Deployment on ALB by ingress LB with ALB internal LB

Document Followed with some required changes: ALB should be present in private subnet

<https://aws.amazon.com/blogs/containers/using-alb-ingress-controller-with-amazon-eks-on-fargate/>

## step 1: Prerequisites

In order to successfully execute these steps, follow the steps in the [EKS getting started guide](https://docs.aws.amazon.com/eks/latest/userguide/getting-started-eksctl.html) (don’t create a cluster) and make sure you have the following components installed:

* The EKS CLI, [eksctl](https://eksctl.io/introduction/installation/), for example, on macOS with brew tap weaveworks/tap and brew install weaveworks/tap/eksctl
* The latest version of the [AWS CLI](https://docs.aws.amazon.com/cli/latest/userguide/install-cliv1.html).
* The Kubernetes CLI, [kubectl](https://kubernetes.io/docs/tasks/tools/install-kubectl/) .  
  Note, if you used the Homebrew installation to install eksctl on macOS, then kubectl has already been installed on your system
* [jq](https://stedolan.github.io/jq/download/)

Now that everything is properly installed in your environment, we can go ahead and start building.

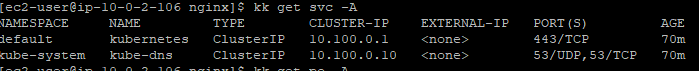
## Step2: Cluster provisioning

AWS\_REGION=<aws\_region>

CLUSTER\_NAME=eks-fargate-alb-demo

eksctl create cluster --name $CLUSTER\_NAME --region $AWS\_REGION –fargate

kubectl get svc



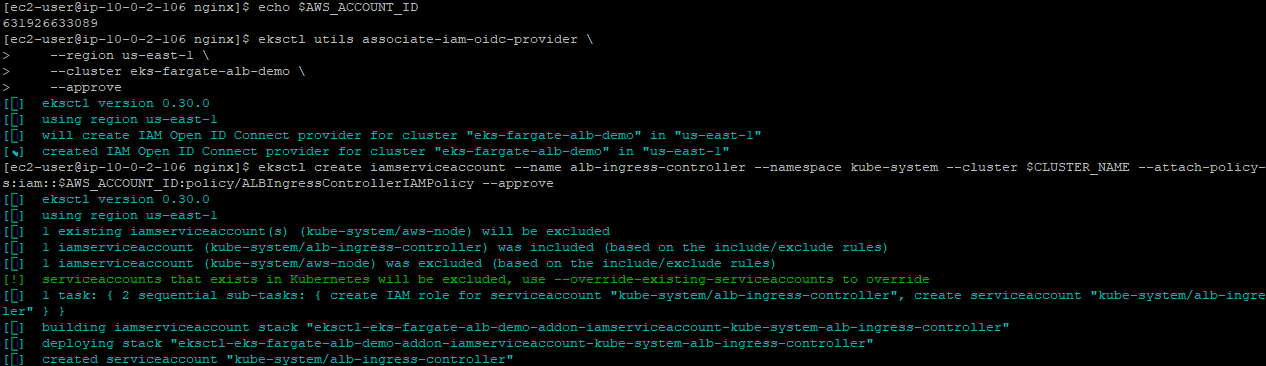
1. OIDC setup

eksctl utils associate-iam-oidc-provider --cluster $CLUSTER\_NAME –approve

1. Download the IAM Policy example document and create it

wget -O alb-ingress-iam-policy.json https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-ingress-controller/master/docs/examples/iam-policy.json

aws iam create-policy --policy-name ALBIngressControllerIAMPolicy --policy-document <file://alb-ingress-iam-policy.json>



## Step3: Create a cluster role, role binding, and a Kubernetes service account

STACK\_NAME=eksctl-$CLUSTER\_NAME-cluster

VPC\_ID=$(aws cloudformation describe-stacks --stack-name "$STACK\_NAME" | jq -r '[.Stacks[0].Outputs[] | {key: .OutputKey, value: .OutputValue}] | from\_entries' | jq -r '.VPC')

## AWS\_ACCOUNT\_ID=$(aws sts get-caller-identity | jq -r '.Account')

## Now, create the Cluster Role and Role Binding:

cat > rbac-role.yaml <<-EOF

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

labels:

app.kubernetes.io/name: alb-ingress-controller

name: alb-ingress-controller

rules:

- apiGroups:

- ""

- extensions

resources:

- configmaps

- endpoints

- events

- ingresses

- ingresses/status

- services

verbs:

- create

- get

- list

- update

- watch

- patch

- apiGroups:

- ""

- extensions

resources:

- nodes

- pods

- secrets

- services

- namespaces

verbs:

- get

- list

- watch

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

labels:

app.kubernetes.io/name: alb-ingress-controller

name: alb-ingress-controller

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: alb-ingress-controller

subjects:

- kind: ServiceAccount

name: alb-ingress-controller

namespace: kube-system

EOF

## kubectl apply -f rbac-role.yaml

O/P: clusterrole.rbac.authorization.k8s.io/alb-ingress-controller created

## clusterrolebinding.rbac.authorization.k8s.io/alb-ingress-controller created

## 

## cat > nginx-deployment.yaml <<-EOF

## apiVersion: apps/v1

## kind: Deployment

## metadata:

## name: "nginx-deployment"

## namespace: "default"

## spec:

## replicas: 3

## selector:

## matchLabels:

## app: "nginx"

## template:

## metadata:

## labels:

## app: "nginx"

## spec:

## containers:

## - image: nginx:latest

## imagePullPolicy: Always

## name: "nginx"

## ports:

## - containerPort: 80

## EOF

## kubectl apply -f nginx-deployment.yaml

## Then, let’s create a service so we can expose the NGINX pod

cat > nginx-ingress.yaml <<-EOF

apiVersion: extensions/v1beta1

kind: Ingress

metadata:

name: "nginx-ingress"

namespace: "default"

annotations:

kubernetes.io/ingress.class: alb

alb.ingress.kubernetes.io/scheme: internet-facing

labels:

app: nginx-ingress

spec:

rules:

- http:

paths:

- path: /\*

backend:

serviceName: "nginx-service"

servicePort: 80

## EOF

## kubectl apply -f nginx-ingress.yaml

## The output will be:

## ingress.extensions/nginx-ingress created

## Step4:

## Once everything is done, you will be able to get the ALB URL by running the following command:

## 

Validation:

