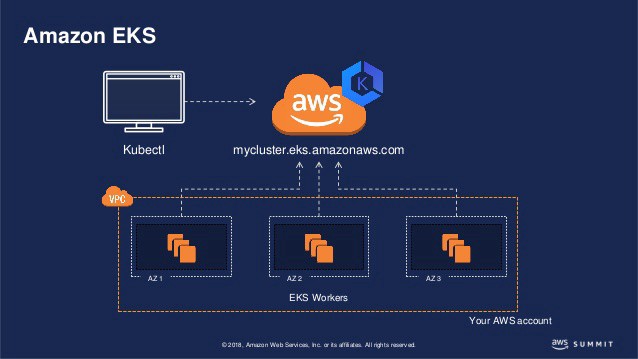
**Deploying a Kubernetes Cluster with Amazon EKS**



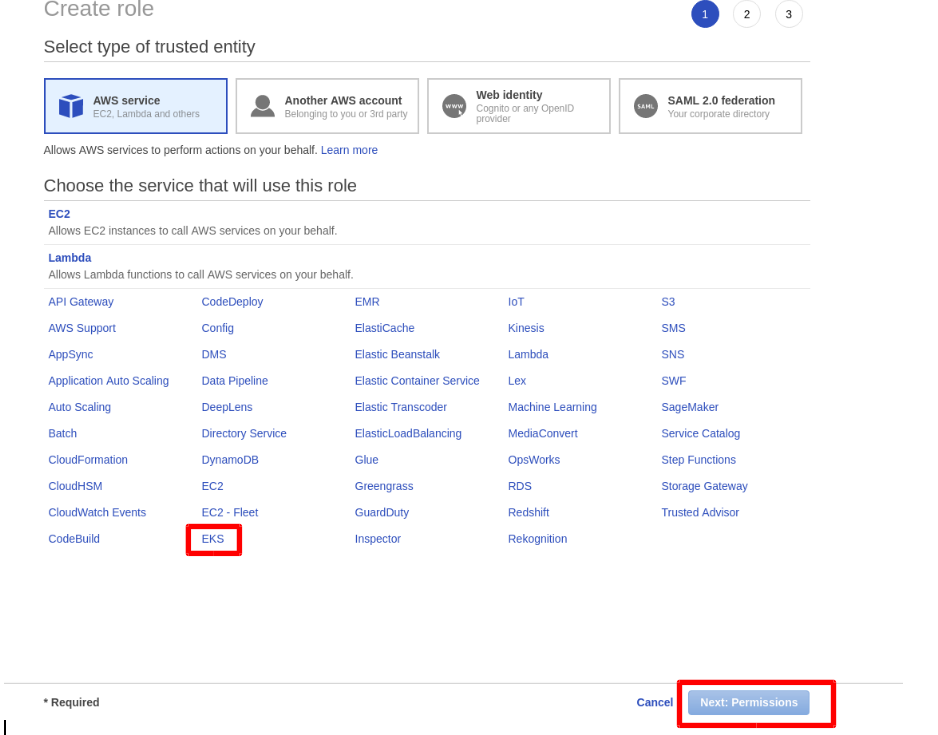
EKS

Kubernetes is one of the widely used Container managed technology, We make k8s cluster which makes it easy to deploy, manage, and scale containerized applications using Kubernetes on AWS Cloud.

It is an OpenSource production-ready Container Orchestration tool that is used for automating deployments, Scaling, and managing the containerized application. Amazon recently on 5th June 2018 launched an [EKS](https://aws.amazon.com/eks/) — Elastic container Service for Kubernetes (Managed Kubernetes) In this blog, We will see Steps to Configure the Kubernetes cluster using AWS EKS.

***Step 1: Create an AWS EKS Role.***

The First Step is to create an EKS role that Kubernetes can assume to provide the required resources. In order to form the EKS Role, login to the AWS Management Console, Click on Role and so Click produce Role. Select EKS under AWS Services and click on Next: Permission.

