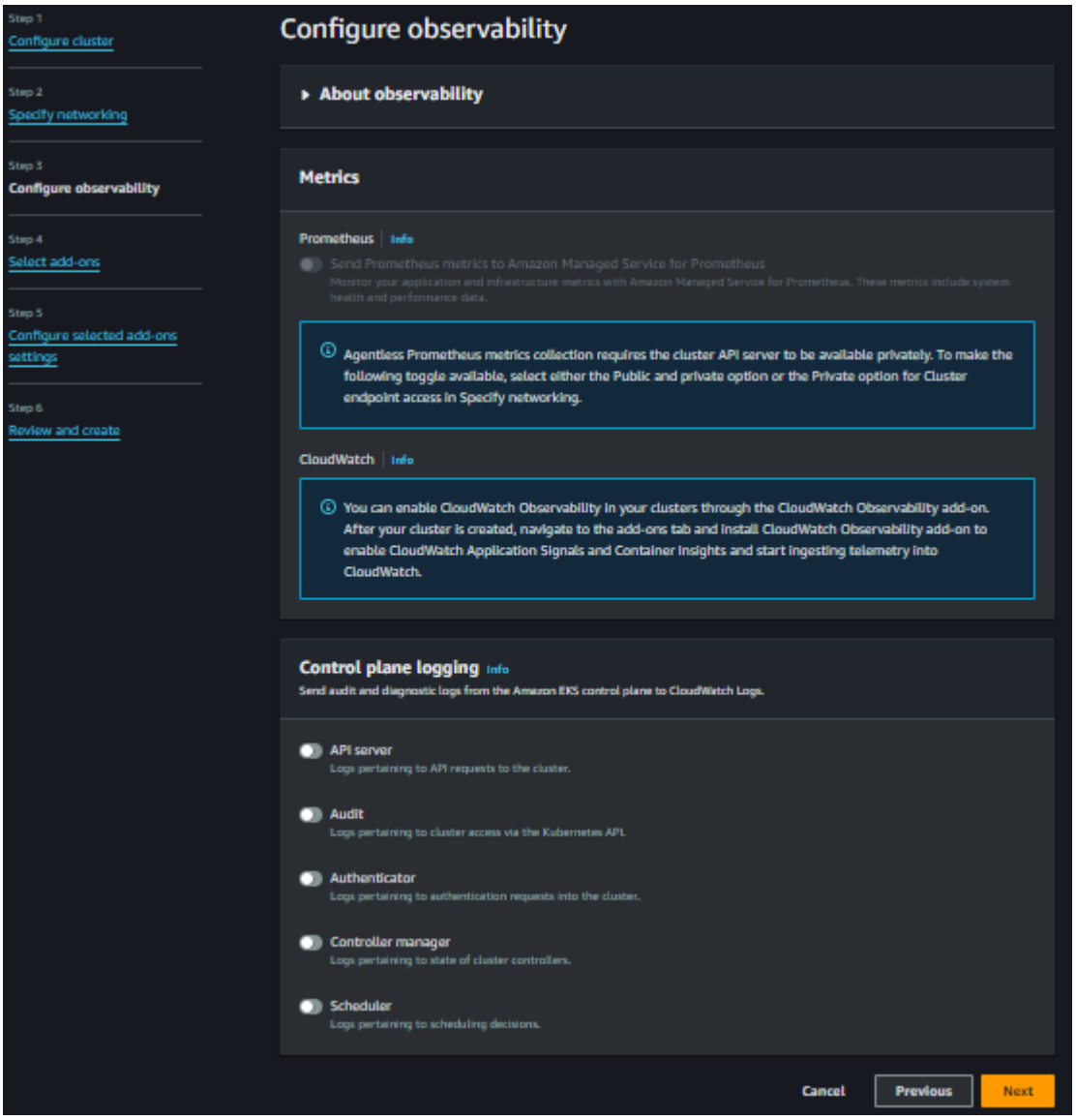
1. **Set up IAM roles for EKS.**
    - **Go to aws IAM service and create a new role for the EKS**

