

Creating EKS(Elastic Kubernetes Service)

The screenshot shows the AWS Management Console for Amazon Elastic Kubernetes Service (EKS). The left sidebar contains navigation links for Amazon Elastic Kubernetes Service, including Dashboard, Clusters, Settings, Amazon EKS Anywhere, and Related services. The main content area displays the 'Elastic Kubernetes Service (Amazon EKS)' header, a 'Get started' section with a 'Create cluster' button, and an 'Introducing the new EKS Dashboard' section. Below this, the 'Create EKS cluster' wizard is visible, showing a progress bar with steps 1 through 6. The current step is 'Specify networking', which includes sections for 'Configuration options' (Quick configuration and Custom configuration), 'EKS Auto Mode' (Use EKS Auto Mode), and 'Cluster configuration' (Name).

While Creating cluster need to create a role (IAM). which give access to clusters (ControlPlane) and other Services.

The screenshot shows the AWS Management Console for Amazon Elastic Kubernetes Service (EKS) during the 'Create EKS cluster' wizard. The progress bar indicates the current step is 'Specify networking'. The 'Cluster configuration' section is expanded, showing the 'Name' field with the value 'eks'. Below this, the 'Cluster IAM role' section is visible, showing a dropdown menu with 'ClusterRole' selected and a 'Create recommended role' button. The 'Kubernetes version settings' section is also visible, showing the 'Kubernetes version' dropdown set to '1.32' and the 'Upgrade policy' section with 'Standard' selected.

aws

Search

[Alt+S]

United States (N. Virginia)

Ganraj

EC2

IAM

VPC

Billing and Cost Manage...

Aurora and RDS

S3

Amazon Elastic Kubernetes Service

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

Specify networking

Networking

Info

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.

vpc-0d9bb05820b87165c | 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-0f976eab839104d9e

X

us-east-1d

172.31.32.0/20

subnet-04829216fe4f61b6

X

us-east-1a

172.31.0.0/20

subnet-0331ba118986b9a4f

X

us-east-1f

172.31.64.0/20

subnet-0b3b28230b801c487

X

us-east-1b

172.31.80.0/20

subnet-02038066e5692cbb

X

us-east-1c

172.31.16.0/20

Clear selected subnets

Additional security groups - optional

Info

EKS automatically creates a cluster security group on cluster creation to facilitate communication between worker nodes and control plane. Optionally, choose additional 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](#).

CloudShell

Feedback

© 2025, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

aws

Search

[Alt+S]

United States (N. Virginia)

Ganraj

EC2

IAM

VPC

Billing and Cost Manage...

Aurora and RDS

S3

Amazon Elastic Kubernetes Service

Create EKS cluster

Configure Kubernetes service IP address block

Info

☐ Specify the range from which cluster services will receive IP addresses.

Configure remote networks to enable hybrid nodes

Info

EKS Hybrid Nodes enables you to use on-premises and edge infrastructure as nodes in EKS clusters.

☐ Specify the CIDR blocks for your on-premises environments that you will use for hybrid nodes.