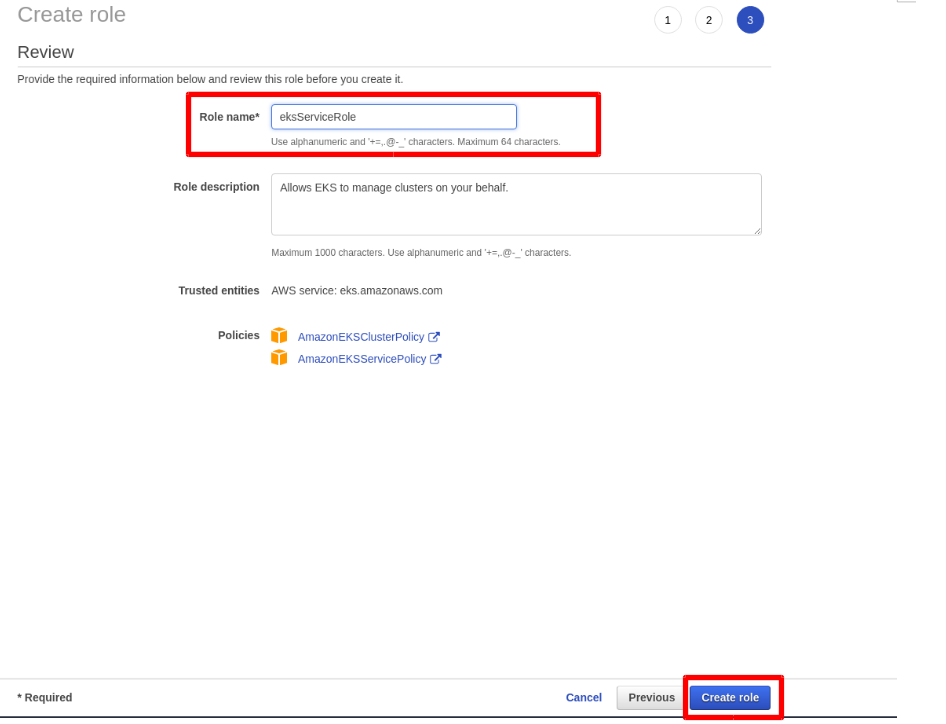


By Default, it will Select 2 Custom Policy for Role Which is,

* ***AmazonEKSClusterPolicy***
* ***AmazonEKSServicePolicy***

***Click Next: Review***

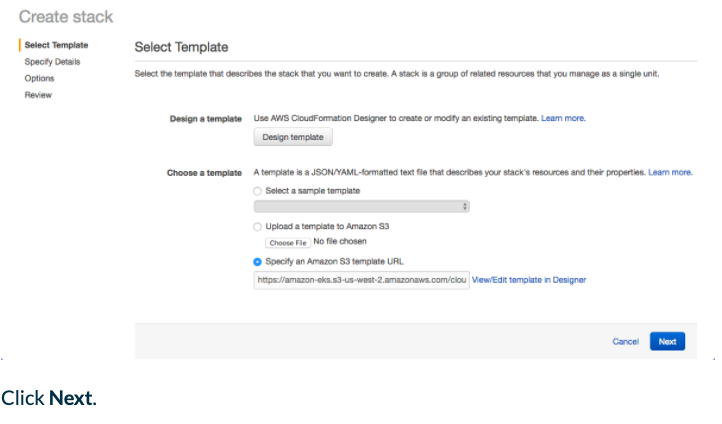
Provide Role Name and ***Click Create Role***.



