Lab Report: EKS Cluster Deployment and Decommissioning

Done by: Marien Baloitcha

Environment: AWS EKS via eksctl on Linux CentOS 9 stream VM

1. Objective

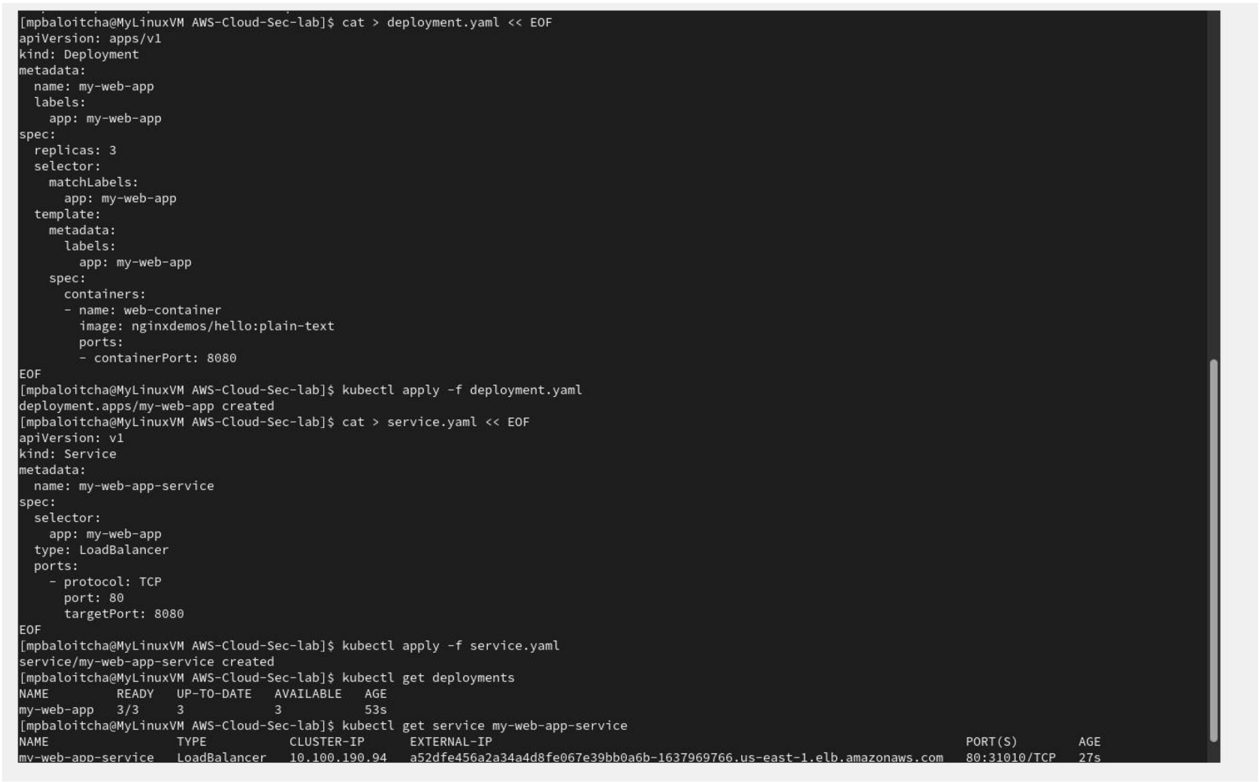
The primary goal of this lab was to successfully provision a production-ready, highly available Kubernetes cluster on Amazon Elastic Kubernetes Service (EKS) and deploy a containerized web application, demonstrating public accessibility and verifying a clean decommissioning process to manage cloud costs.

2. Procedure Summary

The lab followed a standard Infrastructure-as-Code and Kubernetes deployment lifecycle:

A. Infrastructure Provisioning

The EKS cluster was created using the eksctl command-line utility, which manages the entire provisioning process, including the EKS control plane and worker node group infrastructure on AWS CloudFormation.



```
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ cat > deployment.yaml << EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-web-app
  labels:
    app: my-web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-web-app
  template:
    metadata:
      labels:
        app: my-web-app
    spec:
      containers:
      - name: web-container
        image: nginxdemos/hello:plain-text
        ports:
        - containerPort: 8080
EOF
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ kubectl apply -f deployment.yaml
deployment.apps/my-web-app created
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ cat > service.yaml << EOF
apiVersion: v1
kind: Service
metadata:
  name: my-web-app-service
spec:
  selector:
    app: my-web-app
  type: LoadBalancer
  ports:
    - protocol: TCP
      port: 80
      targetPort: 8080
EOF
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ kubectl apply -f service.yaml
service/my-web-app-service created
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ kubectl get deployments
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
my-web-app   3/3     3            3           53s
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ kubectl get service my-web-app-service
NAME                 TYPE           CLUSTER-IP      EXTERNAL-IP                                                               PORT(S)        AGE
my-web-app-service   LoadBalancer   10.100.190.94   a52dfe456a2a34a4d8fe067e39bb0a6b-1637969766.us-east-1.elb.amazonaws.com   80:31010/TCP   27s
```

Cluster Specification:

- **Cluster Name:** wonderful-sheepdog-1760203957
- **Region:** us-east-1
- **Worker Nodes:** Two EC2 instances (t3.medium) managed by a single nodegroup.

