ADVANCE DEVOPS CASE STUDY

Kubernetes Application Deployment on AWS using Cloud9 IDE

1. Introduction

The purpose of this case study is to set up a Kubernetes cluster on AWS using the Cloud9 IDE, deploy a sample application (like Nginx) using kubectl, and ensure that the application runs successfully. Additionally, we will verify the deployment by accessing the application through a NodePort or LoadBalancer service.

Technologies and Tools

- **AWS Cloud9 IDE**: A cloud-based IDE providing an environment to work directly on the cloud infrastructure.
- **Kubernetes**: An open-source platform for automating deployment, scaling, and management of containerized applications.
- **Kubectl**: The command-line tool to manage Kubernetes clusters.
- **AWS EKS**: Elastic Kubernetes Service, a managed Kubernetes service on AWS.
- **Nginx**: A web server that will be deployed as a sample application.

2. Setup and Configuration

2.1 Installing and Configuring Kubectl on Cloud9 IDE

1. **Create an AWS Cloud9 environment**:
    - Open the AWS Management Console and navigate to Cloud9.
    - Create a new environment by configuring the necessary parameters (EC2 instance size, networking, etc.).
2. **Install kubectl**:

On the Cloud9 terminal, install kubectl using the following commands:

```
curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.22.2/2021-10-04/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin
kubectl version --short --client
```

- This installs kubectl, the Kubernetes command-line tool.

Install AWS CLI: Ensure the AWS CLI is installed to interact with AWS services.

```
sudo apt install awscli -y
aws --version
```

3.

Configure AWS credentials:

```
aws configure
```

4. Provide the required AWS access keys to authenticate with AWS services.

Install eksctl: eksctl is the official CLI for managing EKS clusters.

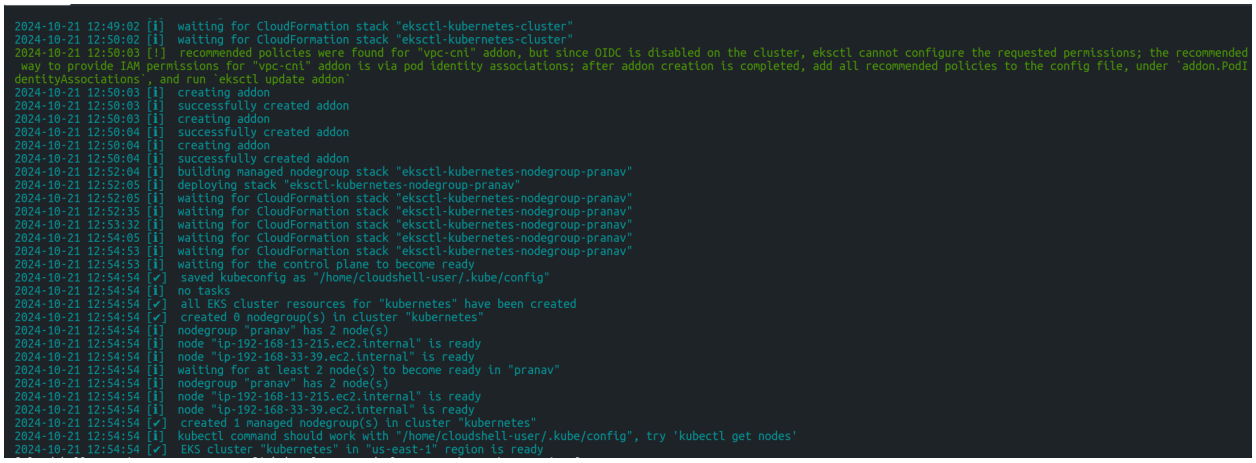
```
curl -sL https://eksctl.io/install | sh
eksctl version
```

5.

2.2 Create the Kubernetes Cluster on AWS EKS

Create an EKS Cluster: Use eksctl to create a Kubernetes cluster on AWS:

```
eksctl create cluster --name my-cluster --region us-west-2 --nodegroup-name standard-workers --node-type t2.medium --nodes 3
```



1. This command creates an EKS cluster with 3 worker nodes.

Configure kubectl to use the EKS Cluster: After the cluster is created, configure kubectl to interact with it:

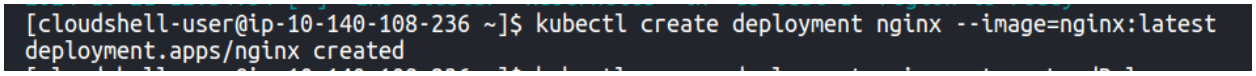
```
aws eks --region us-west-2 update-kubeconfig --name my-cluster
```

2.

