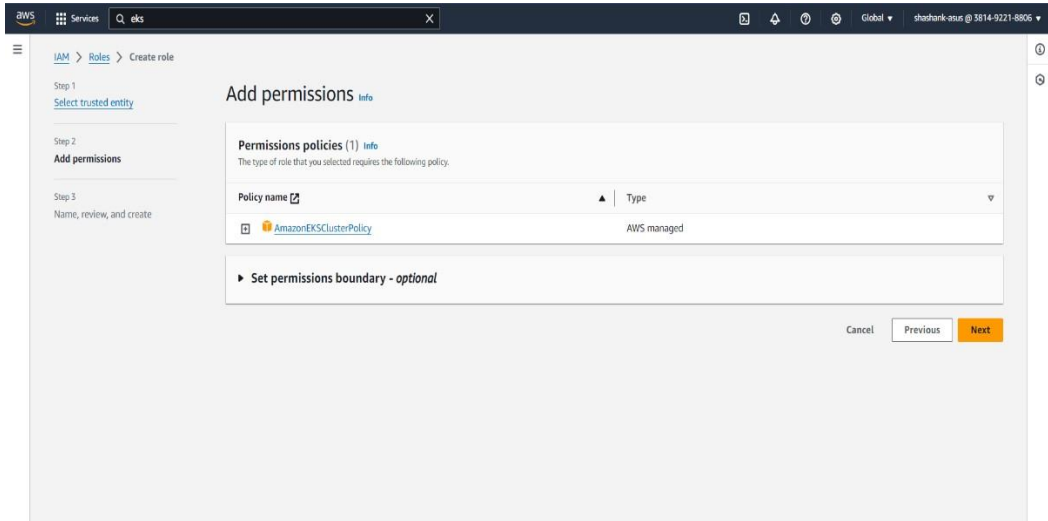


CDEC B24

Name: Kunal vijay Chavhan

Task: Hosting of nginx and tomcat using manifest file

1. Create IAM role for EKS and give EKS permission



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

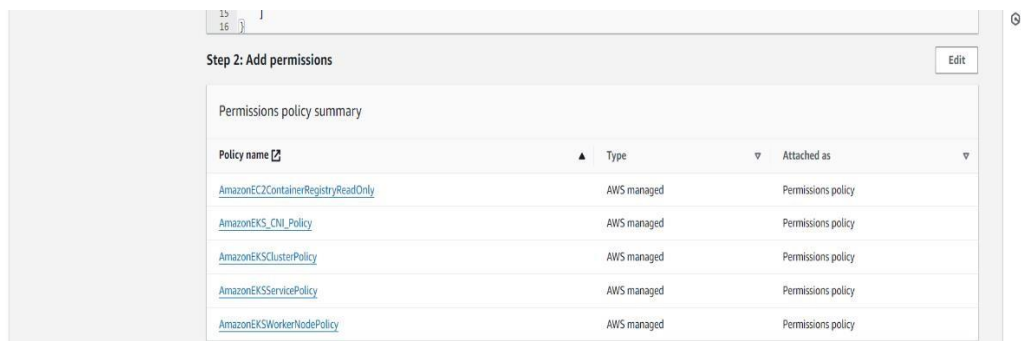
AmazonEC2ContainerRegistryReadOnly

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

Edit

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

Edit

Networking

These properties cannot be changed after the cluster is created.

| | | |
|---------------------------|--|----------------------|
| VPC | Subnets | Security groups |
| vpc-0f692367e7315726d | subnet-0b975bf6e85fbfeac subnet-04a74a5846e4c229c | sg-00347fdf46666f2e2 |
| Cluster IP address family | IPv4 | |

Cluster endpoint access

| | |
|----------------------------|--------------------------------|
| API server endpoint access | Public access source allowlist |
| Public | 0.0.0.0/0 |

Step 3: Observability

Edit

Control plane logging

| | | |
|--------------------|-----------|---------------|
| API server | Audit | Authenticator |
| off | off | off |
| Controller manager | Scheduler | |
| off | off | |

Step 4: Add-ons

Edit

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

Edit

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.

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.

Capacity type
Select the capacity purchase option for this node group.

Instance types [info](#)
Select instance types you prefer for this node group.

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

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

Node group scaling configuration

Desired size
Set the desired number of nodes that the group should launch with initially.
 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.
 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.
 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
 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)

```
[cloudshell-user@ip-10-140-121-42 ~]$ aws configure
AWS Access Key ID [None]: A
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl cluster-info
-bash: kubectl: command not found
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl cluster-info
-bash: kubectl: command not found
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl cluster-info
E0329 05:53:19.209542    234 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused
E0329 05:53:19.211173    234 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused
E0329 05:53:19.211858    234 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused
E0329 05:53:19.216686    234 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused
E0329 05:53:19.219099    234 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
The connection to the server localhost:8080 was refused - did you specify the right host or port?
[cloudshell-user@ip-10-140-121-42 ~]$ aws eks --region us-east-1 update-kubeconfig --name EKS-cluster
Added new context arn:aws:eks:us-east-1:471112957025:cluster/EKS-cluster to /home/cloudshell-user/.kube/config
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl cluster-info
Kubernetes control plane is running at https://9EBF926C1E08FCCC100888848E29A25C.gr7.us-east-1.eks.amazonaws.com
CoreDNS is running at https://9EBF926C1E08FCCC100888848E29A25C.gr7.us-east-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'.
```

6. Now create a pod and expose it after that get port configure it using command of nginx.

#aws eks --region us-east-1 update-kubeconfig --name eks-cluster(cluster name).

#kubectl run nginx --image=nginx

kubectl expose pod nginx --port=80 --target-port=80 --type=NodePort

#kubectl get service

```
pod/nginx deleted
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl run nginx --image=nginx
pod/nginx created
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
nginx     1/1     Running   0           5s
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl expose pod nginx --port=80 --target-port=80 --type=NodePort
service/nginx exposed
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get svc
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes  ClusterIP  10.100.0.1    <none>         443/TCP           38m
nginx      NodePort   10.100.71.182 <none>         80:32560/TCP      18s
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get -o wide nodes
```

7. Know Host the IP Address of instance with the container port on web page.



8. Know for Tomcat create a pod and expose it after that get port configure it using command.

#kubectl run tomcat --image=tomcat

kubectl expose pod tomcat --port=8080 --target-port=8080 --type=NodePort

#kubectl get service

```
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl run tomcat --image=tomcat
pod/tomcat created
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl expose pod tomcat --port=8080 --target-port=8080 --type=NodePort
service/tomcat exposed
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get service
NAME      TYPE      CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes ClusterIP  10.100.0.1      <none>          443/TCP          47m
nginx     NodePort   10.100.71.182   <none>          80:32609/TCP     9m54s
tomcat    NodePort   10.100.229.66   <none>          8080:30610/TCP   9s
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get pod
NAME      READY   STATUS    RESTARTS   AGE
nginx     1/1     Running   0           11m
tomcat    1/1     Running   0           72s
[cloudshell-user@ip-10-140-121-42 ~]$ kubectl get -o wide nodes
NAME                                STATUS    ROLES    AGE   VERSION    INTERNAL-IP    EXTERNAL-IP    OS-IMAGE             KERNEL-VERSION    CONTAINER-RUNTIME
ip-172-31-85-108.ec2.internal        Ready    <none>    42m   v1.29.0-eks-5e0fdde  172.31.85.108  3.95.221.247    Amazon Linux 2       5.10.210-201.852.amzn2.x86_64  containerd://1.7.11
```

9. Know Host the IP Address of instance with the container port on web page.