B. Application Deployment

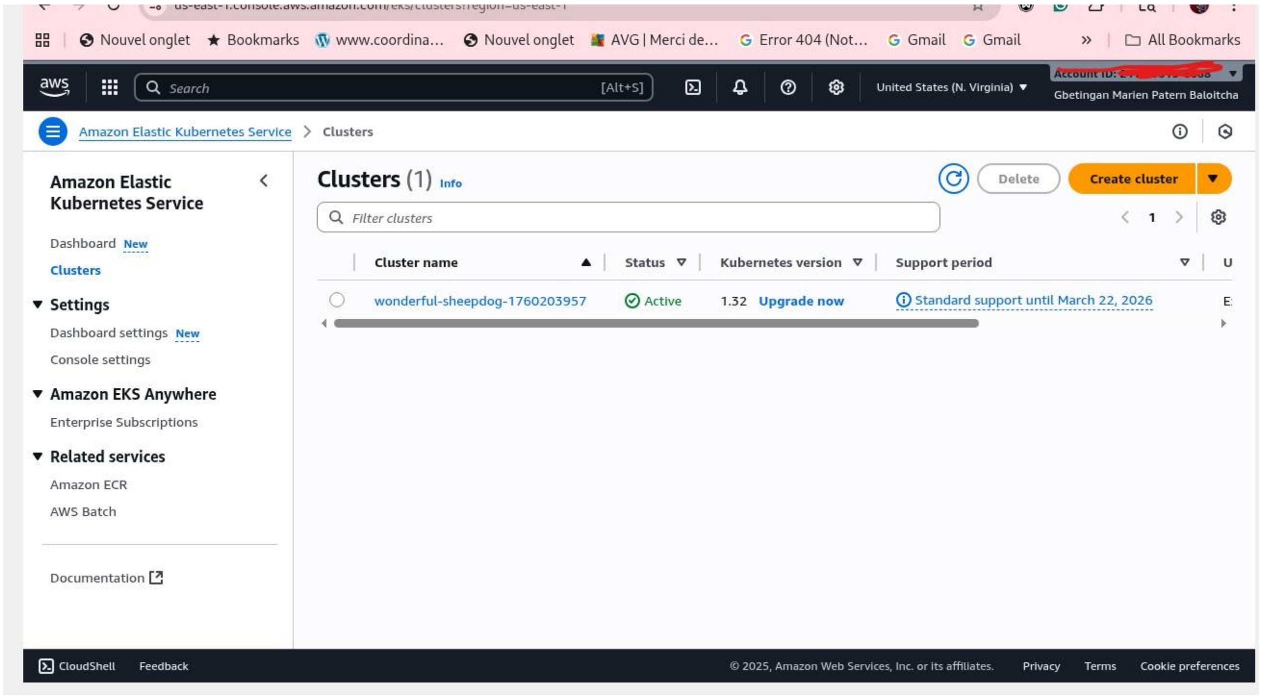
Once the cluster was active, the following Kubernetes resources were applied using kubectl:

1. **Deployment:** A deployment resource (my-web-app) was created to manage replica sets of the containerized web application.
2. **Service:** A service of type LoadBalancer (my-web-app-service) was created. This action automatically provisioned an **AWS Classic Load Balancer** to expose the application to the public internet.

C. Verification

Application health and availability were confirmed by:

- Checking the readiness status (kubectl get pods).
- Confirming the Service successfully obtained a public external IP/hostname.
- Accessing the application URL in a browser.

