**Objective:** Understand the process and advantages of having EKS clusters across multiple AWS regions.

**Tasks:**

1. Set up EKS clusters in two different AWS regions.

2. Configure inter-cluster communication.

3. Deploy an application spanning both clusters.

**Requirements:**

1. An AWS account with required permissions to create and manage EKS clusters, EC2 instances, and EBS volumes.
2. AWS CLI installed and configured.
3. **kubectl** installed.
4. **eksctl** installed.
5. IAM Cluster Permission

**Documentation:**

- Importance of multi-region deployments.

- EKS inter-cluster communication.

- Strategies for multi-region scaling.

**Importance of Multi-Region Deployments:**

Multi-region deployments provide several advantages:

1. **High Availability:** If one region goes down due to any issue, your application will still be available in another region.
2. **Latency Reduction:** Serving users from the nearest region can improve the application's response time.
3. **Regulatory Compliance:** Some industries must ensure their data resides in specific geographic locations.
4. **Disaster Recovery:** If data gets lost or corrupted in one region, backups from another region can be utilized.

**Task 1: Deploying Multi-Region EKS Clusters**

**Task 1:** Set up EKS clusters in two different AWS regions

Pre-requisites:

1. **Kubectl**
2. **AWS CLI configured**
3. **eksctl installed**
4. **Docker**

| eksctl create cluster \  --name=eks-1 \  --version=1.27 \  --region=us-east-2 \  --spot \  --node-type=t2.medium \  --nodes=1 \  --nodes-min=1 \  --nodes-max=2 \  --nodegroup-name=eks-node-grp-2 \  --managed |
| --- |

Or Second Cluster Created in Different Region:

| eksctl create cluster \  --name=eks-2 \  --version=1.27 \  --region=us-west-1 \  --spot \  --node-type=t2.medium \  --nodes=1 \  --nodes-min=1 \  --nodes-max=2 \  --nodegroup-name=eks-node-grp-2 \  --managed \  --vpc-cidr=10.0.0.0/24 |
| --- |

**Task 2: Configure inter-cluster communication**

**Step 1: Navigate to the VPC Dashboard**

1. Log in to your AWS Management Console.
2. In the "Find Services" box, type "VPC" and select the "VPC" service.

**Step 2: Initiate VPC Peering**

1. In the VPC Dashboard's left-hand pane, click on "Peering Connections."
2. Click the "Create Peering Connection" button.
3. For the "Name tag," give it a descriptive name, e.g., "EKS-inter-cluster-peering."
4. For "VPC (Requester)", select the VPC of your EKS cluster in us-east-2.
5. For "Account", select "My Account" (assuming both EKS clusters are in the same AWS account).
6. For "Region", select "Another region" and then select us-west-1.
7. For "VPC (Accepter)", select the VPC of your EKS cluster in us-west-1.
8. Click the "Create Peering Connection" button.

**Step 3: Accept the Peering Request**

1. Now, you need to switch to the us-west-1 region using the region selector at the top-right corner of the AWS Management Console.
2. Go back to the "Peering Connections" under the VPC Dashboard.
3. You'll see the peering connection you created. Its status will be "Pending Acceptance."
4. Select the peering connection and click the "Actions" button, then choose "Accept Request."
5. In the pop-up dialog, click the "Yes, Accept" button.

**Task 3: Deploy an application spanning both clusters**

For demonstration, let's deploy a simple **Nginx deployment:**

| # On us-east-2 cluster kubectl create deployment nginx --image=nginx  # On eu-west-1 cluster kubectl create deployment nginx --image=nginx |
| --- |

Verify

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

| curl paster-nginx-load-balancer |
| --- |

**Documentation:**

**EKS Inter-Cluster Communication:**

1. AWS PrivateLink provides private connectivity between VPCs, AWS services, and on-premises applications.
2. EKS clusters in different regions can communicate over AWS's network backbone without being exposed to the public internet.

**Strategies for Multi-Region Scaling:**

1. Active-Active: All regions are active and share the traffic load. If one region fails, the other region handles all traffic.
2. Active-Passive: One region handles the traffic, while the other region is a standby. In case of failure, traffic is routed to the standby region.
3. Data Replication: Ensure data is replicated across regions to provide fast access to data.
4. Load Balancing: Use AWS Global Accelerator or Route 53 to distribute incoming traffic across multiple regions based on health, geographic conditions, and other factors.