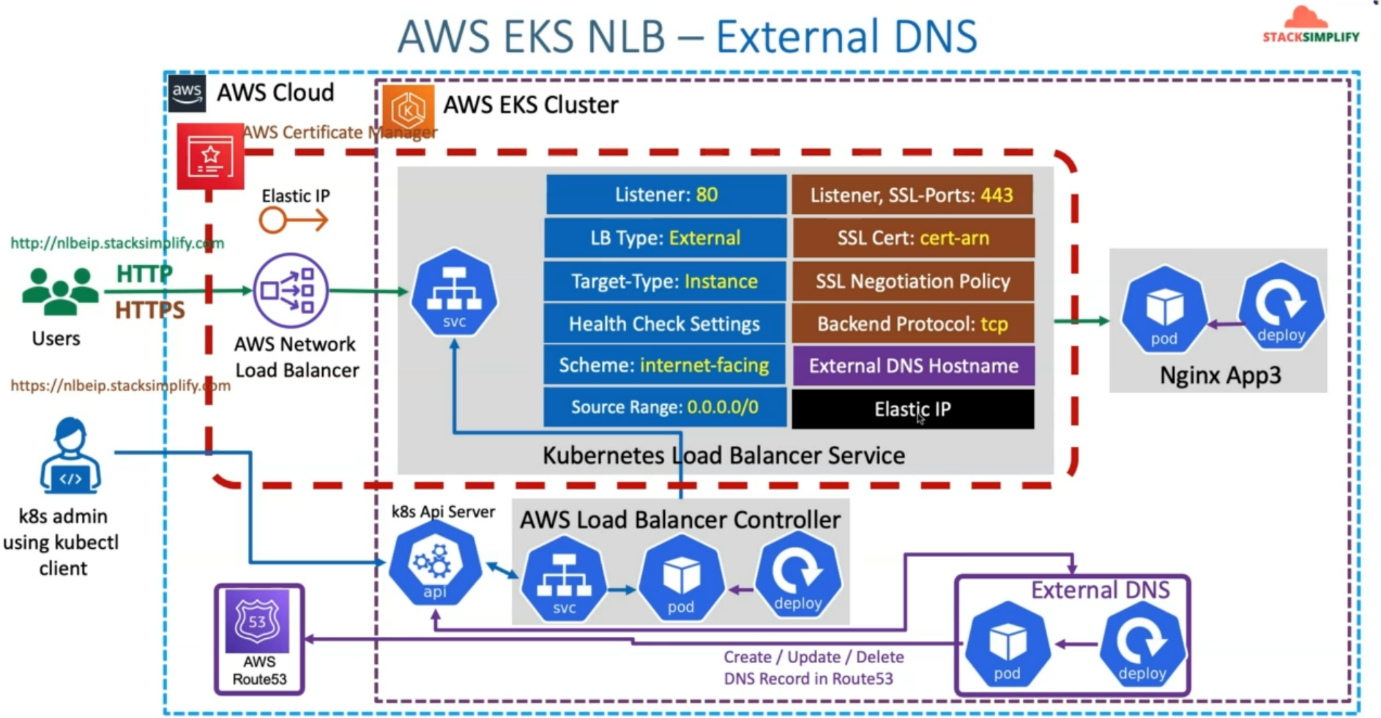
**7. NLB Elastic IPs Demo using k8s Service**