2.3 Deploy a Sample Application (Nginx)

Deploy Nginx as a sample application: Create an Nginx deployment using kubectl:

```
kubectl create deployment nginx --image=nginx
```



1. This command deploys an Nginx web server.

Expose the Nginx deployment: To access the Nginx server, expose it using a NodePort or LoadBalancer service:

```
kubectl expose deployment nginx --type=NodePort --port=80
```

or, for a LoadBalancer (on AWS):

```
kubectl expose deployment nginx --type=LoadBalancer --port=80
```

```
[cloudshell-user@ip-10-140-108-236 ~]$ kubectl expose deployment nginx --type=LoadBalancer --port=80
service/nginx exposed
```

2.

Verify the deployment: To check if the deployment is running:

```
kubectl get pods
kubectl get svc
```

```
[cloudshell-user@ip-10-140-108-236 ~]$ kubectl get svc
NAME         TYPE           CLUSTER-IP     EXTERNAL-IP                                                                PORT(S)        AGE
kubernetes   ClusterIP      10.100.0.1     <none>                                                                     443/TCP        58m
nginx        LoadBalancer   10.100.229.5   a2a71631b805b4781ab5758b6cef05c6-125741757.us-east-1.elb.amazonaws.com   80:31218/TCP   48m
[cloudshell-user@ip-10-140-108-236 ~]$
```

3. The output should show the nginx pod running and the Service exposing it.

3. Accessing the Application

NodePort: If you used a NodePort, you can access the application using the external IP of any node and the assigned port:

```
kubectl get nodes -o wide
Kubectl get deployments
```

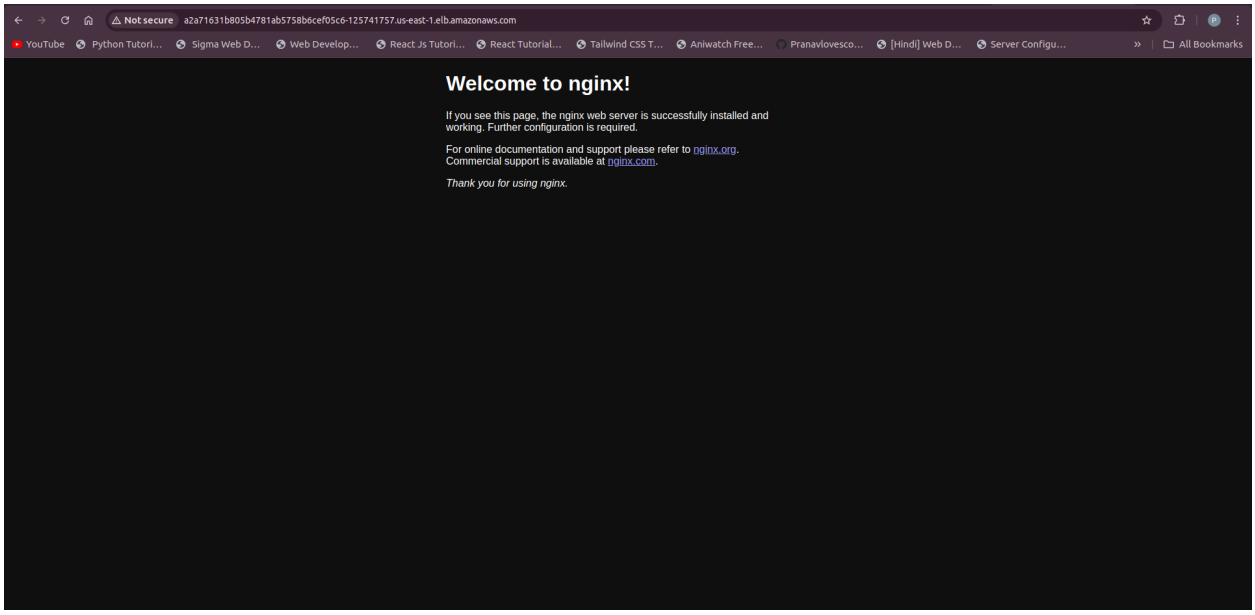
```
[cloudshell-user@ip-10-140-108-236 ~]$ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nginx   1/1     1            1           50m
[cloudshell-user@ip-10-140-108-236 ~]$
```

- Open a web browser and paste the external ip shown as an output of the previous command.

LoadBalancer: If you used a LoadBalancer, check the external IP of the service:

```
kubectl get svc
```

- Open a web browser and paste the external ip shown as an output of the previous command.



4. Conclusion

This case study successfully demonstrates the setup of a Kubernetes cluster on AWS using Cloud9 IDE. We installed and configured kubectl, created an EKS cluster, deployed an Nginx application, and verified its accessibility using both NodePort and LoadBalancer services. Kubernetes simplifies application management and deployment, especially in cloud environments like AWS EKS.