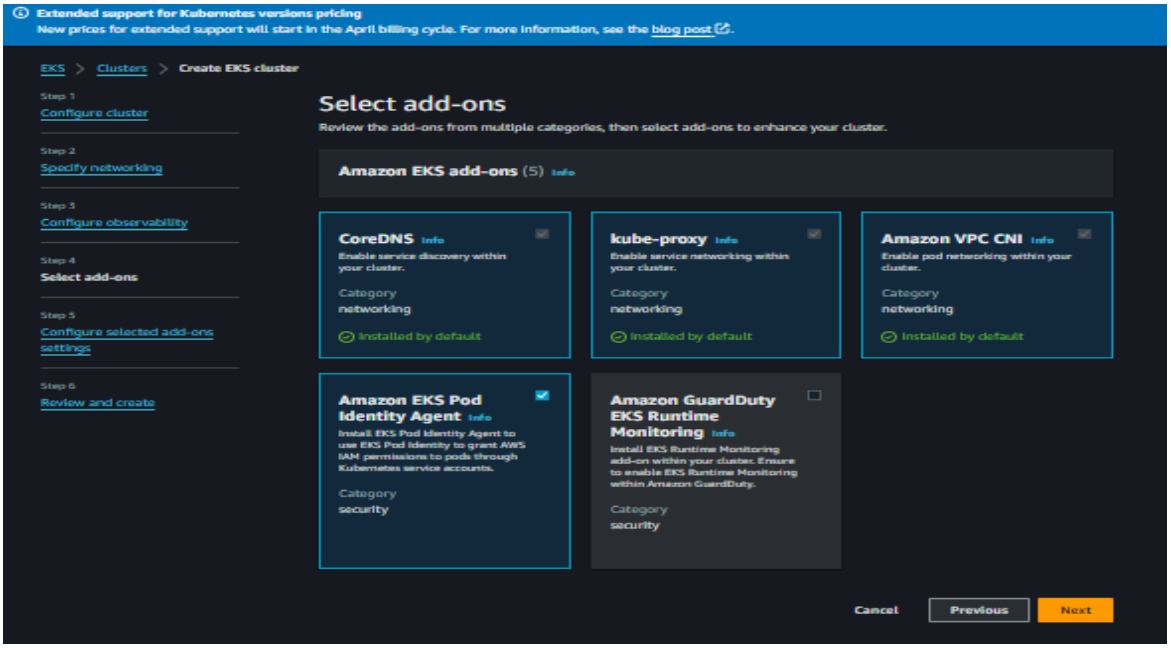


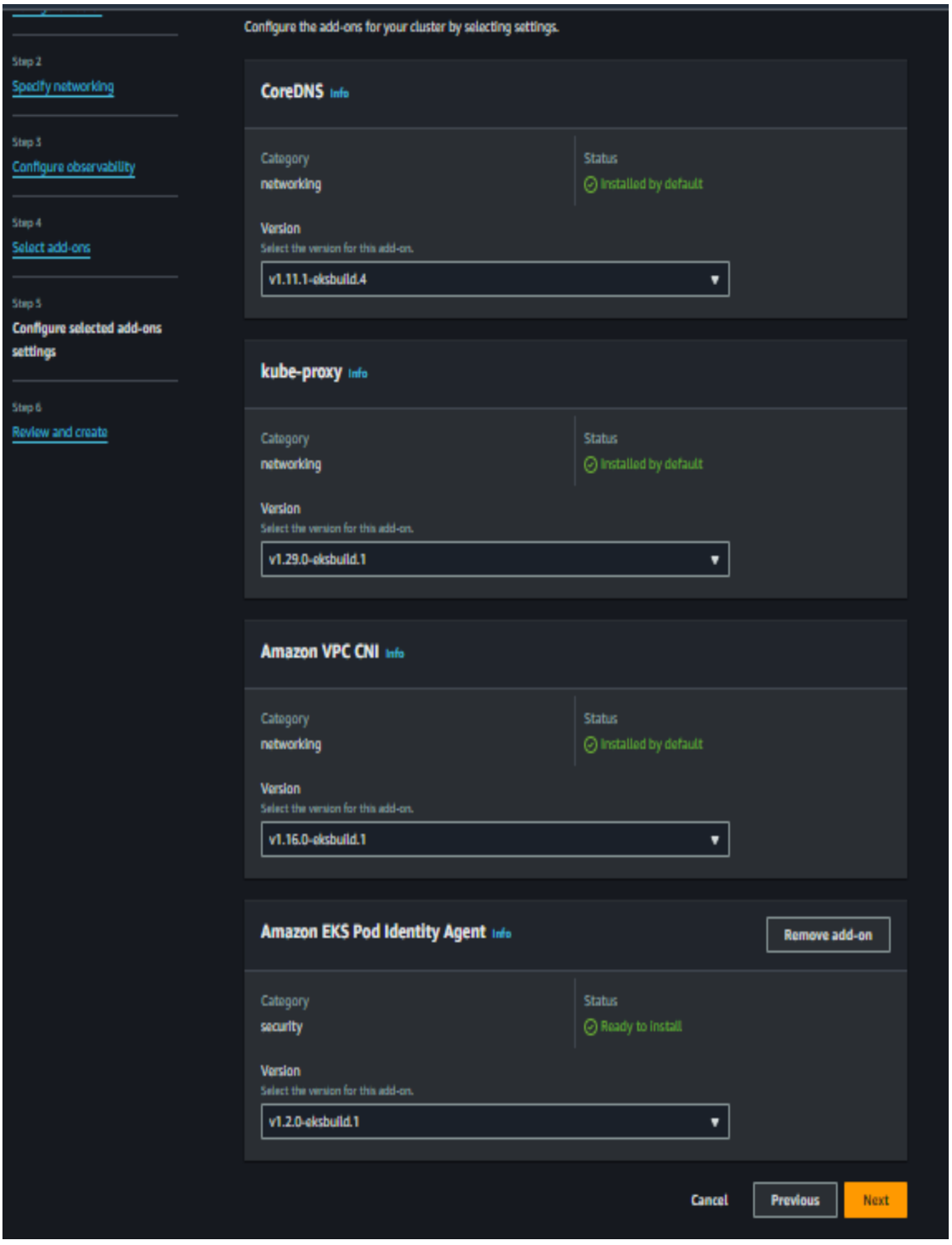
2. **Create an EKS cluster.**
    - **Open the Amazon EKS console.**
    - **Click on “Create Cluster” and choose the “AWS management Console” method.**

