EKS assignment

1. We need to create and run our EKS cluster using the command:

eksctl create cluster --name mycluster02

```
Saidas-MBP:~ dirisova$ eksctl create cluster --name mycluster02
2021-10-29 20:43:02 [ℹ]  eksctl version 0.70.0
2021-10-29 20:43:02 [ℹ]  using region us-east-1
2021-10-29 20:43:02 [ℹ]  setting availability zones to [us-east-1a us-east-1d]
2021-10-29 20:43:02 [ℹ]  subnets for us-east-1a - public:192.168.0.0/19 private:192.168.64.0/19
2021-10-29 20:43:02 [ℹ]  subnets for us-east-1d - public:192.168.32.0/19 private:192.168.96.0/19
2021-10-29 20:43:02 [ℹ]  nodegroup "ng-fef8e854" will use "" [AmazonLinux2/1.20]
```

Checking if it's working: eksctl get cluster

```
Saidas-MBP:~ dirisova$ eksctl get cluster
2021-10-29 21:21:14 [ℹ]  eksctl version 0.70.0
2021-10-29 21:21:14 [ℹ]  using region us-east-1
NAME            REGION          EKSCTL CREATED
mycluster02     us-east-1       True
Saidas-MBP:~ dirisova$
```

2. After creating EKS cluster I'm going to create a Ngnix deployment yaml file with a service. I made a directory to store the yaml files for this particular project:

- mkdir Dep8-EKS
- cd Dep8-EKS
- Then create 2 yaml file, Ngnix deployment yaml file with a service and another yaml file for the ingress controller

```
Saidas-MBP:Dep8-EKS dirisova$ pwd
/Users/dirisova/Dep8-EKS
Saidas-MBP:Dep8-EKS dirisova$ ls
ingress.yaml    ngnix.yaml
Saidas-MBP:Dep8-EKS dirisova$
```

3. I need to add OpenID connect to my cluster:

-aws eks describe-cluster --name mycluster02 --query "cluster.identity.oidc.issuer" --output text

```
Saidas-MBP:~ dirisova$  aws eks describe-cluster --name mycluster02 --query "cluster.identity.oidc.issuer" --output tex
t
https://oidc.eks.us-east-1.amazonaws.com/id/163D3B43BF03E91251CDB222ECF2918C
Saidas-MBP:~ dirisova$
```

4. adding OpenID to the cluster:

```
Saidas-MBP:~ dirisova$ eksctl utils associate-iam-oidc-provider --cluster mycluster02 --approve
2021-10-29 21:55:03 [ℹ]  eksctl version 0.70.0
2021-10-29 21:55:03 [ℹ]  using region us-east-1
2021-10-29 21:55:03 [ℹ]  will create IAM Open ID Connect provider for cluster "mycluster02" in "us-east-1"
2021-10-29 21:55:03 [✔]  created IAM Open ID Connect provider for cluster "mycluster02" in "us-east-1"
Saidas-MBP:~ dirisova$
```

5. Running the previous command again to see if an open ID provider list was added.

```
Saidas-MBP:~ dirisova$  aws eks describe-cluster --name mycluster02 --query "cluster.identity.oidc.issuer" --output tex
t
https://oidc.eks.us-east-1.amazonaws.com/id/163D3B43BF03E91251CDB222ECF2918C
Saidas-MBP:~ dirisova$ aws iam list-open-id-connect-providers
```

6. Download the Role Base Access Control:

```
Saidas-MBP:~ dirisova$ curl -o rbac-role.yaml https://raw.githubusercontent.com/RobinNagpal/kubernetes-tutorials/master/06
_tools/007_alb_ingress/01_eks/rbac-role.yaml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1163  100  1163    0     0   5459      0 --:--:-- --:--:-- --:--:--  5618
Saidas-MBP:~ dirisova$
```

That going to download yaml file (rbac-role.yaml) to the current directory.

