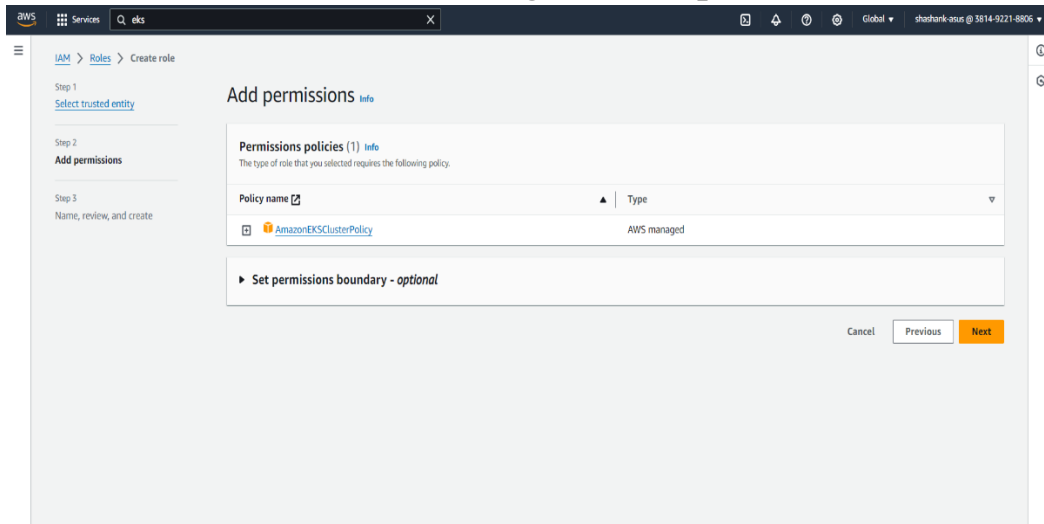


Name : Nirmal Omkar Maruti

CDEC B-24

Task : Hosting of nginx and tomcat via manifest file

1. Create IAM role for EKS and give EKS permission.



2. Create IAM role for EC2 and give permission as below.

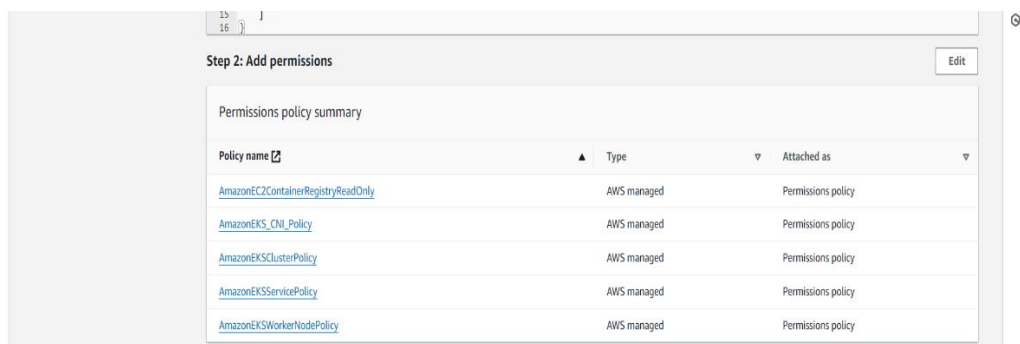
AmazonEC2ContainerRegistryReadOnly

AmazonEKS_CNI_Policy

AmazonEKSClusterPolicy

AmazonEKSServicePolicy

AmazonEKSWorkerNodePolicy



3. Create a cluster.

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

Review and create

Step 1: Cluster

Cluster configuration

Name

shashank-eks-cluster

Kubernetes version

1.29

Cluster service role

arn:aws:iam::381492218806:role/shashank-eks-role

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

Networking

These properties cannot be changed after the cluster is created.