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

CloudShell

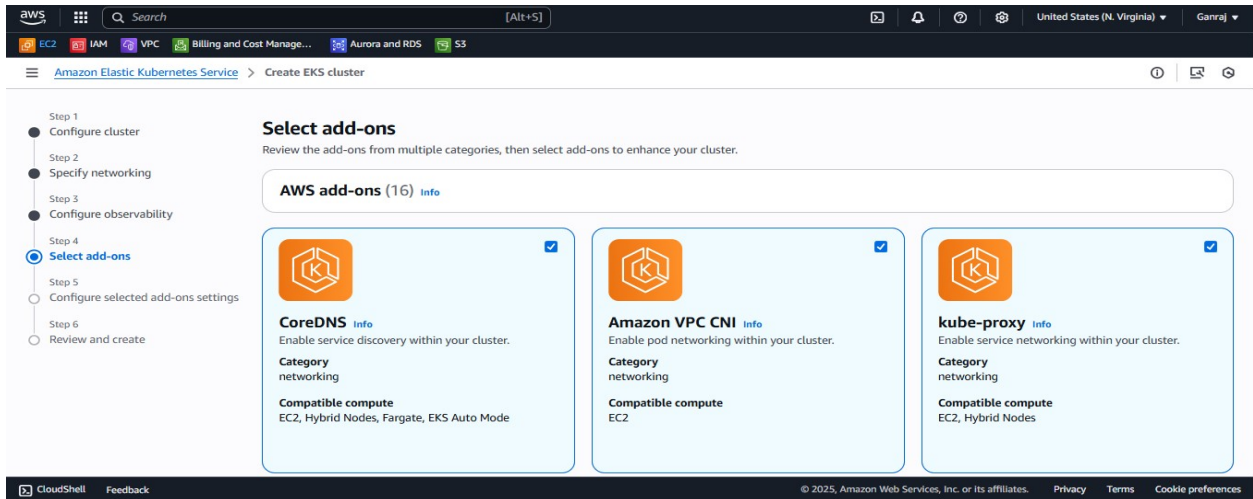
Feedback

© 2025, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

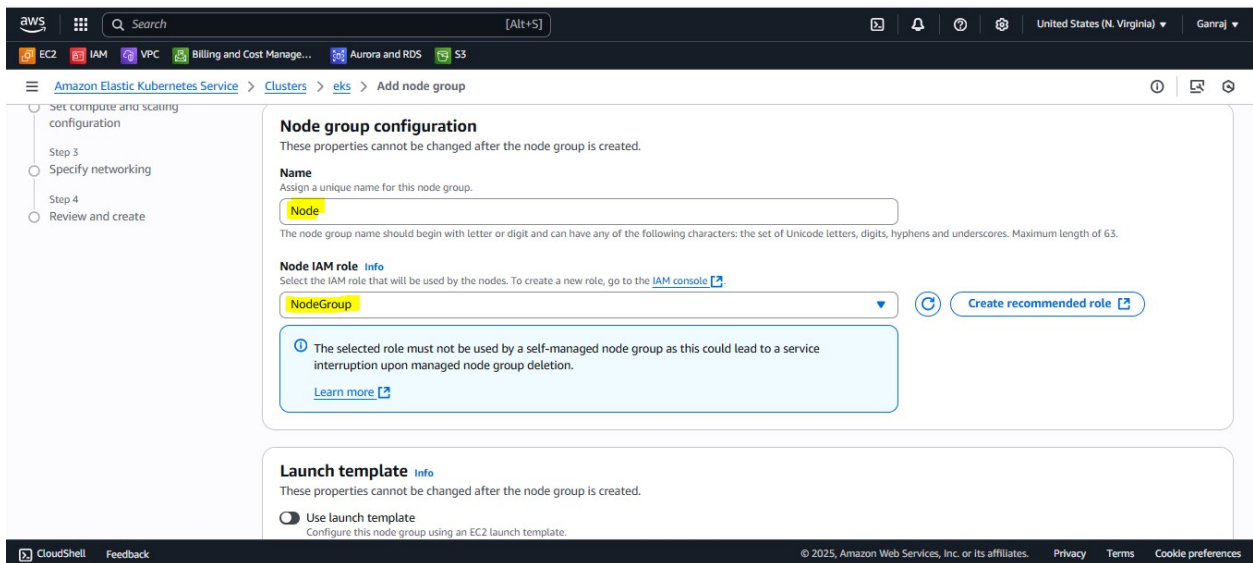
Cookie preferences



If don't need any extra add ons don't click cost apply.

AWS EKS create itself master node

Need create node groups that implements ec2 vm scaling components.



You can set number of scaling nodes (VM's, EC2) in **NodeGroups**

And in the nodegroup also need to create a role which works on setting up rules over services of instances (ec2) or VMs

aws

Search

[Alt+S]

United States (N. Virginia)

Ganraj

EC2IAMVPCBilling and Cost Manage...Aurora and RDS

Amazon Elastic Kubernetes Service > Clusters > eks > Add node group

Step 1: Configure node group

Step 2: Set compute and scaling configuration

Step 3: Specify networking

Step 4: Review and create

Node group compute configuration

These properties cannot be changed after the node group is created.

AMI type info

Select the EKS-optimized Amazon Machine Image for nodes.

Amazon Linux 2023 (x86_64) Standard (AL2023_x86_64_STANDARD)

Capacity type

Select the capacity purchase option for this node group.

