

Kubernetes Task-2

Task Description:

Create the K8s EKS, further you have to do the deployment of the Nginx application and access the application outside the cluster.

Techstacks needs to be used :

- AWS EKS
- EKScctl
- Kubectl

```
Install the latest PowerShell for new features and improvements! https://aka.ms/WindowsPowerShell7.4-LTS

PS C:\WINDOWS\system32> curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/windows/amd64/kubectl.exe.sha256
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 77 100 77 0 0 29 0 0:00:02 0:00:02 --:--:-- 29
PS C:\WINDOWS\system32> Get-FileHash kubectl.exe
Resolve-Path : Cannot find path 'C:\WINDOWS\system32\kubectl.exe' because it does not exist.
At C:\WINDOWS\system32\WindowsPowerShell\v1.0\Modules\Microsoft.PowerShell.Utility\Microsoft.PowerShell.Utility.psm1:110 char:36
+ $pathsToProcess += Resolve-Path $Path | Foreach-Object {
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (C:\WINDOWS\system32\kubectl.exe:String) [Resolve-Path], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.ResolvePathCommand

PS C:\WINDOWS\system32> curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/windows/amd64/kubectl.exe
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 58.8M 100 58.8M 0 0 590k 0 0:01:42 0:01:42 --:--:-- 2430k
PS C:\WINDOWS\system32> Get-FileHash kubectl.exe

Algorithm Hash Path
-----
SHA256 16826333B05D0757F639EB3B4303951550D2B628F290E1722368D925116E9D62 C:\WINDOWS\system32\kubectl.exe

PS C:\WINDOWS\system32>
```

```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> choco install eksctl
chocolatey v2.6.0
Installing the following packages:
eksctl
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading eksctl 0.220.0... 100%

eksctl v0.220.0 [Approved]
eksctl package files install completed. Performing other installation steps.
The package eksctl wants to run 'chocolateyinstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll scripts/[N]o/[P]rint): Y

eksctl is going to be installed in 'C:\ProgramData\chocolatey\lib\eksctl\tools'
Downloading eksctl 64 bit
from 'https://github.com/eksctl-io/eksctl/releases/download/v0.220.0/eksctl_Windows_amd64.zip'
Progress: 100% - Completed download of C:\Users\sheer\AppData\Local\Temp\chocolatey\eksctl\0.220.0\eksctl_Windows_amd64.zip (35.76 MB).
Download of eksctl_Windows_amd64.zip (35.76 MB) completed.
Hashes match.
Extracting C:\Users\sheer\AppData\Local\Temp\chocolatey\eksctl\0.220.0\eksctl_Windows_amd64.zip to C:\ProgramData\chocolatey\lib\eksctl\tools...
C:\ProgramData\chocolatey\lib\eksctl\tools
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type 'refreshenv').
ShimGen has successfully created a shim for eksctl.exe
The install of eksctl was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\eksctl\tools'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> eksctl --version
Error: unknown flag: --version
PS C:\WINDOWS\system32> eksctl version
0.220.0
```

aws Search [Alt+S] Account ID: 3368-3236-1054 sheershinha

Identity and Access Management (IAM) Search IAM

Dashboard

▼ Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings
- Root access management
- Temporary delegation requests

▼ Access reports

- Access Analyzer
- Resource analysis

Harsh info Delete

Summary

ARN
arn:aws:iam::336832361054:user/Harsh

Console access
Enabled without MFA

Access key 1
AKIAU43GF6JPJXEFCINU - Active
Never used. Created today.