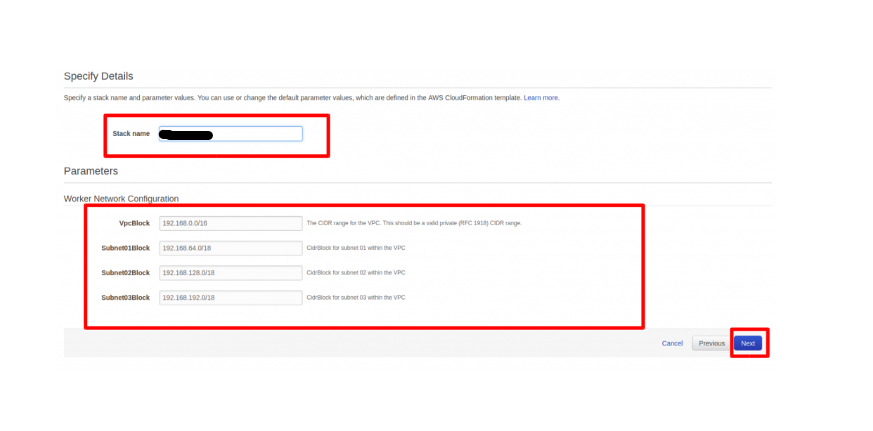
***Step 2: Create VPC for EKS Cluster.***

As we have completed the Role Creation, currently we’ll be making VPC for Cluster resources. We will use ***Cloudformation Script*** to create a VPC Stack. Now Move to the CloudFormation Section and Click on ***Create Stack***. Click on Specify an Amazon S3 template URL and paste the below Link (Cloudformation Template) and ***Click Next***.

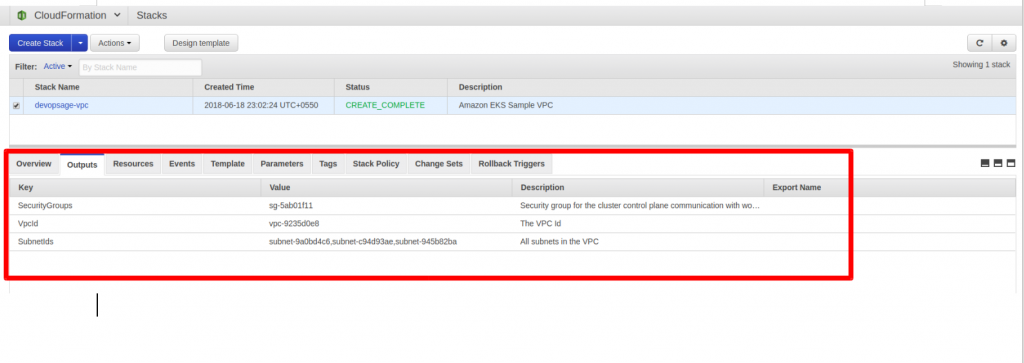
<https://amazon-eks.s3-us-west-2.amazonaws.com/1.10.3/2018-06-05/amazon-eks-vpc-sample.yaml>



Provide the Stack Name and fill out the required parameters for VPC Stack and **Click Next**. Review the details and Click on Create. Cloud formation Stack for VPC will Initiate and will be created in a few mins.



“Make a Note of the Resource Output once the Stack is formed, This we’ll use any for Cluster configuration”.



***Step 3: Install and configure kubectl for EKS Cluster***

The EKS Cluster always requires Kubelet and Kubectl and heptio to allow IAM permissions for the Kubernetes cluster. You can install either the Amazon EKS-amended version of **kubectl**or directly from the Kubernetes documentation.

Note: Here We will launch a t2.micro instance and will use it as a Kubernetes client. Launch the instance (Amazon AMI Preferred) and configure [AWS CLI](http://www.devopsage.com/how-to-setup-aws-command-line-interface-cli/) on it.

We must use API credentials of a constant user with whom you will launch the EKS cluster else it won't be succeeded due to authentication and won't be able to connect to Master node.

Log in to the instance launched and execute the below commands. (Open SSH port in Security Group).