On-Demand

Instance types info

Select instance types you prefer for this node group.

Enter an instance type

t3.medium

vCPU: 2 vCPUs Memory: 4 GiB Network: Up to 5 Gigabit Max ENI: 3 Max IPs: 18

Disk size

Select the size of the attached EBS volume for each node.

20

GiB

Node group scaling configuration

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

aws

Search

[Alt+S]

United States (N. Virginia)

Ganraj

EC2IAMVPCBilling and Cost Manage...Aurora and RDS

Amazon Elastic Kubernetes Service > Clusters > eks > Add node group

Step 1: Configure node group

Step 2: Set compute and scaling configuration

Step 3: Specify networking

Step 4: Review and create

Specify networking

Node group network configuration

These properties cannot be changed after the node group is created.

Subnets info

Specify the subnets in your VPC where your nodes will run. To create a new subnet, go to the corresponding page in the VPC console.

Select subnets

Clear selected subnets

subnet-0f976eab839104d9e

us-east-1d 172.31.32.0/20

subnet-04829216fe64f61b6

us-east-1a 172.31.0.0/20

subnet-0331ba118986b9a4f

us-east-1f 172.31.64.0/20

subnet-0b3b28230b801c487

us-east-1b 172.31.80.0/20

subnet-02038066e5692cbba

us-east-1c 172.31.16.0/20

subnet-09e0679efdf6a696

us-east-1e 172.31.48.0/20

Configure remote access to nodes info

Cancel Previous Next

aws

Search

[Alt+S]

United States (N. Virginia)