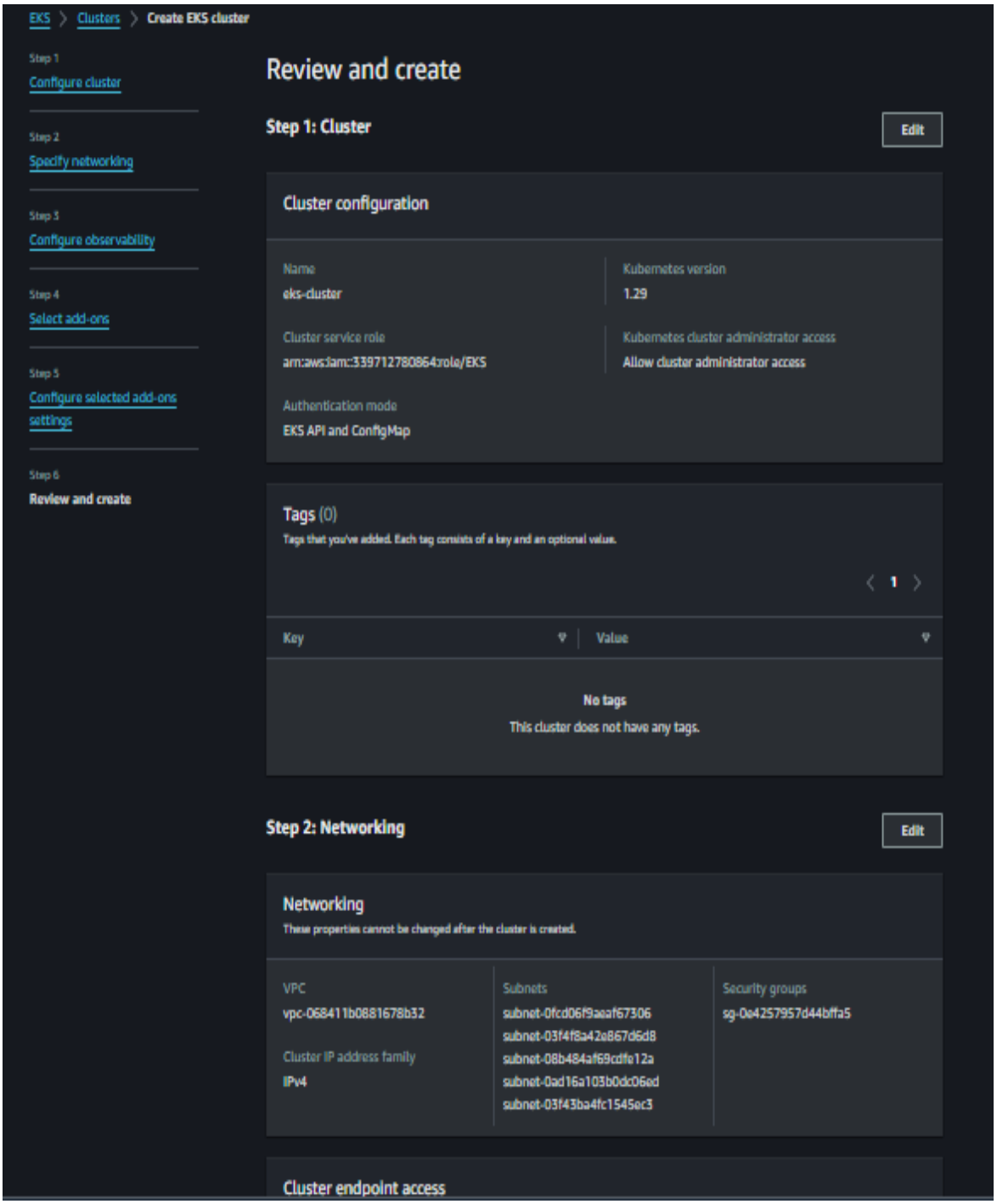



Specify networking
Step 3
Configure observability
Step 4
Select add-ons
Step 5
Configure selected add-ons settings
Step 6
Review and create
IP address family and service IP address range cannot be changed after cluster creation.
VPC | Info
Select a VPC to use for your EKS cluster resources.To create a new VPC, go to the VPC console
vpc-06841 1b0881678b32 | Default
Subnets Info
Choose the subnets in your VPC where the control plane may place elastic network interfaces (ENIs) to facilitate communication with your cluster. To create a new subnet, go to the corresponding page in the VPC console
Select subnets
subnet-0fcd06f9aeaf67306
us-east-1a 172.31.32.0/20
subnet-03f4f8a42a867d6d8
us-east-1b 172.31.0.0/20
subnet-08b484af69cdfe12a
us-east-1c 172.31.80.0/20
subnet-0ad16a103b0dc06ed
us-east-1d 172.31.16.0/20
subnet-03f43ba4fc1545ec3
us-east-1f 172.31.64.0/20
Security groups | Info
Choose the security groups to apply to the EKS-managed Elastic Network Interfaces that are created in your control plane subnets. To create a new security group, go to the corresponding page in the VPC console
Select security groups
sg-0e4257957d44bffa5
Choose cluster IP address family | Info
Specify the IP address type for pods and services in your cluster.
IPv4
IPv6
Configure Kubernetes service IP address range Info
Specify the range from which cluster services will receive IP addresses.
Cluster endpoint access Info
Configure access to the Kubernetes API server endpoint.
Public
The cluster endpoint is accessible from outside of your VPC. Worker node traffic will leave your VPC to connect to the endpoint.
Public and private
The cluster endpoint is accessible from outside of your VPC. Worker node traffic to the endpoint will stay within your VPC.
Private
The cluster endpoint is only accessible through your VPC. Worker node traffic to the endpoint will stay within your VPC.
Advanced settings
Cancel
Previous
Next

