**Objective:** Understand the dynamics of scaling in Kubernetes on AWS.

**Tasks:**

**1.** Deploy an application on EKS.

**2.** Implement Horizontal Pod Autoscaler

3. Test and verify scaling.

**Documentation:**

- Kubernetes auto-scaling concepts.

- AWS CloudWatch metrics.

- Integrating AWS metrics with EKS.

**Prerequisites:**

* AWS Account
* AWS CLI installed and configured
* kubectl installed
* eksctl installed

**Tasks:**

**Task 1. Deploy an application on EKS:**

**a.** Create a new EKS cluster:

| eksctl create cluster --name=kubernetes-clusters-labs --version=1.27 --region=us-east-2 --node-type=t2.medium --nodes=2 --nodegroup-name=muzammil-kubernetes-lab --managed |
| --- |

**b.** Deploy a sample application:

| # simple-deployment.yaml apiVersion: apps/v1 kind: Deployment metadata:  name: simple-app spec:  replicas: 2  selector:  matchLabels:  app: simple-app  template:  metadata:  labels:  app: simple-app  spec:  containers:  - name: nginx  image: nginx  ports:  - containerPort: 80 |
| --- |

| kubectl apply -f simple-deployment.yaml |
| --- |

Create Yaml file testing-hpa.yaml

| kind: Deployment apiVersion: apps/v1 metadata:  name: testing-hpa spec:  replicas: 1  selector:  matchLabels:  name: deployment  template:  metadata:  name: testpod8  labels:  name: deployment  spec:  containers:  - name: app-nginx  image: nginx  ports:  - containerPort: 80  resources:  limits:  cpu: 500m  requests:  cpu: 200m |
| --- |

| kubectl apply -f testing-hpa |
| --- |

**Task 2. Implement Horizontal Pod Autoscaler (HPA):**

**a.** Install metrics server (required for HPA):

| kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml |
| --- |

**b.** Create the HPA:

| # hpa.yaml apiVersion: autoscaling/v2 kind: HorizontalPodAutoscaler metadata:  name: simple-app-hpa spec:  scaleTargetRef:  apiVersion: apps/v1  kind: Deployment  name: simple-app  minReplicas: 1  maxReplicas: 5  metrics:  - type: Resource  resource:  name: cpu  target:  type: Utilization  averageUtilization: 20 |
| --- |

Apply the HPA configuration:

| kubectl apply -f hpa.yaml |
| --- |

**Task 3. Test and verify scaling:**

**a. Generate CPU load:**

This can be done by creating a busy box pod and running a **while** loop:

| kubectl run -i --tty load-generator --image=busybox /bin/sh while true; do wget -q -O- http://simple-app; done |
| --- |

1.2. Once inside the pod's shell, generate CPU load using an infinite loop:

| while true; do wget -q -O- http://simple-app.default.svc.cluster.local; done |
| --- |

**b.** Monitor the HPA status:

| **kubectl autoscale deployment testing-hpa --cpu-percent=20 --min=1 --max=10** kubectl get hpa simple-app-hpa -w |
| --- |

**Documentation:**

* **Kubernetes auto-scaling concepts:**
* Autoscaling in Kubernetes is facilitated by the HPA, which automatically scales the number of pods in a deployment or replica set based on observed metrics like CPU utilization or, with custom metrics support, other selected metrics.
* **AWS CloudWatch metrics:**
* AWS CloudWatch provides data and actionable insights to monitor applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health.
* **Integrating AWS metrics with EKS:**
* To integrate AWS metrics, you can install the k8s-cloudwatch-adapter. This enables custom metrics from AWS CloudWatch, allowing you to scale based on any CloudWatch metric.

**Troubleshooting tips and tricks:**

1. **Issue:** EKS cluster creation fails.

**Solution:** Ensure AWS CLI is properly configured with appropriate permissions.

1. **Issue:** Metrics not available in HPA.

**Solution:** Check the metrics server logs for issues. Sometimes it might take a few minutes for metrics to be collected.

1. **Issue:** HPA doesn't scale up despite load.

**Solution**: Ensure the load generated is significant enough to cross the threshold set in the HPA configuration.

1. **Issue:** CloudWatch metrics not showing in EKS.

**Solution:** Ensure k8s-cloudwatch-adapter is correctly installed and has appropriate IAM permissions.

**Lab Performance:**

Students will first deploy an application, implement autoscaling, and then generate load to test the scaling. Ensure that they monitor the HPA status to see if the scaling is happening as expected.