Ganraj

EC2IAMVPCBilling and Cost Manage...Aurora and RDS

Amazon Elastic Kubernetes Service > Clusters > eks > Add node group

Node auto repair configuration

Node auto repair

Disabled

Step 3: Networking

Edit

Node group network configuration

Subnets

subnet-0f976eab839104d9e

subnet-04829216fe64f61b6

subnet-0331ba118986b9a4f

subnet-0b3b28230b801c487

subnet-02038066e5692cbba

subnet-09e0679efdf6a696

Configure remote access to nodes

off

Cancel Previous Create

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

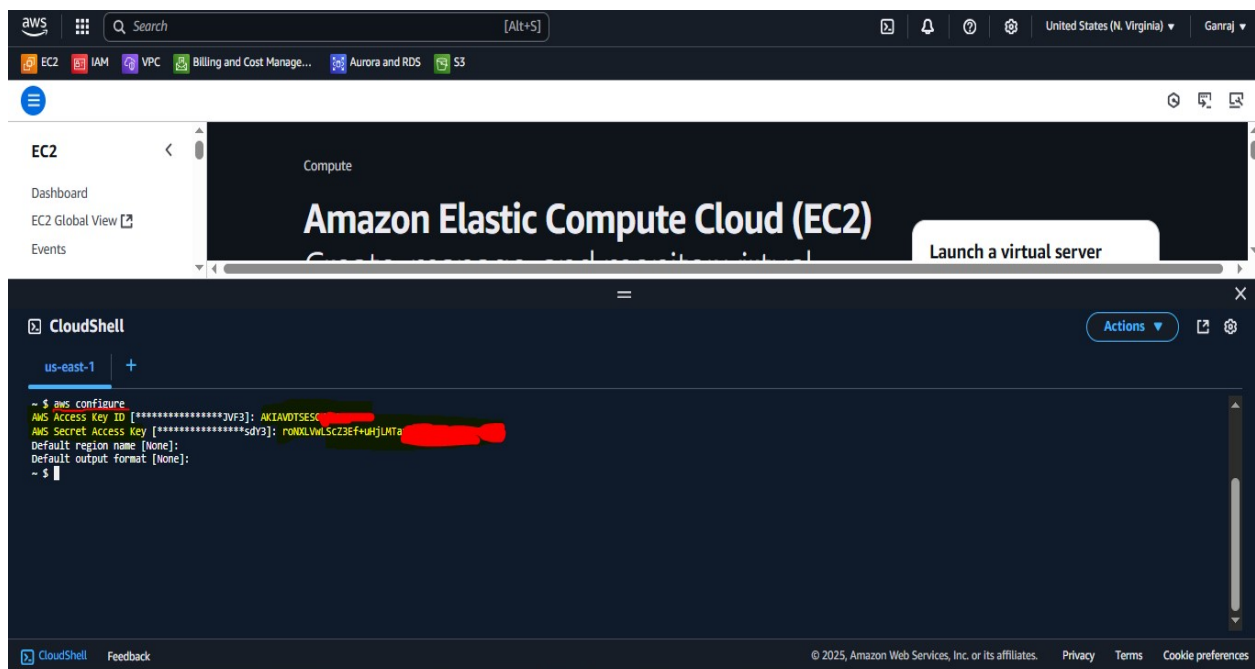
Node Group Created

Now need to configure AWS CLI for using eks cluster

Use Command **aws configure** then you have put access keys two of them.

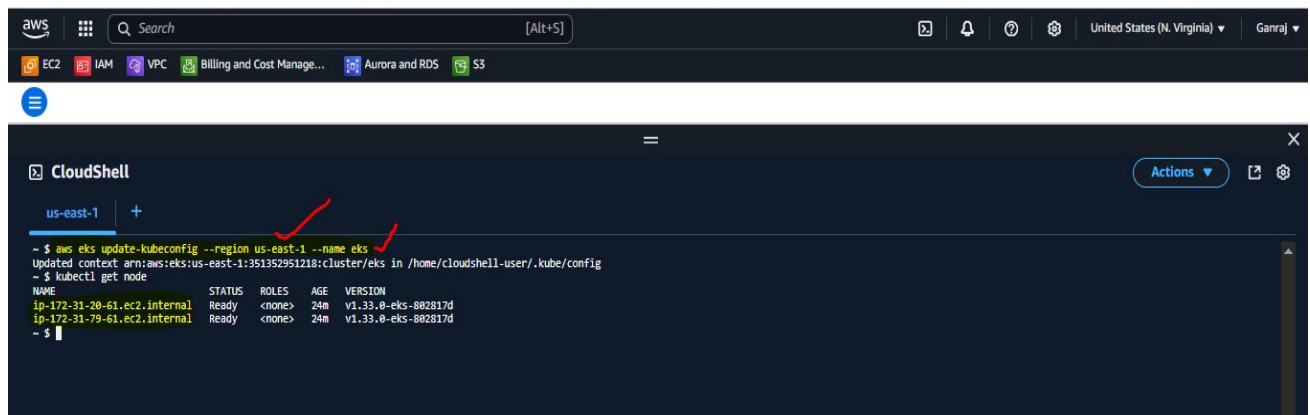
Important that when configuring AWS we have the access key and secret access key to connect your account to AWS CLI.

As you can see in the Snap



Need to connect EKS cluster to AWS CLI for that we use following command.

aws eks update-kubeconfig --region <your-region> --name <your-cluster-name>



Now listen we have only created cluster and nodes there is no image or data (working pods) have been given to cluster for that we have to create a pod for that we use below command

```
kubectl run nginxpod --image nginx:latest
```

Now we will expose ports IPs of the pod :-

The Command

```
kubectl expose pod nginxpod --port=80 --target-port=80 --type=NodePort
```

This command creates a Service to expose a running Pod to the network.

Part	What It Means
<code>kubectl</code>	The Kubernetes CLI tool.
<code>expose</code>	Tells Kubernetes to create a Service that exposes something (Pod, Deployment, ReplicaSet, etc.).
<code>pod nginxpod</code>	The resource to expose: - pod = type of resource. - nginxpod = name of the pod you want to expose.
<code>--port=80</code>	The port number your Service exposes inside the cluster. Other Pods will connect via this port.
<code>--target-port=80</code>	The port number inside the container that traffic will go to. In this example, container's port 80.
<code>--type=NodePort</code>	The type of Service: - NodePort means Kubernetes will assign a port on every node (usually in 30000-32767 range) so you can reach the Pod from outside the cluster.

How It Works

Example Flow:

You already have a Pod:

```
NAME    READY  STATUS
nginxpod 1/1    Running
```

You run:

```
kubectl expose pod nginxpod --port=80 --target-port=80 --type=NodePort
```

Kubernetes creates a Service like this behind the scenes:

```
apiVersion: v1
kind: Service
metadata:
  name: nginxpod
spec:
  selector:
    app: nginxpod # (automatically created label selector)
  ports:
    - port: 80
      targetPort: 80
      nodePort: <assigned-port>
  type: NodePort
```

You can now:

Access the Pod from inside the cluster via nginxpod:80.