7. So next I'm going to do is apply access yaml file by doing
- kubectl apply -f rbac-role.yaml

```
Saidas-MBP:~ dirisova$ kubectl apply -f rbac-role.yaml
clusterrole.rbac.authorization.k8s.io/alb-ingress-controller created
clusterrolebinding.rbac.authorization.k8s.io/alb-ingress-controller created
serviceaccount/alb-ingress-controller created
```

```
clusterrolebinding.rbac.authorization.k8s.io/alb-ingress-controller created
serviceaccount/alb-ingress-controller created
Saidas-MBP:~ dirisova$ kubectl get serviceaccount
NAME      SECRETS   AGE
default   1         99m
Saidas-MBP:~ dirisova$ kubectl get clusterrolebinding.rbac.authorization.k8s.io
NAME                                                ROLE
AGE
alb-ingress-controller                              ClusterRole/alb-ingress-controller
2m9s
aws-node                                            ClusterRole/aws-node
100m
cluster-admin                                       ClusterRole/cluster-admin
100m
eks:addon-manager                                   ClusterRole/eks:addon-manager
```

That shows me the resources cluster role, cluster binding etc.

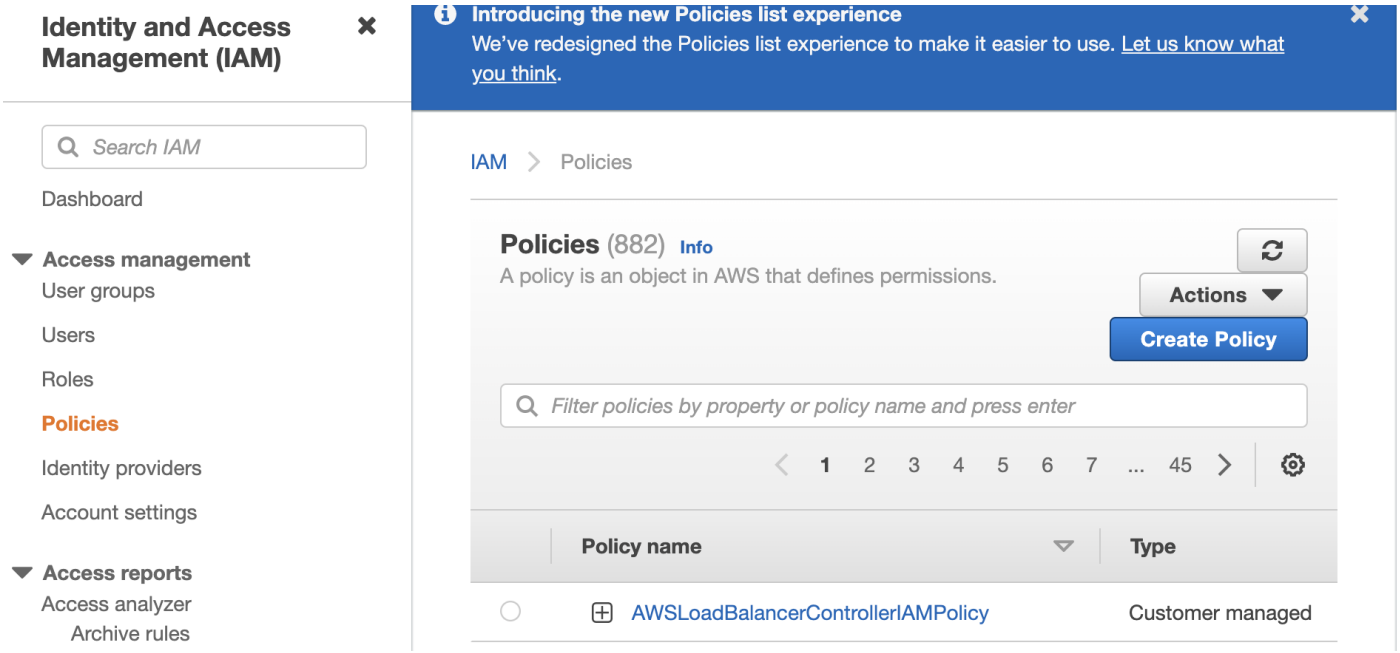
8. Next, I'm going to create policy on aws, so i need to download iam policy

```
Saidas-MBP:~ dirisova$ curl -o iam_policy.json https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-control
ler/v2.3.0/docs/install/iam_policy.json
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7585  100  7585    0     0  30736      0 --:--:-- --:--:-- --:--:-- 32004
Saidas-MBP:~ dirisova$
```