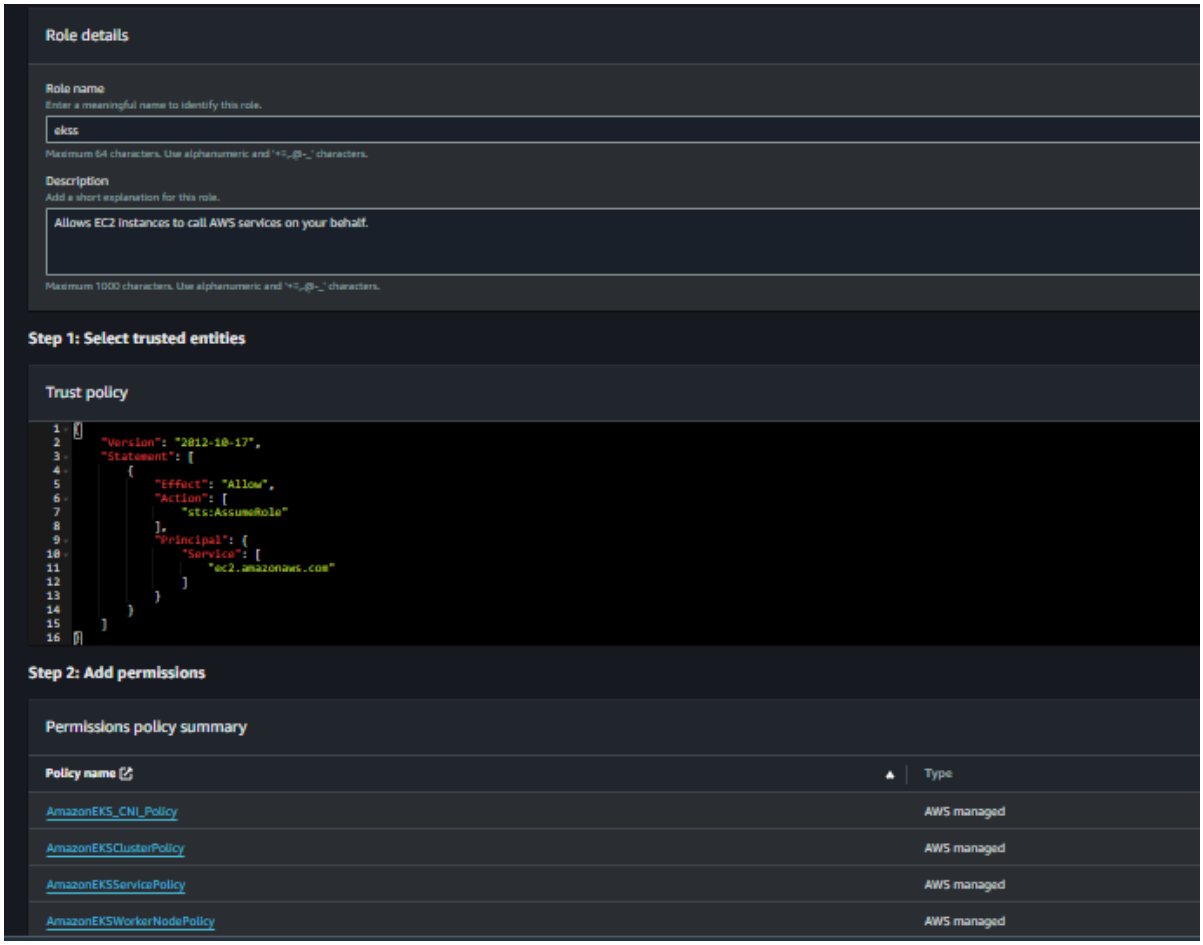

Step 1
Configure cluster
Step 2
Specify networking
Step 3
Configure observability
Step 4
Select add-ons
Step 5
Configure selected add-ons settings
Step 6
Review and create
Configure observability
About observability
Metrics
Prometheus Info
Send Prometheus metrics to Amazon Managed Service for Prometheus
Monitor your application and infrastructure metrics with Amazon Managed Service for Prometheus. These metrics include system health and performance data.
Agentless Prometheus metrics collection requires the cluster API server to be available privately. To make the following toggle available, select either the Public and private option or the Private option for Cluster endpoint access in Specify networking.
CloudWatch Info
You can enable CloudWatch Observability in your clusters through the CloudWatch Observability add-on. After your cluster is created, navigate to the add-ons tab and install CloudWatch Observability add-on to enable CloudWatch Application Signals and Container Insights and start ingesting telemetry into CloudWatch.
Control plane logging Info
Send audit and diagnostic logs from the Amazon EKS control plane to CloudWatch Logs.
API server
Logs pertaining to API requests to the cluster.
Audit
Logs pertaining to cluster access via the Kubernetes API.
Authenticator
Logs pertaining to authentication requests into the cluster.
Controller manager
Logs pertaining to state of cluster controllers.
Scheduler
Logs pertaining to scheduling decisions.
Cancel
Previous
Next


Extended support for Kubernetes versions pricing
New prices for extended support will start in the April billing cycle. For more information, see the blog post.
EKS > Clusters > Create EKS cluster
Step 1
Configure cluster
Step 2
Specify networking
Step 3
Configure observability
Step 4
Select add-ons
Step 5
Configure selected add-ons settings
Step 6
Review and create
Select add-ons
Review the add-ons from multiple categories, then select add-ons to enhance your cluster.
Amazon EKS add-ons (5) Info
CoreDNS Info
Enable service discovery within your cluster.
Category
networking
Installed by default
kube-proxy Info
Enable service networking within your cluster.
Category
networking
Installed by default
Amazon VPC CNI Info
Enable pod networking within your cluster.
Category
networking
Installed by default
Amazon EKS Pod Identity Agent Info
Install EKS Pod Identity Agent to use EKS Pod Identity to grant AWS IAM permissions to pods through Kubernetes service accounts.
Category
security
Amazon GuardDuty EKS Runtime Monitoring Info
Install EKS Runtime Monitoring add-on within your cluster. Ensure to enable EKS Runtime Monitoring within Amazon GuardDuty.
Category
security
Cancel
Previous
Next


Step 2
Specify networking
Step 3
Configure observability
Step 4
Select add-ons
Step 5
Configure selected add-ons settings
Step 6
Review and create
Configure the add-ons for your cluster by selecting settings.
CoreDNS Info
Category
networking
Status
Installed by default
Version
Select the version for this add-on.
v1.11.1-eksbuild.4
kube-proxy Info
Category
networking
Status
Installed by default
Version
Select the version for this add-on.
v1.29.0-eksbuild.1
Amazon VPC CNI Info
Category
networking
Status
Installed by default
Version
Select the version for this add-on.
v1.16.0-eksbuild.1
Amazon EKS Pod Identity Agent Info
Remove add-on
Category
security
Status
Ready to install
Version
Select the version for this add-on.
v1.2.0-eksbuild.1
Cancel
Previous
Next


EKS > Clusters > Create EKS cluster
Step 1
Configure cluster
Step 2
Specify networking
Step 3
Configure observability
Step 4
Select add-ons
Step 5
Configure selected add-ons settings
Step 6
Review and create
Review and create
Step 1: Cluster
Edit
Cluster configuration
Name
eks-cluster
Kubernetes version
1.29
Cluster service role
arn:aws:iam::339712780864:role/EKS
Kubernetes cluster administrator access
Allow cluster administrator access
Authentication mode
EKS API and ConfigMap
Tags (0)
Tags that you've added. Each tag consists of a key and an optional value.
1
Key
Value
No tags
This cluster does not have any tags.
Step 2: Networking
Edit
Networking
These properties cannot be changed after the cluster is created.
VPC
vpc-068411b0881678b32
Cluster IP address family
IPv4
Subnets
subnet-0fcd06f9aeaf67306
subnet-03f4f8a42a867d6d8
subnet-08b484af69cdfe12a
subnet-0ad16a103b0dc06ad
subnet-03f43ba4fc1545ec3
Security groups
sg-0e4257957d44bffa5
Cluster endpoint access