Access the Pod from outside the cluster via <NodeIP>:<nodePort>.

Quick Example

If Kubernetes picks nodePort=31333, and your Node IP is 10.0.0.5, you can:

Inside the cluster:

```
nginx
```

```
curl nginxpod:80
```

Outside the cluster:

```
nginx
```

```
curl 10.0.0.5:31333
```

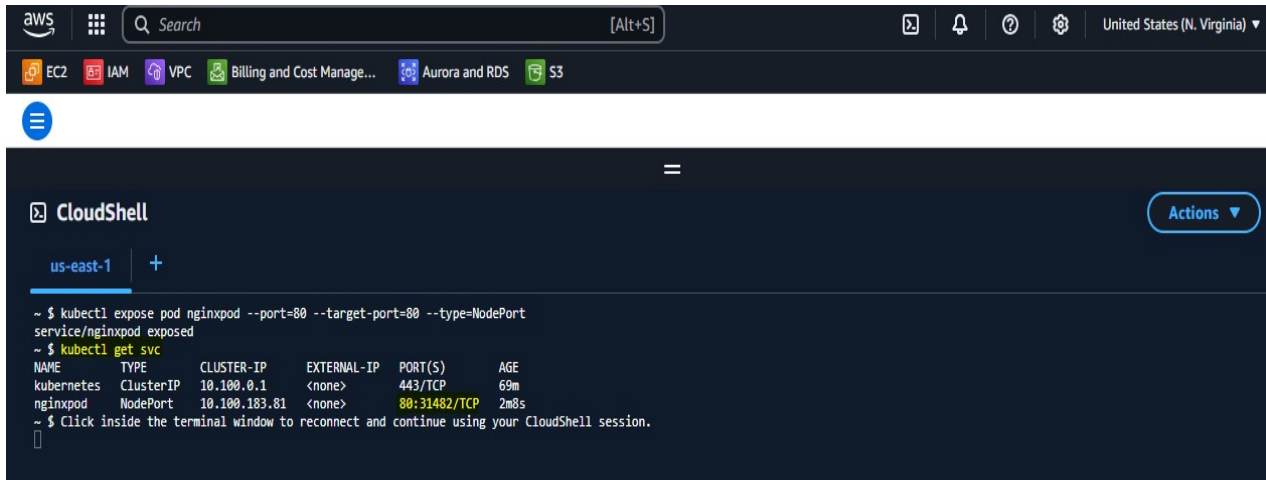

Summary Table of Keys

Flag	Purpose
<code>--port</code>	The port number your Service will open inside the cluster (Service port).
<code>--target-port</code>	The port inside the Pod's container to route traffic to (Container port).
<code>--type</code>	Service type (ClusterIP, NodePort, LoadBalancer). NodePort makes it externally accessible.

IP:

If you omit `--type`, it defaults to ClusterIP (internal only).

You can specify `--name=myservice` if you want a different Service name.



The screenshot shows the AWS CloudShell interface. At the top, there's a navigation bar with AWS services like EC2, IAM, VPC, and others. Below that, the CloudShell terminal is open in the 'us-east-1' region. The terminal shows the following commands and output:

```
~ $ kubectl expose pod nginxpod --port=80 --target-port=80 --type=NodePort
service/nginxpod exposed
~ $ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.100.0.1	<none>	443/TCP	69m
nginxpod	NodePort	10.100.183.81	<none>	80:31482/TCP	2m8s

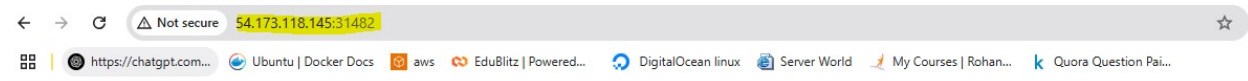
~ \$ Click inside the terminal window to reconnect and continue using your CloudShell session.

And using command

`kubectl get svc`

we can check port no and cluster IP also

So now we can see we have created pod with nginx webserver on port on 31482 and we can Access it through EC2's Public IP



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.