VPC

vpc-0f692367e7315726d

Subnets

subnet-0b975bf6e85fbfeac
subnet-04a74a5846e4c229c

Security groups

sg-00347fd46666f2e2

Cluster IP address family

IPv4

Cluster endpoint access

API server endpoint access

Public

Public access source allowlist

0.0.0.0/0

Step 3: Observability

Control plane logging

API server

off

Audit

off

Authenticator

off

Controller manager

off

Scheduler

off

Step 4: Add-ons

Selected add-ons

Find add-on

< 1 >

Add-on name

Type

Status

coredns

networking

Installed by default

eks-pod-identity-agent

security

Ready to install

kube-proxy

networking

Installed by default

vpc-cni

networking

Installed by default

Step 5: Versions

Selected add-ons version

Add-on name

Version

coredns

v1.11.1-eksbuild.4

Add-on name

Version

kube-proxy

v1.29.0-eksbuild.1

Add-on name

Version

vpc-cni

v1.16.0-eksbuild.1

Add-on name

Version

eks-pod-identity-agent

v1.2.0-eksbuild.1

Cancel

Previous

Create

CloudShell

Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

4. After creation of cluster add node group to it.

The screenshot shows the AWS Management Console interface for adding a node group to an EKS cluster. The breadcrumb navigation is: EKS > Clusters > shashank-eks-cluster > Node groups > Add node group. The left sidebar shows the steps: Step 1: Configure node group, Step 2: Set compute and scaling configuration (active), Step 3: Specify networking, and Step 4: Review and create.

Set compute and scaling configuration

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 2 (AL2_x86_64)

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 GB Network: Up to 5 Gigabit Max ENI: 3 Max IP: 18

Disk size
Select the size of the attached EBS volume for each node.
20 GIB

Node group scaling configuration

Desired size
Set the desired number of nodes that the group should launch with initially.
1 nodes
Desired node size must be greater than or equal to 0

Minimum size
Set the minimum number of nodes that the group can scale in to.
1 nodes
Minimum node size must be greater than or equal to 0

Maximum size
Set the maximum number of nodes that the group can scale out to.
2 nodes
Maximum node size must be greater than or equal to 1 and cannot be lower than the minimum size

Node group update configuration [info](#)

Maximum unavailable
Set the maximum number or percentage of unavailable nodes to be tolerated during the node group version update.

☒ Number
Enter a number

☐ Percentage
Specify a percentage

Value
1 node
Node count must be greater than 0.

Cancel Previous Next

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

5. After adding node group to the cluster open cloud shell and configure it using command.

aws configure

(add your access key, secret access key, region)

```
us-west-1

[cloudshell-user@ip-10-4-18-207 ~]$ aws configure
AWS Access Key ID [*****AFXB]: 
AWS Secret Access Key [*****3fh]: 
Default region name [None]: 
Default output format [None]: 
[cloudshell-user@ip-10-4-18-207 ~]$ aws eks update-kubeconfig --region us-west-1 --name shashank-eks-cluster
Added new context arn:aws:eks:us-west-1:381492218806:cluster/shashank-eks-cluster to /home/cloudshell-user/.kube/config
[cloudshell-user@ip-10-4-18-207 ~]$ kubectl cluster-info
Kubernetes control plane is running at https://3c992a1aee6182203aff45986f808873.sk1.us-west-1.eks.amazonaws.com
CoreDNS is running at https://3c992a1aee6182203aff45986f808873.sk1.us-west-1.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[cloudshell-user@ip-10-4-18-207 ~]$
```

6. Create pod.yml & service.yml file in your VS code and upload files on your git repo.
7. Create pod file for nginx and tomcat with extension pod.yml.

Pod.yml

```
! pods.yml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: demopod
5    labels:
6      app: new-app
7  spec:
8    containers:
9      - name: nginx
10       image: nginx:latest
11       ports:
12         - containerPort: 80
13           protocol: TCP
14       - name: tomcat
15         image: tomcat:latest
16         ports:
17           - containerPort: 8080
18             protocol: TCP
19
20
```

8. After completing the script create pod using command.

#git clone <your-git-repo-link>

#git clone https://github.com/Nirmalomkar/Kubernetes.git
(in my case my file present in Kubernetes 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-10-6-16-163 kubernetes]$ kubectl apply -f pods.yml
pod/demopod created
[cloudshell-user@ip-10-6-16-163 kubernetes]$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
demopod	2/2	Running	0	29s

```
[cloudshell-user@ip-10-6-16-163 kubernetes]$ kubectl get -o wide pods
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
demopod	2/2	Running	0	43s	172.31.26.254	ip-172-31-19-77.us-west-1.compute.internal	<none>	<none>

