AWS - EKS

* EKS stands for "Elastic Kubernetes Service"
* EKS is a fully managed AWS service
* EKS is the best place to run K8S applications because of its security, reliability and scalability
* EKS can be integrated with other AWS services such as ELB, CloudWatch, AutoScaling, IAM and VPC
* EKS makes it easy to run K8S on AWS without needing to install, operate and maintain your own k8s control plane.
* Amazon EKS runs the 'K8S control Plane' across three availability zones in order to ensure high availability and it automatically detects and replaces unhealthy masters.
* AWS will have complete control over Control Plane. We don't have control on Control Plane.
* We need to create Worker Nodes and attach to Control Plane.
* Note: We will create Worker Nodes Group using ASG Group
* Control Plane Charges + Worker Node Charges (Based on Instance Type & No.of Instances)

**Pre-Requisites**

1. AWS account with admin privileges.
2. Instance to manage/access EKS cluster using Kubectl (K8S-Client-VM)
3. AWS CLI access to use kubectl utility

**Steps to Create EKS Cluster in AWS**

**Step-1) Create VPC using Cloud Formation ( with below S3 URL )**

URL : https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml

Stack name : EKSVPCCloudFormation

**Step-2) Create IAM role in AWS**

Entity Type : AWS Service

Select Usecase as 'EKS' = EKS Cluster

Role Name : EKSClusterRole (you can give any name for the role)

**Step-3) Create EKS Cluster using Created VPC and created IAM Role**

Cluster endpoint access : Public & Private

**Step-4) Create RedHat ec2 Instance (K8S\_Client\_Machine)**

Connect to K8S\_Client\_Machine using Mobaxterm

**######## Install Kubectl with below commands ###############**

$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

$ sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

$ kubectl version --client

**########## Install AWS CLI in K8S\_Client\_Machine with below commands ##############**

$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

$ sudo yum install unzip

$ unzip awscliv2.zip

$ sudo ./aws/install

**############ Configure AWS Cli with Credentials ###############**

We can create AWS Access Keys using IAM service

Access Key ID: AKIAYG3UVCXWDIVZ6GMD

Secret Access Key: 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

$ aws configure

Note: We can use root user accesskey and secret key access

**##########################################################**

$ aws eks list-clusters

$ ls ~/.

**######## Update kubeconfig file in remote machine from cluster using below command ######**

$ aws eks update-kubeconfig --name <cluster-name> --region <region-code>

Ex: aws eks update-kubeconfig --name EKSCluster --region ap-south-1

**########## ###################################################################**

**Step-5 ) Create IAM role for EKS worker nodes (usecase as EC2) with below policies**

Entity Type : AWS Service

a) AmazonEKSWorkerNodePolicy

b) AmazonEKS\_CNI\_Policy

c) AmazonEC2ContainerRegistryReadOnly

**Step-6) Create Worker Node Group ( Go to cluster Compute Node Group )**

Select the Role we have created for WorkerNodes

Use t2.medium

Min 2 and Max 5

**Step-7) Once Node Group added then check nodes in K8s\_client\_machine**

$ kubectl get nodes

$ kubectl get pods --all-namespaces

**Step-8) Create POD and Expose the POD using NodePort service**

**Note: Enable NODE PORT in security Group to access that in our browser**

**Finally**

1) Delete Worker Node Group

2) Delete EKS Cluster

3) Delete VPC + NAT Gateway + Elastic IP