Access key 2
Create access key

Created
October 17, 2025, 18:52 (UTC+05:30)

Last console sign-in
1 month ago

Permissions Groups Tags (2) Security credentials Last Accessed

Permissions policies (1) Remove Add permissions

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type
All types

Search

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Directly

aws Search [Alt+S] Account ID: 3368-3236-1054 sheershinha

Identity and Access Management (IAM) Search IAM

Dashboard

▼ Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings
- Root access management
- Temporary delegation requests

▼ Access reports

- Access Analyzer

No MFA devices. Assign an MFA device to improve the security of your AWS environment

Assign MFA device

Access keys (1) Create access key

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

Type	Identifier	Certifications	Created on
	AKIAU43GF6JPJXEFCINU		

Description
eks-access-key

Status
Active

Last used
None

Created
Now

Last used region
N/A

Last used service
N/A

```
sheer@SheershPC MINGW64 ~ (main)
$ aws configure
AWS Access Key ID [*****BKN7]: AKIAU43GF6JPJXEFCINU
AWS Secret Access Key [*****LE9z]: RxWg50fWLPxfVATufbAqn2s4sB+9waeR6N
273F4Y
Default region name [us-east-2]: us-east-2
Default output format [json]:

sheer@SheershPC MINGW64 ~ (main)
$ |
```

```

sheer@SheersPC MINGW64 ~ (main)
$ eksctl create cluster --name demo-cluster-eks --region us-east-2 --node-type t3.micro
2025-12-05 20:25:00 [i] eksctl version 0.220.0
2025-12-05 20:25:00 [i] using region us-east-2
2025-12-05 20:25:02 [i] setting availability zones to [us-east-2c us-east-2a us-east-2b]
2025-12-05 20:25:02 [i] subnets for us-east-2c - public:192.168.0.0/19 private:192.168.96.0/19
2025-12-05 20:25:02 [i] subnets for us-east-2a - public:192.168.32.0/19 private:192.168.128.0/19
2025-12-05 20:25:02 [i] subnets for us-east-2b - public:192.168.64.0/19 private:192.168.160.0/19
2025-12-05 20:25:02 [i] nodegroup "ng-Sa0c83eb" will use [AmazonLinux2023/1.32]
2025-12-05 20:25:02 [i] Auto Mode will be enabled by default in an upcoming release of eksctl. This means managed node groups and managed networking add-ons will no longer be created by default. To maintain cu
rent behavior, explicitly set 'autoModeConfig.enabled: false' in your cluster configuration. Learn more: https://eksctl.io/usage/auto-mode/
2025-12-05 20:25:02 [i] using Kubernetes version 1.32
2025-12-05 20:25:02 [i] creating EKS cluster "demo-cluster-eks" in "us-east-2" region with managed nodes
2025-12-05 20:25:02 [i] will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2025-12-05 20:25:02 [i] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-east-2 --cluster=demo-cluster-eks'
2025-12-05 20:25:02 [i] Kubernetes API endpoint access will use default of [publicAccess=true, privateAccess=false] for cluster "demo-cluster-eks" in "us-east-2"
2025-12-05 20:25:02 [i] Cloudwatch logging will not be enabled for cluster "demo-cluster-eks" in "us-east-2"
2025-12-05 20:25:02 [i] you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=us-east-2 --cluster=demo-cluster-eks'
2025-12-05 20:25:02 [i] default addons metrics-server, vpc-cni, kube-proxy, coredns were not specified, will install them as EKS addons
2025-12-05 20:25:02 [i]
2 sequential tasks: { create cluster control plane "demo-cluster-eks",
  2 sequential sub-tasks: {
    1 task: { create addons },
    wait for control plane to become ready,
  },
  create managed nodegroup "ng-Sa0c83eb",
}
2025-12-05 20:25:02 [i] building cluster stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:25:04 [i] deploying stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:25:34 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:26:05 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:27:06 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:28:08 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:29:09 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:30:11 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:31:12 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:32:13 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:33:15 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:34:16 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:34:24 [i] recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl cannot configure the requested permissions; the recommended way to provide IAM per
missions for "vpc-cni" addon is via pod identity associations; after addon creation is completed, add all recommended policies to the config file, under 'addon.PodIdentityAssociations', and run 'eksctl update a
addon'
2025-12-05 20:34:24 [i] creating addon: vpc-cni
2025-12-05 20:34:24 [i] successfully created addon: vpc-cni
2025-12-05 20:34:25 [i] creating addon: kube-proxy
2025-12-05 20:34:27 [i] successfully created addon: kube-proxy
2025-12-05 20:34:28 [i] creating addon: coredns
2025-12-05 20:34:28 [i] successfully created addon: coredns
2025-12-05 20:36:35 [i] building managed nodegroup stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:36:37 [i] deploying stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:36:38 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:37:09 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:37:46 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:38:56 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:38:56 [i] waiting for the control plane to become ready