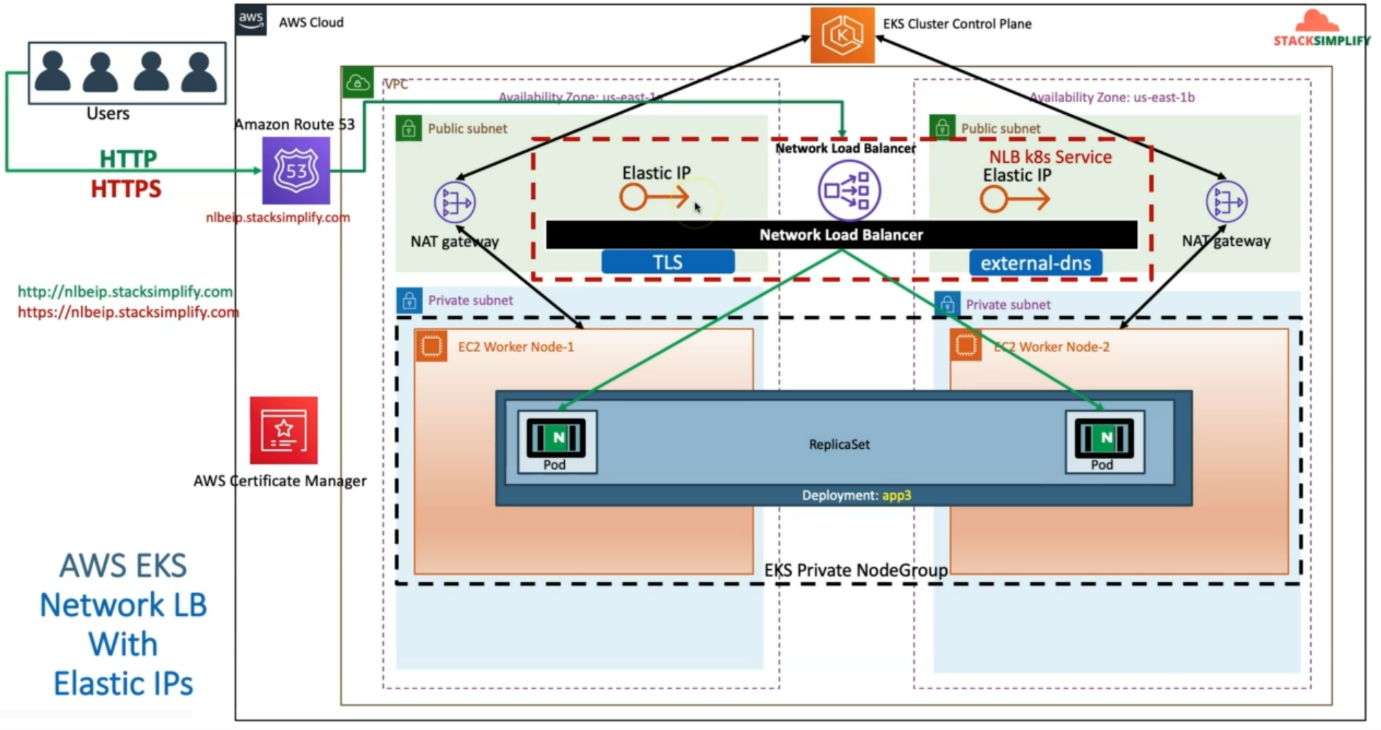
--- Reference - <https://github.com/stacksimplify/aws-eks-kubernetes-masterclass/tree/master/19-ELB-Network-LoadBalancers-with-LBC/19-04-LBC-NLB-ElasticIP>

--- in this demo, we are going to create a network load balancer with Kubernetes service and ensure that that network load balancer uses elastic IP.