```
[cloudshell-user@ip-10-6-16-163 kubernetes]$ kubectl describe pods
```

```
Name: demopod
Namespace: default
Priority: 0
Service Account: default
Node: ip-172-31-19-77.us-west-1.compute.internal/172.31.19.77
Start Time: Thu, 28 Mar 2024 07:53:04 +0000
Labels: app=new-app
Annotations: <none>
Status: Running
IP: 172.31.27.238
IPs:
  IP: 172.31.27.238
Containers:
  nginx:
    Container ID: containerd://9ceae70ff287f6242f57e61142515153226e98e543d9cc3251e65a4286c4a6e8
    Image: nginx:latest
    Image ID: docker.io/library/nginx@sha256:6db391d1c0cfb30588ba0bf72ea999404f2764feb0f1f196acd5867ac7efa7e
    Port: 80/TCP
    Host Port: 0/TCP
    State: Running
      Started: Thu, 28 Mar 2024 07:53:05 +0000
    Ready: True
    Restart Count: 0
    Environment: <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-gwfhm (ro)
  tomcat:
```

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

Service.yml

```
! service.yml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: demoxyz
5  spec:
6    selector:
7      app: new-app
8    type: NodePort
9    ports:
10     - port: 80
11       targetPort: 80
12       name: nginx
13       protocol: TCP
14
15     - port: 8080
16       targetPort: 8080
17       name: tomcat
18       protocol: TCP
19
```

10. After writing the script use git pull command to pull your service.yml file

git pull

```
[cloudshell-user@ip-10-6-16-163 kubernetes]$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 942 bytes | 471.00 KiB/s, done.
From https://github.com/shashanksharma1309/kubernetes
   f47670a..863338c  main      -> origin/main
Updating f47670a..863338c
Fast-forward
 service.yml | 2 ++
1 file changed, 2 insertions(+)
```

11. Use commands to create service.

```
#ls
```

```
#kubectl apply -f service.yml
```

```
#kubectl get svc (services)
```

```
[cloudshell-user@ip-10-6-16-163 kubernetes]$ kubectl apply -f service.yml
service/demoxyz created
[cloudshell-user@ip-10-6-16-163 kubernetes]$ kubectl get svc
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
demoxyz       NodePort    10.100.59.75 <none>        80:31648/TCP,8080:30444/TCP 15s
kubernetes    ClusterIP   10.100.0.1   <none>        443/TCP          147m
[cloudshell-user@ip-10-6-16-163 kubernetes]$
```

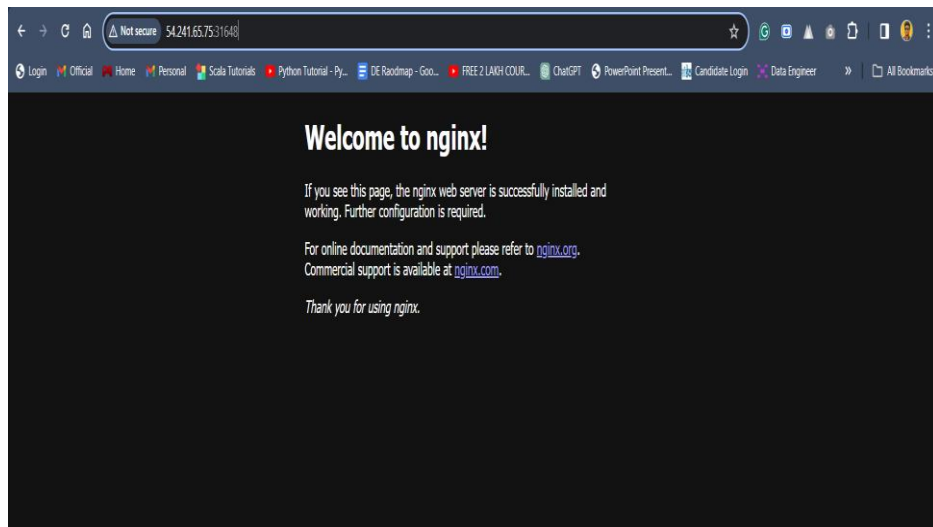
12. After creation of service hit the IP of your instance which is created while creation of node group.

For nginx

<instance-IP>:<port-IP>

In my case;

54.241.65.75:31648



13. For tomcat;

54.241.65.75:30444

