**[Tags](https://community.aws/tags)**

**Featured Spaces**

[Amazon Q](https://community.aws/amazon-q)

[Cost Optimization](https://community.aws/cost-optimization)

[DevOps](https://community.aws/devops)

[Generative AI](https://community.aws/generative-ai)

[Kubernetes](https://community.aws/kubernetes)

[Livestreams](https://community.aws/livestreams)

[Resilience](https://community.aws/resilience)

[Training and Certification](https://community.aws/training)

**Community Programs**

[AWS Heroes](https://aws.amazon.com/developer/community/heroes/?community-heroes-all.sort-by=item%5b%E2%80%A6%5dlocation=*all&awsf.filter-year=*all&awsf.filter-activity=*all)

[AWS Community Builders](https://aws.amazon.com/developer/community/community-builders/?intClick=dev-center-2021_main)

[AWS User Groups](https://aws.amazon.com/developer/community/usergroups/?intClick=dev-center-2021_main)

[Student Communities](https://community.aws/students)

**Site Terms, Privacy, and more.**



**Creating a EKS cluster 1.24 version from scratch using eksctl.**

Create an Amazon EKS cluster 1.24 version from scratch. When your cluster is ready, you can configure your favorite Kubernetes tools, such as kubectl, to communicate with your cluster.

[eks](https://community.aws/tags/eks)[aws](https://community.aws/tags/aws)[kub](https://community.aws/tags/kub)[ec2](https://community.aws/tags/ec2)[devops](https://community.aws/tags/devops)

[AG](https://community.aws/@ashsih)

[Ashish Gajjar](https://community.aws/@ashsih)

Published Dec 27, 2023

|

Last Modified Dec 28, 2023

Like (1)

[Comments](https://community.aws/content/2Z18R1xBAeJbKyigk6oB1mxDYPb" \l "comments)

Share

**Introduction :**

Amazon Elastic Kubernetes Service (Amazon EKS) is a managed service that you can use to run Kubernetes on AWS without needing to install, operate, and maintain your own Kubernetes control plane or nodes. Kubernetes is an open-source system for automating the deployment, scaling, and management of containerized applications.

**Step 1: Login AWS console**

* Open IAM Dashboard
* \*\*\*\* Create a user. username : ashish
* Attach AdministratorAccess policy.
* Create access and secret key.

**Step 2: Create a Ec2 machine**

1. Open a EC2 Dashboard.
2. Launch instance
3. Name and Tags : MyTest
4. Application and OS Image ( AMI ) : Amazon Linux 2023 AMI
5. Instance Type: t2.micro
6. Keypair : ashish.pem
7. Network Settings : VPC, subnet
   * Security Group : 22 - SSH (inbound)
8. Storage : Min 8 GiB , GP3
9. Click Launch instance

**Step 3: Login EC2 instance and configure Access/Secret key.**

Login to EC2 instance.

1

ssh -i "ashish.pem" ec2-user@ec2-52-90-59-5.compute-1.amazonaws.com

Configure Access key and Secret key using AWS CLI.

1  
2  
3  
4  
5

[root@ip-172-31-18-194 ~]# aws configure  
AWS Access Key ID ]: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*4E4R  
AWS Secret Access Key]: [\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HRJx]:  
Default region name]: [Region Name]:  
Default output format]: [None]:

**Step 4: Install eksctl and kubectl using AWS CLI.**

**Setup eksctl**

* Download and extract the latest release
* Move the extracted binary to /usr/local/bin
* Test that your eksclt installation was successful

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12

# for ARM systems, set ARCH to: `arm64`, `armv6` or `armv7`  
ARCH=amd64  
PLATFORM=$(uname -s)\_$ARCH  
  
curl -sLO "https://github.com/eksctl-io/eksctl/releases/latest/download/eksctl\_$PLATFORM.tar.gz"  
  
# (Optional) Verify checksum  
curl -sL "https://github.com/eksctl-io/eksctl/releases/latest/download/eksctl\_checksums.txt" | grep $PLATFORM | sha256sum --check  
  
tar -xzf eksctl\_$PLATFORM.tar.gz -C /tmp && rm eksctl\_$PLATFORM.tar.gz  
  
sudo mv /tmp/eksctl /usr/local/bin

[Ref: https://eksctl.io/installation/](https://eksctl.io/installation/)

**Setup kubectl**

* Download kubectl version
* Grant execution permissions to kubectl executable
* Move kubectl onto /usr/local/bin
* Test that your kubectl installation was successful

1  
2  
3  
4

wget https://amazon-eks.s3.us-west-2.amazonaws.com/1.16.8/2020-04-16/bin/linux/amd64/kubectl  
chmod +x ./kubectl  
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin  
kubectl version --short --client

**Step 5: Create a cluster using eksctl command.**

1

eksctl create cluster --name ashish --version 1.24 --region us-east-1 --nodegroup-name ashish-workers --node-type t3.medium --nodes 2 --nodes-min 1 --nodes-max 4 --managed

**eksctl command Overview :**

* **eksctl create cluster** : Creating a cluster​​ eksctl
* -**-name ashish** :\*\*\*\* Name of Cluster
* **--version 1.24** : EKS cluster version
* **--region us-east-1** : AWS Region Name
* **--nodegroup-name ashish-workers :** Autoscaling Group Name
* **--node-type t3.medium** : instance type
* **--nodes 2** : Desire Node capacity is 2.
* **--nodes-min 1** : Minimum Node capacity is 1.
* -**-nodes-max 4** **--managed** : Maximum capacity is 4.

After executing eksctl command output is as below:

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54

[root@ip-172-31-18-194 ~]# eksctl create cluster --name ashish --version 1.24 --region us-east-1 --nodegroup-name des-max 4 --managed  
2023-12-28 00:36:10 [ℹ] eksctl version 0.167.0  
2023-12-28 00:36:10 [ℹ] using region us-east-1  
2023-12-28 00:36:11 [ℹ] skipping us-east-1e from selection because it doesn't support the following instance type(  
2023-12-28 00:36:11 [ℹ] setting availability zones to [us-east-1d us-east-1a]  
2023-12-28 00:36:11 [ℹ] subnets for us-east-1d - public:192.168.0.0/19 private:192.168.64.0/19  
2023-12-28 00:36:11 [ℹ] subnets for us-east-1a - public:192.168.32.0/19 private:192.168.96.0/19  
2023-12-28 00:36:11 [ℹ] nodegroup "ashish-workers" will use "" [AmazonLinux2/1.24]  
2023-12-28 00:36:11 [ℹ] using Kubernetes version 1.24  
2023-12-28 00:36:11 [ℹ] creating EKS cluster "ashish" in "us-east-1" region with managed nodes  
2023-12-28 00:36:11 [ℹ] will create 2 separate CloudFormation stacks for cluster itself and the initial managed no  
2023-12-28 00:36:11 [ℹ] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-st  
2023-12-28 00:36:11 [ℹ] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false  
2023-12-28 00:36:11 [ℹ] CloudWatch logging will not be enabled for cluster "ashish" in "us-east-1"  
2023-12-28 00:36:11 [ℹ] you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-L  
2 sequential tasks: { create cluster control plane "ashish",  
 2 sequential sub-tasks: {  
 wait for control plane to become ready,  
 create managed nodegroup "ashish-workers",  
 }  
}  
2023-12-28 00:36:11 [ℹ] building cluster stack "eksctl-ashish-cluster"  
2023-12-28 00:36:11 [ℹ] deploying stack "eksctl-ashish-cluster"  
2023-12-28 00:36:41 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:37:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:38:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:39:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:40:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:41:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:42:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:43:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:44:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:45:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:46:11 [ℹ] waiting for CloudFormation stack "eksctl-ashish-cluster"  
2023-12-28 00:48:12 [ℹ] building managed nodegroup stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:48:12 [ℹ] deploying stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:48:13 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:48:43 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:49:33 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:50:19 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:51:01 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:52:35 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 00:52:35 [ℹ] waiting for the control plane to become ready  
2023-12-28 00:52:35 [✔] saved kubeconfig as "/root/.kube/config"  
2023-12-28 00:52:35 [✔] all EKS cluster resources for "ashish" have been created  
2023-12-28 00:52:36 [ℹ] nodegroup "ashish-workers" has 2 node(s)  
2023-12-28 00:52:36 [ℹ] node "ip-192-168-3-161.ec2.internal" is ready  
2023-12-28 00:52:36 [ℹ] node "ip-192-168-48-222.ec2.internal" is ready  
2023-12-28 00:52:36 [ℹ] waiting for at least 1 node(s) to become ready in "ashish-workers"  
2023-12-28 00:52:36 [ℹ] nodegroup "ashish-workers" has 2 node(s)  
2023-12-28 00:52:36 [ℹ] node "ip-192-168-3-161.ec2.internal" is ready  
2023-12-28 00:52:36 [ℹ] node "ip-192-168-48-222.ec2.internal" is ready  
2023-12-28 00:52:37 [ℹ] kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'  
2023-12-28 00:52:37 [✔] EKS cluster "ashish" in "us-east-1" region is ready

