**\*\*Lab 7: Implementing Service Mesh with AWS App Mesh and EKS\*\***

**\*\*Objective\*\*: Dive into service mesh capabilities in EKS.**

**\*\*Tasks\*\*:**

**1. Set up AWS App Mesh.**

**2. Integrate with EKS.**

**3. Observe and optimize service-to-service communication.**

**Prerequisites:**

**Install kubectl**

**Install eks**

**Install awscli**

**Add access key**

**Add roles for cluster and node permissions**

**Step 1: Set Up AWS App Mesh**

**Define Virtual Services and Virtual Nodes:  
Create virtual services for your microservices and virtual nodes for individual pods/tasks. Define routes to route traffic between them using App Mesh configuration. Here's an example of defining a virtual service:**

| apiVersion: appmesh.k8s.aws/v1beta2 kind: VirtualService metadata:  name: my-virtual-service  namespace: my-namespace spec:  meshName: my-app-mesh  virtualRouter:  name: my-virtual-router |
| --- |

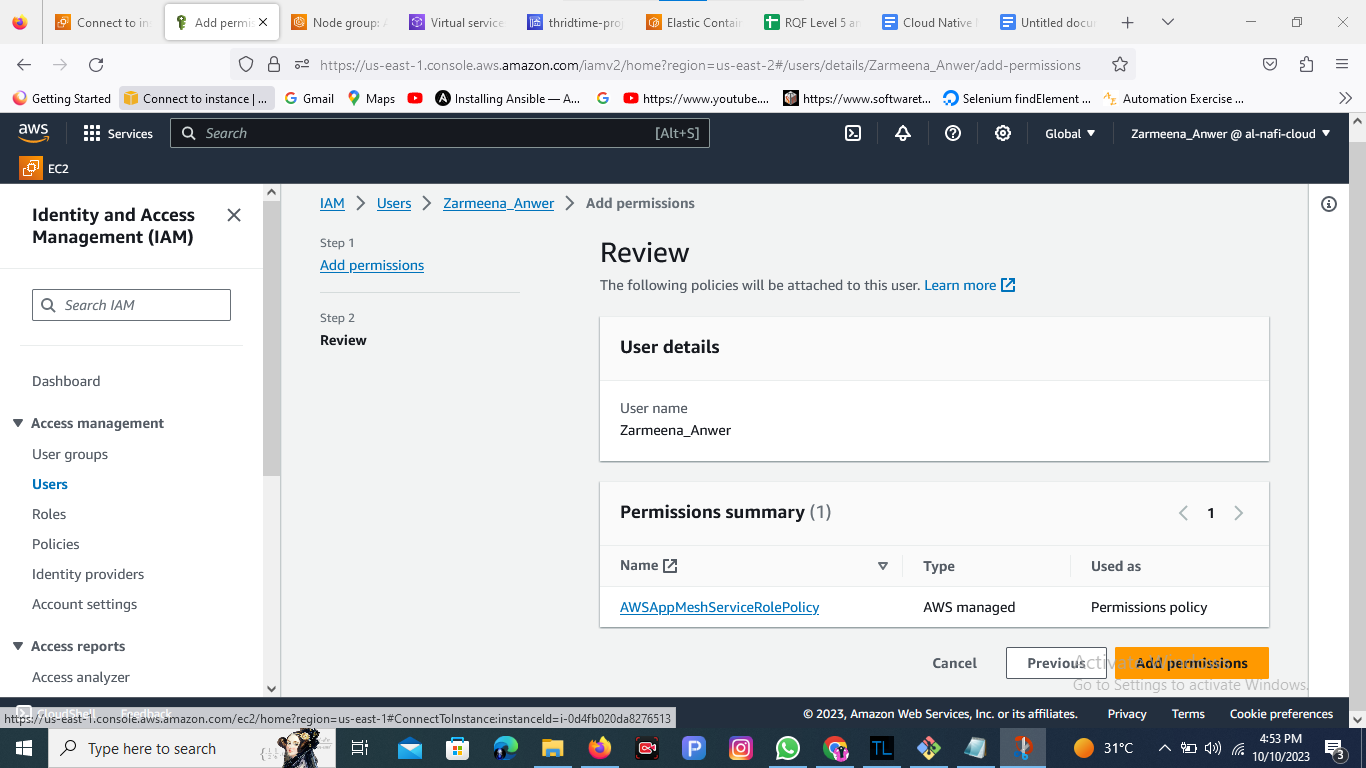
| apiVersion: apps/v1 kind: Deployment metadata:  name: my-microservice-deployment  labels:  app: my-microservice spec:  replicas: 3  selector:  matchLabels:  app: my-microservice  template:  metadata:  labels:  app: my-microservice  annotations:  appmesh.k8s.aws/sidecarInjectorWebhook: enabled # Enable sidecar injection by App Mesh controller  spec:  containers:  - name: my-microservice-container  image: my-microservice-image:latest  ports:  - containerPort: 80 |
| --- |

* Install AWS App Mesh Controller and CRDs:
  + If you haven't already, install the AWS App Mesh controller and the Custom Resource Definitions (CRDs) required for AWS App Mesh. You can use the provided Helm chart to do this
  + helm repo add ekshttps://aws.github.io/eks-charts helm install appmesh-controller eks/appmesh-controller --namespace appmesh-system -- region=<AWS\_REGION>  
     Make sure to replace <AWS\_REGION> with your AWS region.
* Verify CRDs Installation:  
  After installing the controller and CRDs, verify that the CRDs have been successfully installed in your cluster. You can check the list of CRDs by running:

| kubectl get crd | grep appmesh |
| --- |

| kubectl get pods -n appmesh-system |
| --- |

Add permission for this lab.



**Conclusion of the lab**

**The implementation of a service mesh architecture significantly contributes to enhancing the security posture of microservices-based applications.**

* **Robust Security Measures: Service mesh, through its capabilities such as mTLS encryption, access control, and policy enforcement, strengthens security by providing a secure communication layer among microservices.**
* **Operational Insights and Control: The observability and control features of service mesh ensure visibility into microservices interactions and enable centralized policy enforcement, facilitating proactive security measures and incident response.**
* **Scalable and Resilient Infrastructure: Service mesh architecture supports scalable and resilient microservices environments by providing dynamic routing, fault isolation, and service discovery while maintaining security.**

**Adopting a service mesh architecture represents a significant step towards ensuring a secure and resilient microservices ecosystem. Continuous monitoring, adaptation of security policies, and adherence to best practices within the service mesh framework are essential for maintaining a robust security posture in microservices-based applications.**

**===========================**