That going to download iam_policy.json file to the current directory.

9. Next you will create the AWS policy with the following command:

```
-aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```



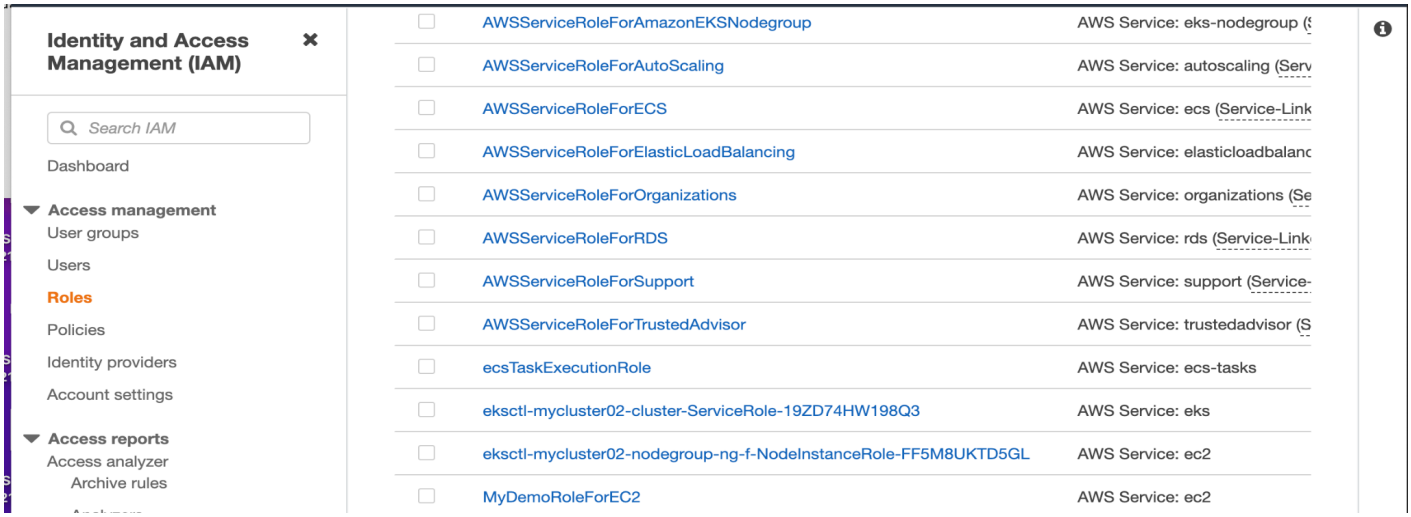
To see the policy which we just created login to the aws account aws-IAM-Policies-AWSLoadBalancerControllerIAMPolicy

10. Now I'm going to create a service account for my cluster.

- eksctl create iamserviceaccount --cluster=mycluster02 --namespace=kube-system --name=aws-load-balancer-controller --attach-policy-arn=arn:aws:iam::AWS ID number:policy/AWSLoadBalancerControllerIAMPolicy --override-existing-serviceaccounts --approve