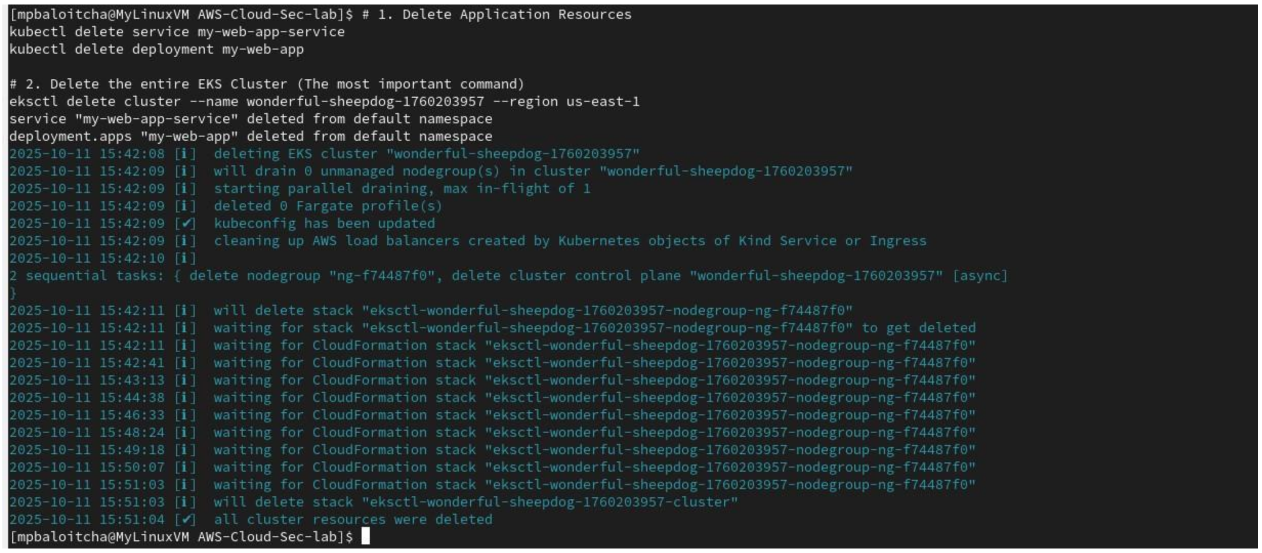


Clusters (1) Info

Cluster name	Status	Kubernetes version	Support period
wonderful-sheepdog-1760203957	Active	1.32 Upgrade now	Standard support until March 22, 2026

D. Decommissioning (Cost Management)

All cloud resources were systematically removed to prevent ongoing charges. The application was deleted first, followed by the complete infrastructure.



```
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$ # 1. Delete Application Resources
kubectl delete service my-web-app-service
kubectl delete deployment my-web-app

# 2. Delete the entire EKS Cluster (The most important command)
eksctl delete cluster --name wonderful-sheepdog-1760203957 --region us-east-1
service "my-web-app-service" deleted from default namespace
deployment.apps "my-web-app" deleted from default namespace
2025-10-11 15:42:08 [i]  deleting EKS cluster "wonderful-sheepdog-1760203957"
2025-10-11 15:42:09 [i]  will drain 0 unmanaged nodegroup(s) in cluster "wonderful-sheepdog-1760203957"
2025-10-11 15:42:09 [i]  starting parallel draining, max in-flight of 1
2025-10-11 15:42:09 [i]  deleted 0 Fargate profile(s)
2025-10-11 15:42:09 [✓]  kubeconfig has been updated
2025-10-11 15:42:09 [i]  cleaning up AWS load balancers created by Kubernetes objects of Kind Service or Ingress
2025-10-11 15:42:10 [i]
2 sequential tasks: { delete nodegroup "ng-f74487f0", delete cluster control plane "wonderful-sheepdog-1760203957" [async]
}
2025-10-11 15:42:11 [i]  will delete stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:42:11 [i]  waiting for stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0" to get deleted
2025-10-11 15:42:11 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:42:41 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:43:13 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:44:38 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:46:33 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:48:24 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:49:18 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:50:07 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:51:03 [i]  waiting for CloudFormation stack "eksctl-wonderful-sheepdog-1760203957-nodegroup-ng-f74487f0"
2025-10-11 15:51:03 [i]  will delete stack "eksctl-wonderful-sheepdog-1760203957-cluster"
2025-10-11 15:51:04 [✓]  all cluster resources were deleted
[mpbaloitcha@MyLinuxVM AWS-Cloud-Sec-lab]$
```

Cleanup Commands Executed:

1. kubectl delete service my-web-app-service
2. kubectl delete deployment my-web-app
3. eksctl delete cluster --name wonderful-sheepdog-1760203957 --region us-east-1

3. Results and Conclusion

The lab was completed successfully, achieving all technical objectives:

Phase	**Outcome**	**Status**
EKS Cluster Creation	Provisioned the EKS control plane and two worker nodes.	Complete
Application Deployment	Deployed my-web-app and exposed it via an AWS Load Balancer.	Complete
Cleanup & Decommissioning	Successfully deleted all application resources and the entire EKS cluster infrastructure.	Success

The final step, verified by the output, confirmed that **all cluster resources were deleted**, meaning the AWS environment is clean and no further billing is associated with this lab.