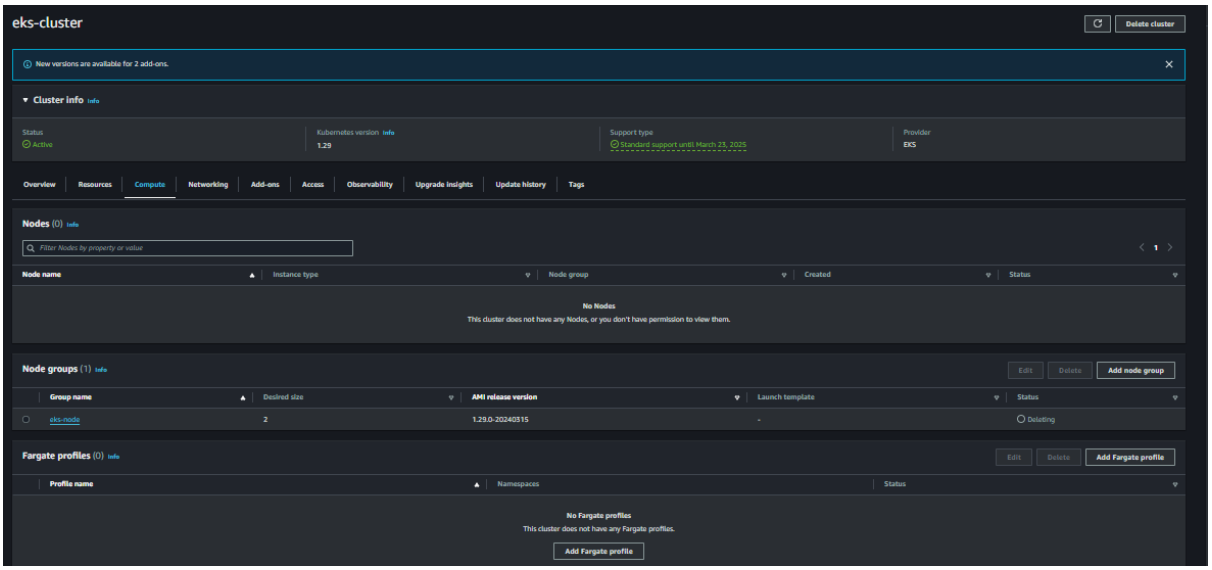
3. **Set up IAM roles for EC2.**



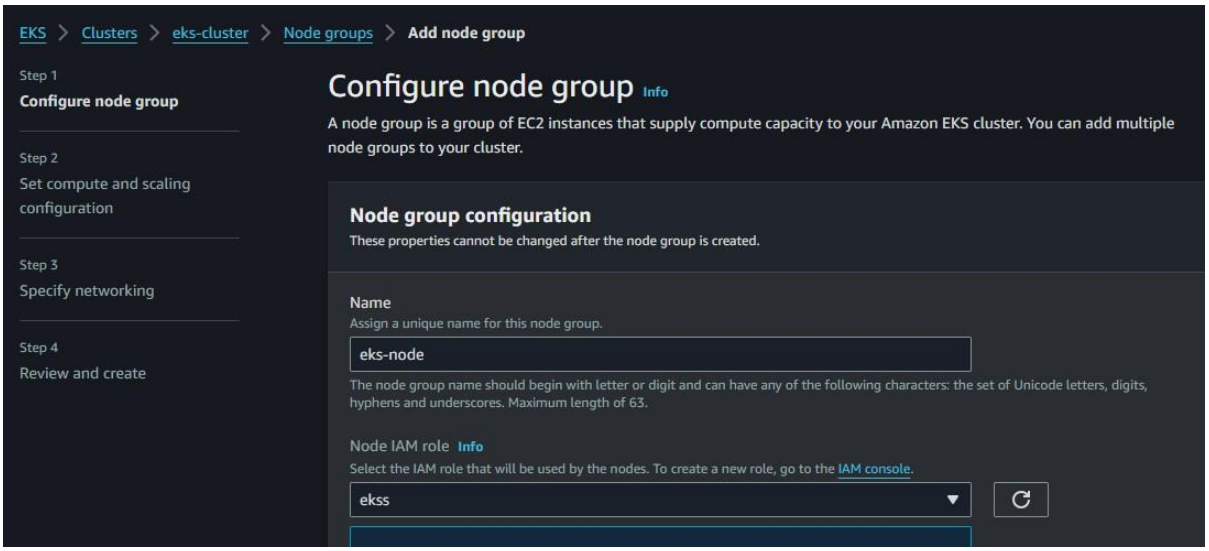
4. **Configure the AWS Cloudshell.**
    - **Open aws cloudshell & configure aws.**

```
[cloudshell-user@ip-10-134-62-175 ~]$ aws configure
AWS Access Key ID [****************D573]: AKIAU6GDVVJAM3NWD573
AWS Secret Access Key [****************hH0O]: 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
Default region name [None]:
Default output format [None]:
```

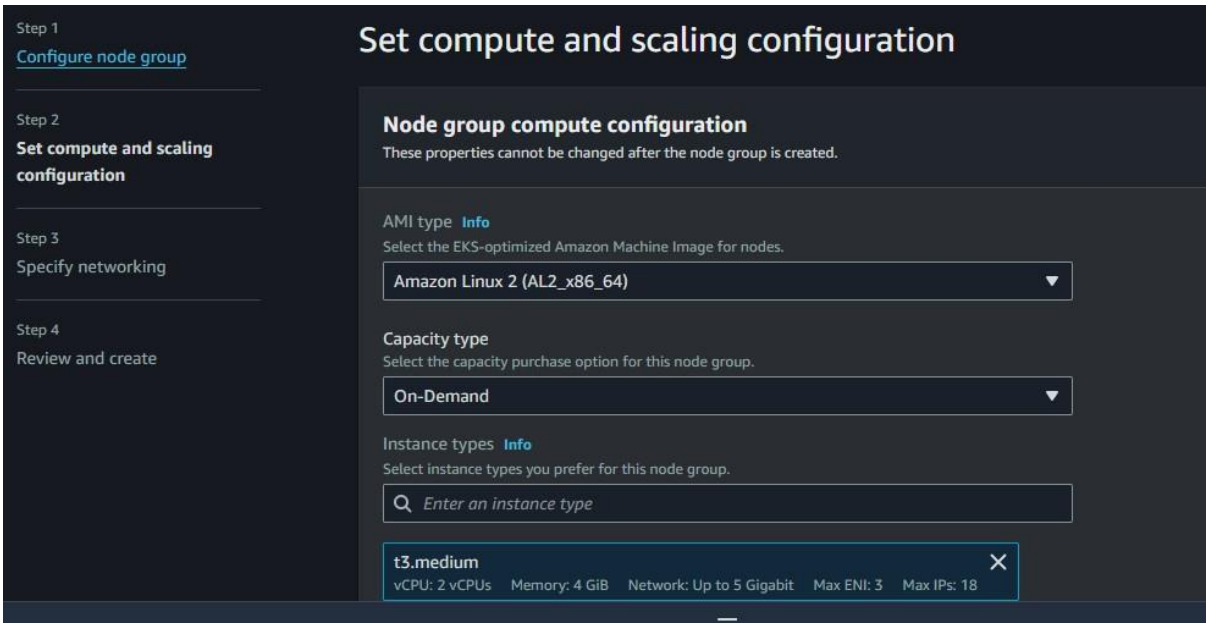
5. **Add worker nodes.**
    - **In the AWS EKS console select your cluster.**
    - **In cluster go to compute service.**