```
2021-10-29 23:17:54 [ℹ]  eksctl version 0.70.0
2021-10-29 23:17:54 [ℹ]  using region us-east-1
2021-10-29 23:17:56 [ℹ]  1 iamserviceaccount (kube-system/aws-load-balancer-controller) was included (based on the include
/exclude rules)
2021-10-29 23:17:56 [!]  metadata of serviceaccounts that exist in Kubernetes will be updated, as --override-existing-serv
iceaccounts was set
2021-10-29 23:17:56 [ℹ]  1 task: {
    2 sequential sub-tasks: {
        create IAM role for serviceaccount "kube-system/aws-load-balancer-controller",
        create serviceaccount "kube-system/aws-load-balancer-controller",
    } }2021-10-29 23:17:56 [ℹ]  building iamserviceaccount stack "eksctl-mycluster02-addon-iamserviceaccount-kube-system-a
ws-load-balancer-controller"
2021-10-29 23:17:57 [ℹ]  deploying stack "eksctl-mycluster02-addon-iamserviceaccount-kube-system-aws-load-balancer-control
ler"
2021-10-29 23:17:57 [ℹ]  waiting for CloudFormation stack "eksctl-mycluster02-addon-iamserviceaccount-kube-system-aws-load
-balancer-controller"
2021-10-29 23:18:13 [ℹ]  waiting for CloudFormation stack "eksctl-mycluster02-addon-iamserviceaccount-kube-system-aws-load
-balancer-controller"
2021-10-29 23:18:33 [ℹ]  waiting for CloudFormation stack "eksctl-mycluster02-addon-iamserviceaccount-kube-system-aws-load
-balancer-controller"
2021-10-29 23:18:34 [ℹ]  created serviceaccount "kube-system/aws-load-balancer-controller"
```

So to check out our roles go to aws-roles



This is going to bind the policy that we created to give access to create resources on aws to the ingress controller in our cluster.

11. Next is I'm going to create a certificate manager for the ingress controller:

```
Saidas-MBP:~ dirisova$ kubectl apply \
>     --validate=false \
>     -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/cert-manager.yaml
customresourcedefinition.apiextensions.k8s.io/certificaterequests.cert-manager.io created
customresourcedefinition.apiextensions.k8s.io/certificates.cert-manager.io created
customresourcedefinition.apiextensions.k8s.io/challenges.acme.cert-manager.io created
customresourcedefinition.apiextensions.k8s.io/clusterissuers.cert-manager.io created
```

This will apply to the ingress controller to communicate to aws

12. I'm going to download the file

```
Saidas-MBP:~ dirisova$ curl -Lo v2_3_0_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/
download/v2.3.0/v2_3_0_full.yaml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   622  100   622    0     0   2677      0 --:--:-- --:--:-- --:--:--  2764
100 31812  100 31812    0     0  79209      0 --:--:-- --:--:-- --:--:-- 2348k
Saidas-MBP:~ dirisova$
```

After downloading the file go to the local directory open the file that was downloaded v2_3_0_full.yaml (replace {cluster-name=*your-cluster-name*} with your cluster name)

13. Run this commands