Use the below commands to install the kubectl

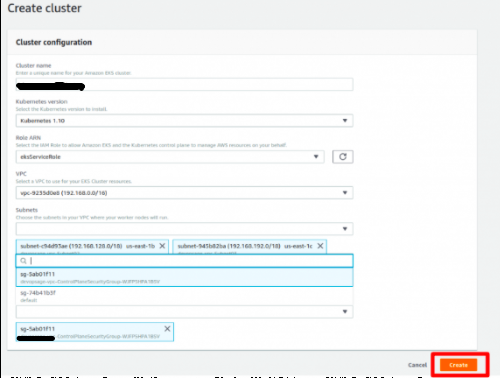
***(OPTIONAL) You can also install the kubectl from the***[***Kubernetes documentation.***](https://kubernetes.io/docs/tasks/tools/install-kubectl/#install-kubectl)***or refer to the below commands.***

***Step 4: Install heptio-authenticator-aws for Amazon EKS***

Use the below set of command to install heptio-authenticator-aws

***Step 5: Install AWS EKS Cluster from AWS Console***

Select EKS service in the AWS Console. Provide the below details in the screenshot and click Create.



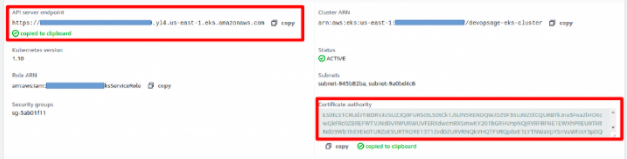
***Note: It will take around 5 mins to reach a cluster in the Active State.***

***Step 6: Configure Kubectl for EKS Cluster***

Follow along with the below steps to configure the kubectl for EKS Cluster,

Make Sure to replace the below parameters in the above config files,

* ***server: <endpoint-url>***
* ***certificate-authority-data: “<base64-encoded-ca-cert>”***
* ***“<cluster-name>”***



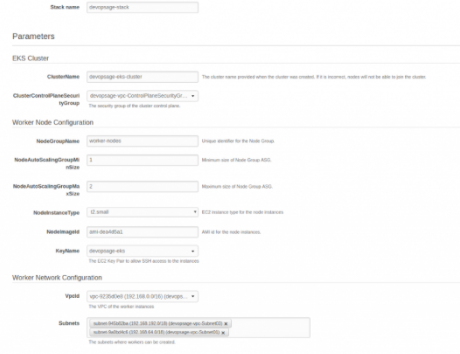
So it Shows that the configuration is properly done and we can proceed.

***Step 7: Launch and Configure EKS Worker Nodes***

Currently, we are going to add client nodes and can attach it to the Cluster wherever we are able to deploy an application. We will be using the cloud formation stack to launch the Worker nodes. It launches the nodes in the AutoScaling group and will be charged accordingly.

Go to CloudFormation and Click on Create Stack, Click on Specify an Amazon S3 template URL and paste below URL there,

<https://amazon-eks.s3-us-west-2.amazonaws.com/1.10.3/2018-06-05/amazon-eks-nodegroup.yaml>



Fill Out all the required parameters and click on Create.

Note:

* In CF Parameter Cluster Name Should match exactly with the EKS Cluster name or else, the worker node will not be able to join the cluster.
* Use the VPC which we have created above.
* Amazon Provides Custom AMI for the worker node, which is usually an Amazon Linux 2 AMI

Use AMI: N Virginia : ami-Bea4d4e1 Oregon: ami-73asbsdbj30b

***Step 8: Join Worker Nodes to the Cluster.***

**Note**: Your Client Server Terminal might hang due to the change in the outbound traffic made by the cloud formation Stack. go and check the Security group outbound rule of the client node and allow all traffic from everywhere.

So we have Successfully configured the EKS Cluster and registered the worker nodes to it, Quickly verify it by using below commands,

***Step 9: Deploy Sample Application on the Cluster***

As we are done with the setup of the EKS Cluster, we will deploy the famous Guestbook Application on the cluster. The guestbook is a public application available, We will be deploying it just for testing, use below commands,

Execute below commands to check pods and services.

Now hit the ELB endpoint along with the port on URL to check whether the Application is successfully deployed or not.

[http://ad4674e91732d11e8b16c12bd24666d9-1968070884.us-east-1.elb.amazonaws.com:3000](http://ad4674e91732d11e8b16c12bd24666d9-1968070884.us-east-1.elb.amazonaws.com:3000/) // Will be different in your case, get your endpoint from "kubectl get svc"