EKS cluster successfully launch verification steps:

1. AWS CLI
   * Check how many pods are running

1  
2  
3  
4  
5  
6  
7  
8

[root@ip-172-31-18-194 ~]# kubectl get po -A  
NAMESPACE NAME READY STATUS RESTARTS AGE  
kube-system aws-node-4fdfg 1/1 Running 0 2m50s  
kube-system aws-node-mm84r 1/1 Running 0 2m53s  
kube-system coredns-79989457d9-798tx 1/1 Running 0 10m  
kube-system coredns-79989457d9-7fhzl 1/1 Running 0 10m  
kube-system kube-proxy-rkbzz 1/1 Running 0 2m50s  
kube-system kube-proxy-vfq7k 1/1 Running 0 2m53s

1. AWS Console

* Verify EKS Cluster and version.

A screenshot of a computer

Description automatically generated

EKS-1.24

* Verify ASG Group

A screenshot of a computer

Description automatically generated

ASG

**Step 6: Delete the EKS Cluster**

When you’re done using an Amazon EKS cluster, you should delete the resources associated with it so that you don’t incur any unnecessary costs.

Delete the cluster and its associated nodes with the following command,

1

eksctl delete cluster ashish --region us-east-1

**Output:**

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21

[root@ip-172-31-18-194 ~]# eksctl delete cluster --name ashish  
2023-12-28 01:36:39 [ℹ] deleting EKS cluster "ashish"  
2023-12-28 01:36:39 [ℹ] will drain 0 unmanaged nodegroup(s) in cluster "ashish"  
2023-12-28 01:36:39 [ℹ] starting parallel draining, max in-flight of 1  
2023-12-28 01:36:39 [ℹ] deleted 0 Fargate profile(s)  
2023-12-28 01:36:40 [✔] kubeconfig has been updated  
2023-12-28 01:36:40 [ℹ] cleaning up AWS load balancers created by Kubernetes objects of Kind Service or Ingress   
3 sequential tasks: { delete nodegroup "ashish-workers", delete IAM OIDC provider, delete cluster control plane "as }  
2023-12-28 01:36:40 [ℹ] will delete stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:36:40 [ℹ] waiting for stack "eksctl-ashish-nodegroup-ashish-workers" to get deleted  
2023-12-28 01:36:40 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:37:10 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:37:47 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:39:15 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:40:18 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:41:52 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:42:53 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:44:21 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:45:34 [ℹ] waiting for CloudFormation stack "eksctl-ashish-nodegroup-ashish-workers"  
2023-12-28 01:45:34 [ℹ] will delete stack "eksctl-ashish-cluster"  
2023-12-28 01:45:34 [✔] all cluster resources were deleted

**Conclusion :**

In this blogs, we learned how to setup a Kubernetes cluster on EC2 machine using AWS CLI.