```
Saidas-MBP:~ dirisova$ kubectl apply -f v2_3_0_full.yaml
customresourcedefinition.apiextensions.k8s.io/ingressclassparams.elbv2.k8s.aws created
customresourcedefinition.apiextensions.k8s.io/targetgroupbindings.elbv2.k8s.aws created
Warning: resource serviceaccounts/aws-load-balancer-controller is missing the kubectl.kubernetes.io/last-applied-configura
tion annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively b
y either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.
serviceaccount/aws-load-balancer-controller configured
role.rbac.authorization.k8s.io/aws-load-balancer-controller-leader-election-role created
clusterrole.rbac.authorization.k8s.io/aws-load-balancer-controller-role created
rolebinding.rbac.authorization.k8s.io/aws-load-balancer-controller-leader-election-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/aws-load-balancer-controller-rolebinding created
service/aws-load-balancer-webhook-service created
deployment.apps/aws-load-balancer-controller created
certificate.cert-manager.io/aws-load-balancer-serving-cert created
issuer.cert-manager.io/aws-load-balancer-selfsigned-issuer created
mutatingwebhookconfiguration.admissionregistration.k8s.io/aws-load-balancer-webhook created
validatingwebhookconfiguration.admissionregistration.k8s.io/aws-load-balancer-webhook created
Saidas-MBP:~ dirisova$ kubectl get deployment -n kube-system aws-load-balancer-controller
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   1/1     1            1           32s
Saidas-MBP:~ dirisova$
```

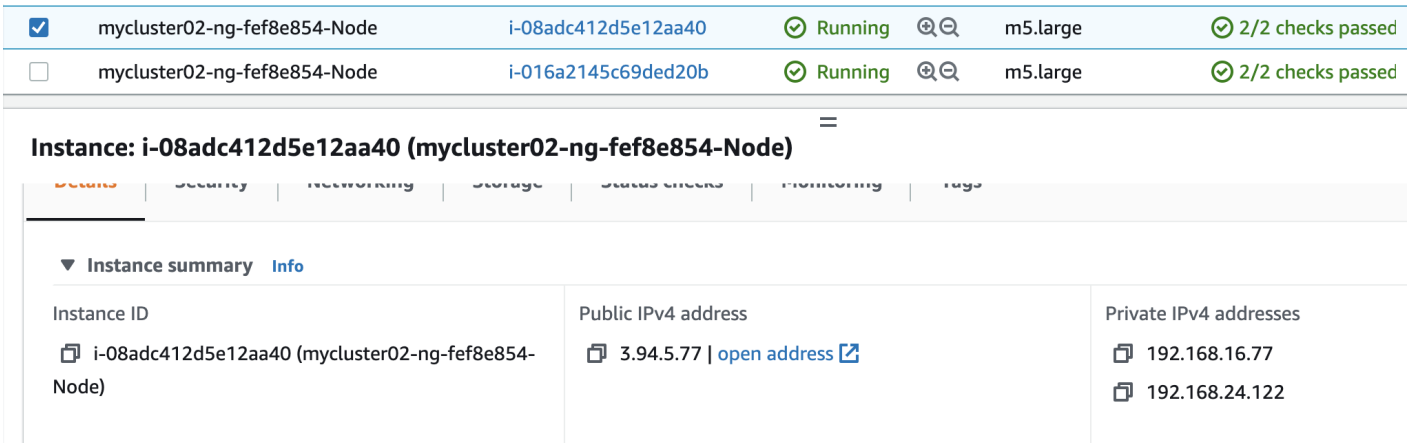
14. Now I'm going to deploy our application. So to do that I have to apply or create our yaml file, let's do so by doing

- kubectl apply nginx.yaml
  (deployment.apps/sample-app created
  service/nginx-service-nodeport created)

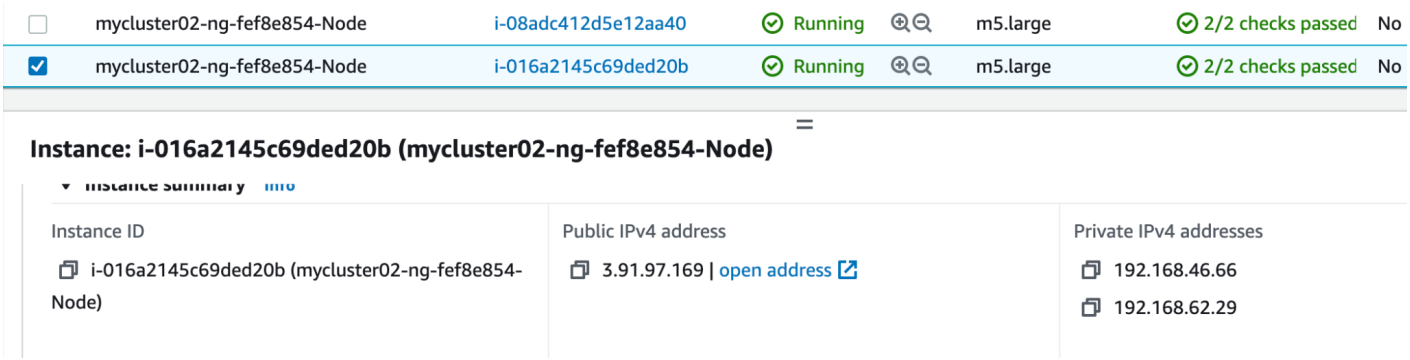
- kubectl get all

15. Go to the aws and make sure that we have our EC2 ‘s, we got 2 Private addresses in our EC2 for our nod, now we see it running we have 2 pod’s, we have the service which is call the “service/nginx-service-nodeport”, NodePort that means our application can be access through port 80:31784 on the nod