```

```

MINGW64/C/Users/sheer
2025-12-05 20:25:34 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:26:05 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:27:06 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:28:08 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:29:09 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:30:11 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:31:12 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:32:13 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:33:15 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:34:16 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-cluster"
2025-12-05 20:34:24 [i] recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl cannot configure the requested permissions; the recommended way to provide IAM per
missions for "vpc-cni" addon is via pod identity associations; after addon creation is completed, add all recommended policies to the config file, under 'addon.PodIdentityAssociations', and run 'eksctl update a
addon'
2025-12-05 20:34:24 [i] creating addon: vpc-cni
2025-12-05 20:34:24 [i] successfully created addon: vpc-cni
2025-12-05 20:34:25 [i] creating addon: kube-proxy
2025-12-05 20:34:27 [i] successfully created addon: kube-proxy
2025-12-05 20:34:28 [i] creating addon: coredns
2025-12-05 20:34:28 [i] successfully created addon: coredns
2025-12-05 20:36:35 [i] building managed nodegroup stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:36:37 [i] deploying stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:36:38 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:37:09 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:37:46 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:38:56 [i] waiting for CloudFormation stack "eksctl-demo-cluster-eks-nodegroup-ng-Sa0c83eb"
2025-12-05 20:38:56 [i] waiting for the control plane to become ready
2025-12-05 20:38:57 [x] saved kubeconfig as "C:\Users\sheer\.kube\config"
2025-12-05 20:38:57 [x] no tasks
2025-12-05 20:38:57 [x] all EKS cluster resources for "demo-cluster-eks" have been created
2025-12-05 20:38:58 [x] nodegroup "ng-Sa0c83eb" has 2 node(s)
2025-12-05 20:38:58 [x] node "ip-192-168-19-45.us-east-2.compute.internal" is ready
2025-12-05 20:38:58 [x] node "ip-192-168-91-134.us-east-2.compute.internal" is ready
2025-12-05 20:38:58 [x] waiting for at least 2 node(s) to become ready in "ng-Sa0c83eb"
2025-12-05 20:38:59 [x] nodegroup "ng-Sa0c83eb" has 2 node(s)
2025-12-05 20:38:59 [x] node "ip-192-168-19-45.us-east-2.compute.internal" is ready
2025-12-05 20:38:59 [x] node "ip-192-168-91-134.us-east-2.compute.internal" is ready
2025-12-05 20:38:59 [x] created 1 managed nodegroup(s) in cluster "demo-cluster-eks"
2025-12-05 20:39:00 [i] creating addon: metrics-server
2025-12-05 20:39:01 [i] successfully created addon: metrics-server
2025-12-05 20:39:08 [i] kubectl command should work with "C:\Users\sheer\.kube\config", try 'kubectl get nodes'
2025-12-05 20:39:08 [x] EKS cluster "demo-cluster-eks" in "us-east-2" region is ready

sheer@SheersPC MINGW64 ~ (main)
$ kubectl get pos
error: the server doesn't have a resource type "pos"

sheer@SheersPC MINGW64 ~ (main)
$ kubectl get pods
No resources found in default namespace.

sheer@SheersPC MINGW64 ~ (main)
$ kubectl get nodes
NAME                                STATUS    ROLES    AGE     VERSION
ip-192-168-19-45.us-east-2.compute.internal Ready    <none>   2m31s   v1.32.9-eks-ecaa3a6
ip-192-168-91-134.us-east-2.compute.internal Ready    <none>   2m31s   v1.32.9-eks-ecaa3a6