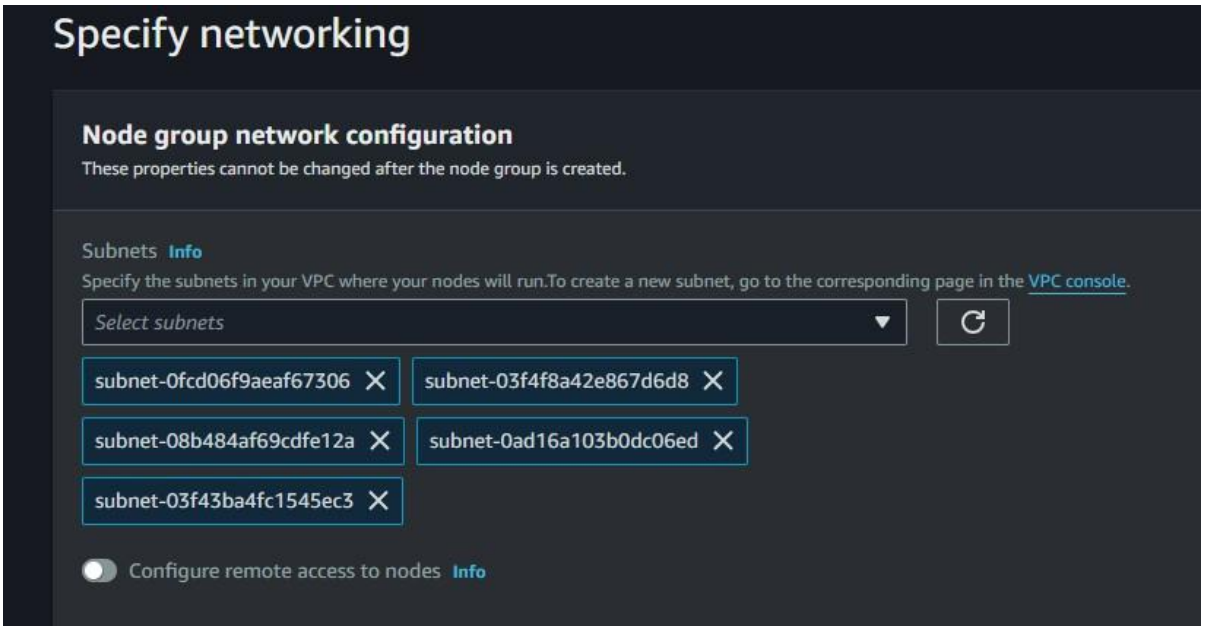
- **Click on “Ad Node Group”.**
- **Select the “Name” & “IAM ROLE”.**



- **Click on next.**
- **Select the values for the node configuration a below.**



- **Click on next.**
- **Select the subnets.**



- **Click on "next" and then ''Create''**

6. **Verify the cluster.**
   - **Open cloudshell and execute the following commands.**
     # aws eks update-kubeconfig --region <region> --name <cluster-name>
     # kubectl cluster-info

```
[cloudshell-user@ip-10-134-62-175 ~]$ aws eks update-kubeconfig --region ap-southeast-2 --name vintage
Added new context arn:aws:eks:ap-southeast-2:339712780864:cluster/vintage to /home/cloudshell-user/.kube/config
[cloudshell-user@ip-10-134-62-175 ~]$ kubectl cluster-info
Kubernetes control plane is running at https://3E4C6A3CB9CFD6BD6E0C9E3E074D47D4.gr7.ap-southeast-2.eks.amazonaws.com
CoreDNS is running at https://3E4C6A3CB9CFD6BD6E0C9E3E074D47D4.gr7.ap-southeast-2.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

7. **Create pod.yml & service.yml file in your VS code and upload files on your git repo**

8. **Create pod file for nginx and tomcat with extension pod.yml.**

- **Pods.yml**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: ompod
  labels:
    app: new-app
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80
      protocol: TCP
  - name: tomcat
    image: tomcat:latest
    ports:
    - containerPort: 8080
      protocol: TCP