First EC2 with 2 Private IP addresses



Second EC2 with 2 Private IP addresses



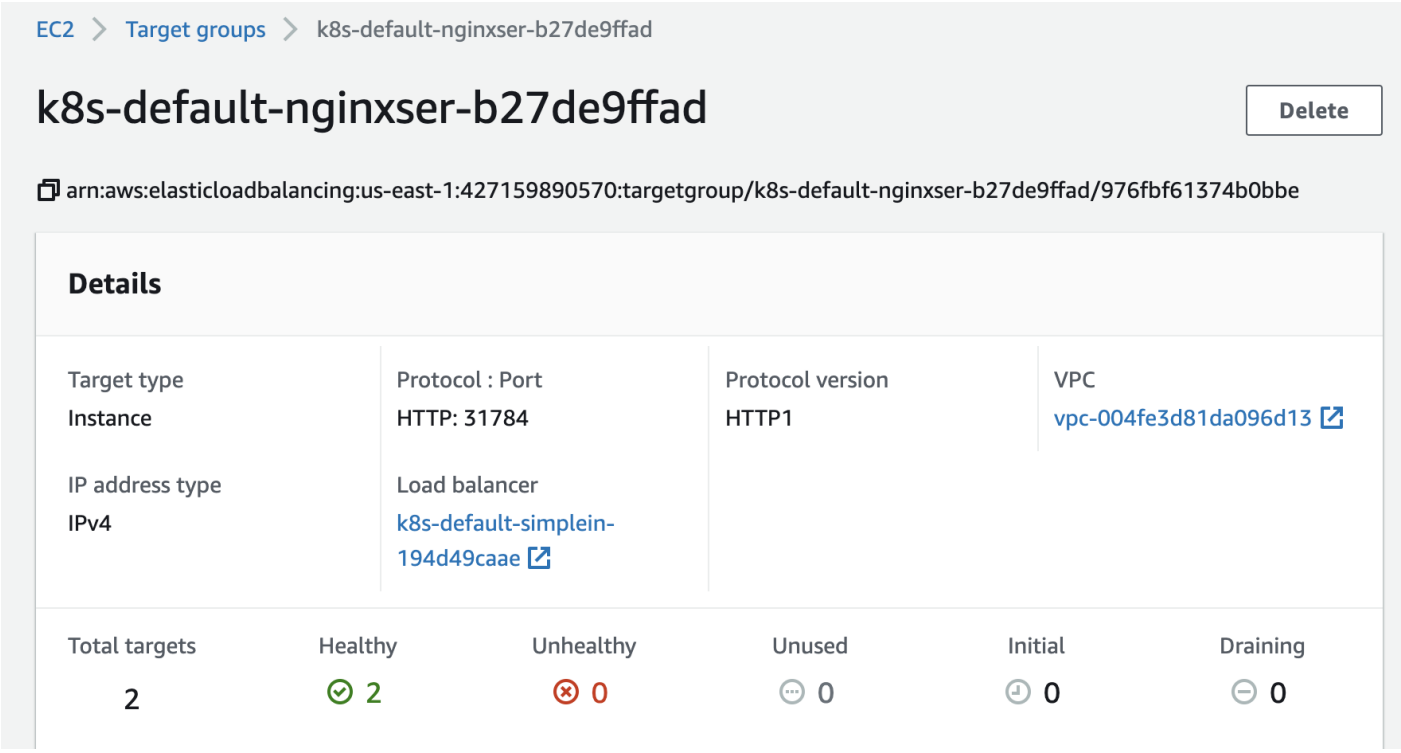
16. Next part lets create ingress controller by running our created yaml file.

- kubectl apply -f ingress.yaml

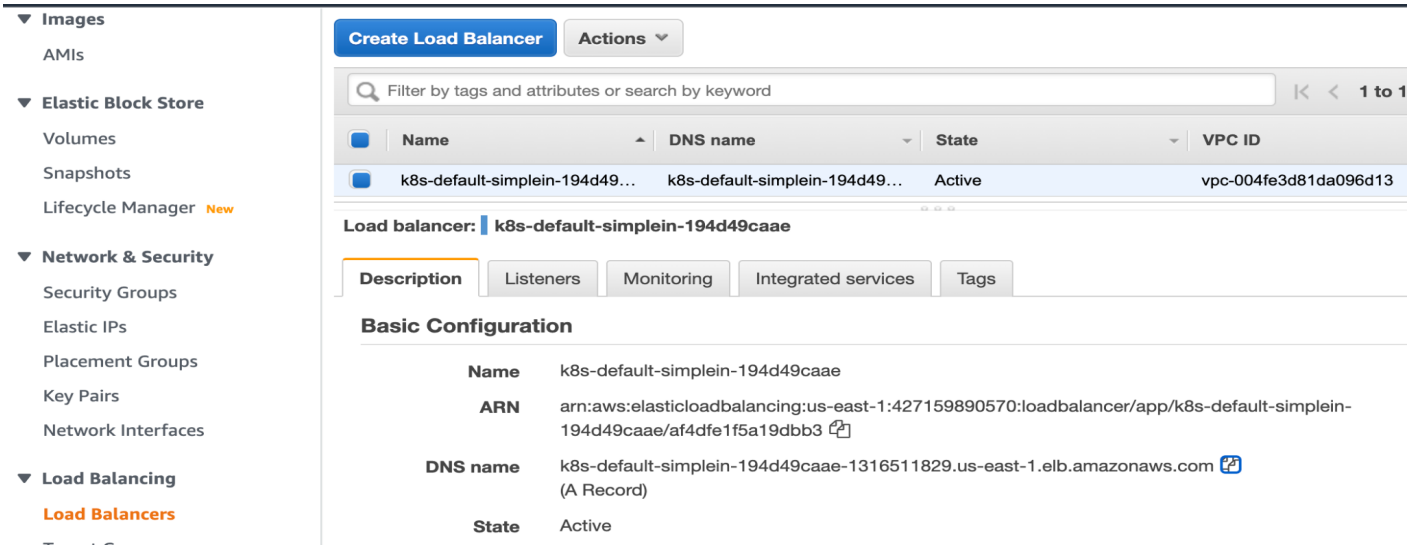
ingress.networking.k8s.io/simple-ingress created

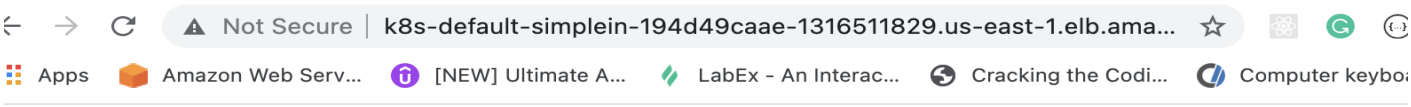
- kubectl get ingress.networking.k8s.io
- kubectl get ingress.networking.k8s.io -o wide
- kubectl describe ingress.networking.k8s.io

17. Go to the aws-so we can see we have LoadBalancer - Listeners - select listeners - edit rules
Also, go to the target group so we can see our target group which we just created is healthy,



so we are going back to Load Balancers - Description - copy DNS name - open new tab - paste and we have our new application





Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.