```

```

Kubernetes > EKS > ** Deployment.yml > {} spec > {} template > {} spec
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: nginx-deployment
5  spec:
6    replicas: 2
7    selector:
8      matchLabels:
9        app: nginx-deployment
10   template:
11     metadata:
12       labels:
13         app: nginx-deployment
14   spec:
15     containers:
16       - name: nginx-deployment
17         image: nginx:alpine
18         ports:
19           - containerPort: 80
20

```

```

PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> eksctl get cluster
NAME      REGION    EKSCluster CREATED
demo-eks  us-east-2 True
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get nodes
NAME                                             STATUS    ROLES    AGE    VERSION
ip-192-168-39-45.us-east-2.compute.internal    Ready     <none>    2m12s  v1.32.9-eks-ecaa3a6
ip-192-168-82-79.us-east-2.compute.internal    Ready     <none>    2m12s  v1.32.9-eks-ecaa3a6
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> ls

Directory: C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS

Mode                LastWriteTime         Length Name
----                -
-a---             06-12-2025   15:13           368 Deployment.yml
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl apply -f deployment.yml
deployment.apps/nginx-deployment created
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get kube-system
error: the server doesn't have a resource type "kube-system"
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get pods
NAME                                     READY    STATUS    RESTARTS   AGE
nginx-deployment-6d98745f75-26crp       1/1      Running   0           35s
nginx-deployment-6d98745f75-4hzfw       1/1      Running   0           35s
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get pods -n kube-system
NAME                                     READY    STATUS    RESTARTS   AGE
aws-node-pmdxn                          2/2      Running   0           3m48s
aws-node-stvdz                          2/2      Running   0           3m48s
coredns-64dfc67578-68kvr                1/1      Running   0           7m20s
coredns-64dfc67578-sj86h                1/1      Running   0           7m20s
kube-proxy-l25cw                        1/1      Running   0           3m48s
kube-proxy-zfdzf                        1/1      Running   0           3m48s
metrics-server-5c9ffc7c9-86nbf          1/1      Running   0           2m51s
metrics-server-5c9ffc7c9-npx86          1/1      Running   0           2m51s

```

```

Kubernetes > EKS > xxx service.yml > {} spec > {} ports > {} 0 > abc protocol
io.k8s.api.core.v1.Service (v1@service.json)
1  apiVersion: v1
2  kind: Service
3  metadata:
4    | name: nginx-service
5  spec:
6    | type: LoadBalancer
7    | selector:
8      | app: nginx-deployment
9    | ports:
10   - port: 90
11     targetPort: 80
12   protocol: TCP

```

```

PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl apply -f service.yml
service/nginx-service created
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get svc

```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.100.0.1	<none>	443/TCP	83m
nginx-service	LoadBalancer	10.100.82.246	aa6a98cd6cbef41fba07ae0c913c5a57-667152870.us-east-2.elb.amazonaws.com	90:31203/TCP	14s

```

PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> curl http://aa6a98cd6cbef41fba07ae0c913c5a57-667152870.us-east-2.elb.amazonaws.com:90
curl: (6) Could not resolve host: aa6a98cd6cbef41fba07ae0c913c5a57-667152870.us-east-2.elb.amazonaws.com
PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> kubectl get endpoints

```

NAME	ENDPOINTS	AGE
kubernetes	192.168.114.155:443,192.168.172.26:443	84m
nginx-service	192.168.61.199:80,192.168.90.211:80	72s

```

PS C:\Users\sheer\Documents\kubernetes\Kubernetes\EKS> curl http://aa6a98cd6cbef41fba07ae0c913c5a57-667152870.us-east-2.elb.amazonaws.com:90
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>

```