```

9. **After completing the script create pod using command.**
   git clone <your repo URL >
   - **git clone https://github.com/vaibhavjorvekar2306/kubic.git**
     (in my case my file present in Kubic repo.)
   - **ls**
   - **(goes upto your pod.yml file for creation of node)**
   - **kubectl apply -f pods.yml**
   - **kubectl get pods**
   - **kubectl get -o wide pods**
   - **kubectl describe pods**

```
[cloudshell-user@ip-10-134-62-175 kubic]$ ls
2048game.yml  Nginx_web.yml  pods.yml  prod_service-namespace.yml  README.md  service.ym
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl apply -f pods.yml
pod/ompod created
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
ompod   2/2     Running   0          40s
```

```
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl get -o wide pods
NAME    READY   STATUS    RESTARTS   AGE   IP              NODE                                                NOMINATED NODE   READINESS GATES
ompod   2/2     Running   0          61s   172.31.16.212   ip-172-31-16-78.ap-southeast-2.compute.internal   <none>           <none>
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl describe pods
Name:             ompod
Namespace:        default
Priority:         0
Service Account:  default
Node:             ip-172-31-16-78.ap-southeast-2.compute.internal/172.31.16.78
Start Time:       Mon, 01 Apr 2024 05:22:24 +0000
Labels:           app=new-app
Annotations:      <none>
Status:           Running
IP:               172.31.16.212
IPs:
  IP:  172.31.16.212
Containers:
  nginx:
    Container ID:   containerd://b1c81a3db95e728420cf73118ce1eb6225aff2b768c5f2e75890f2f4bba992a6
    Image:          nginx:latest
    Image ID:       docker.io/library/nginx@sha256:6db391d1c0cfb30588ba0bf72ea999404f2764febf0f1f196acd5867ac7efa7e
    Port:           80/TCP
    Host Port:      0/TCP
    Image:          tomcat:latest
    Image ID:       docker.io/library/tomcat@sha256:0c6f42391c80066ce4ebf635726ace10b1ccd22861683a57c40791ce129d4cb4
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 01 Apr 2024 05:22:44 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-zxvwq (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-zxvwq:
Volumes:
  kube-api-access-zxvwq:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
```

10. Create service file for nginx and tomcat with extension service.yml.

- **Service.yml**

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nodesvc
spec:
  selector:
    app: new-app
  type: NodePort
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
      name: nginx

    - protocol: TCP
      port: 8080
      targetPort: 8080
      name: tomcat

```

11. Use commands to create service.

- ls
- kubectl apply -f service.yml
- kubectl get svc (services)

```
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl apply -f service.yml
service/nodesvc created
[cloudshell-user@ip-10-134-62-175 kubic]$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)                   AGE
kubernetes   ClusterIP   10.100.0.1     <none>        443/TCP                   25m
nodesvc      NodePort    10.100.56.87   <none>        80:30783/TCP,8080:31962/TCP   21s
[cloudshell-user@ip-10-134-62-175 kubic]$
```

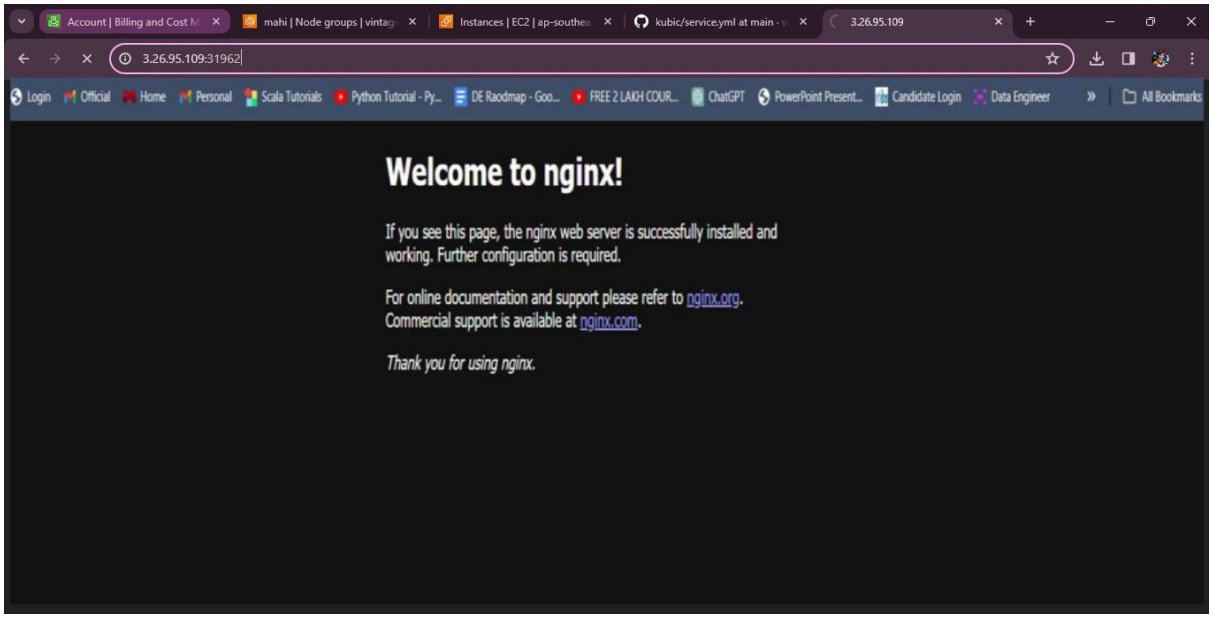
12. After creation of service hit the IP of your instance which is created while creaΘon of node group.

For nginx :

<Instance-ip>:<port-no>

In my case;

http://3.26.95.109:31962/



- For tomcat
- http://3.26.95.109:30783/