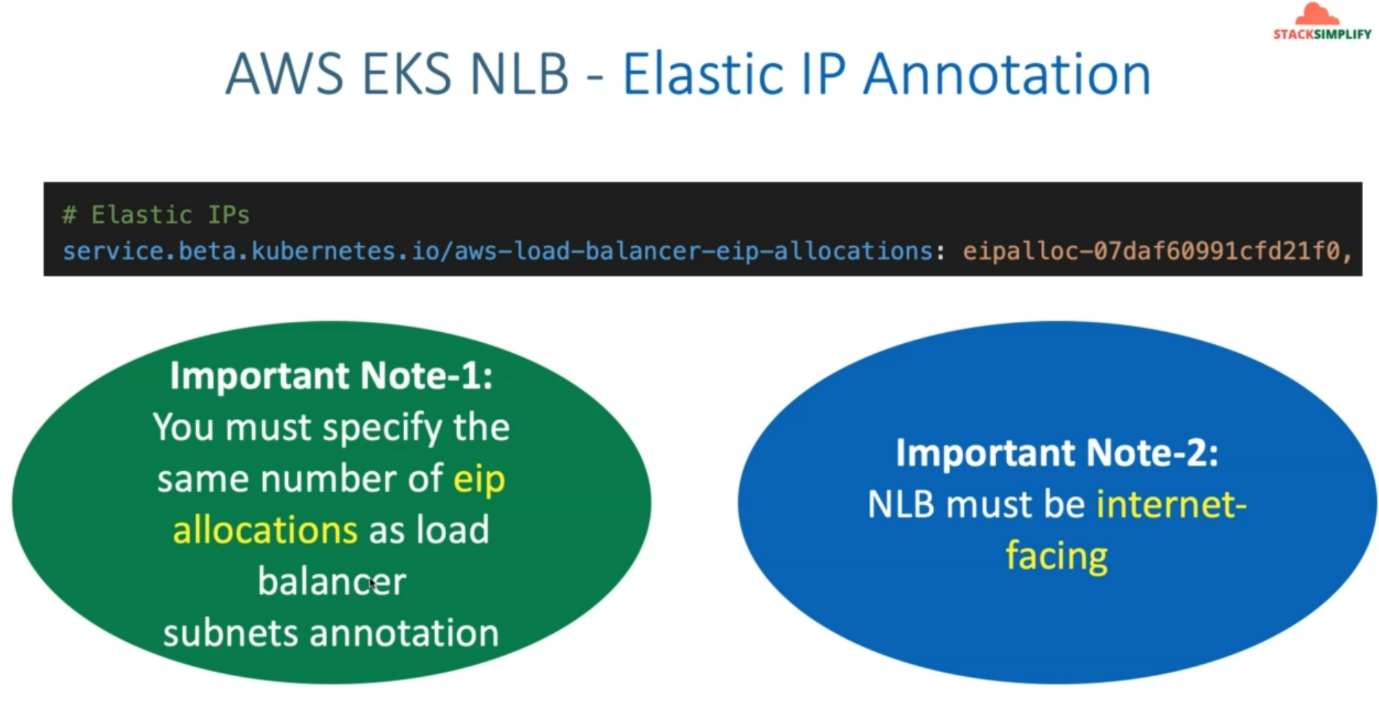
--- So first, we will create elastic IP’S and take those elastic IP allocation id and update them in our elastic IP related annotation in our Kubernetes Load Balancer Service Manifest.

--- **note** - from network design perspective, there are no changes.



--- Only thing is we are going to add additional elastic APIs here in the public subnet.

**Elastic ip annotation**



--- One important thing here is whenever we add this elastic IP related annotation in aws load balancer IP allocations.

--- **important** - So, we need to ensure that these elastic IP are directly proportional to the number of subnet where your network load balancer is getting created.

--- which means we are creating network load balancer in the public. our elastic IP’s should be equal to the number of subnet.

--- **note** – you’re NLB must be in the internet facing edge for these elastic IP to be associated for your NLB.

--- network load balancers is not only deals with the TCP but also deals with the UDP and also TLS connections.

--- some applications which are using UDP. instead of DNS, if they expect to connect to your NLB using IP address. at that point of time, giving a static elastic IP, which never changes, is a reasonable option.

--- at that point of time. this elastic IP association to our network load balancer will be a big advantage.

**Introduction**

--- Create Elastic IPs

--- Update NLB Service k8s manifest with Elastic IP Annotation with EIP Allocation IDs

**Create two Elastic IPs and get EIP Allocation IDs**

--- This configuration is optional and use can use it to assign static IP addresses to your NLB

--- You must specify the same number of ip allocations as load balancer subnets annotation

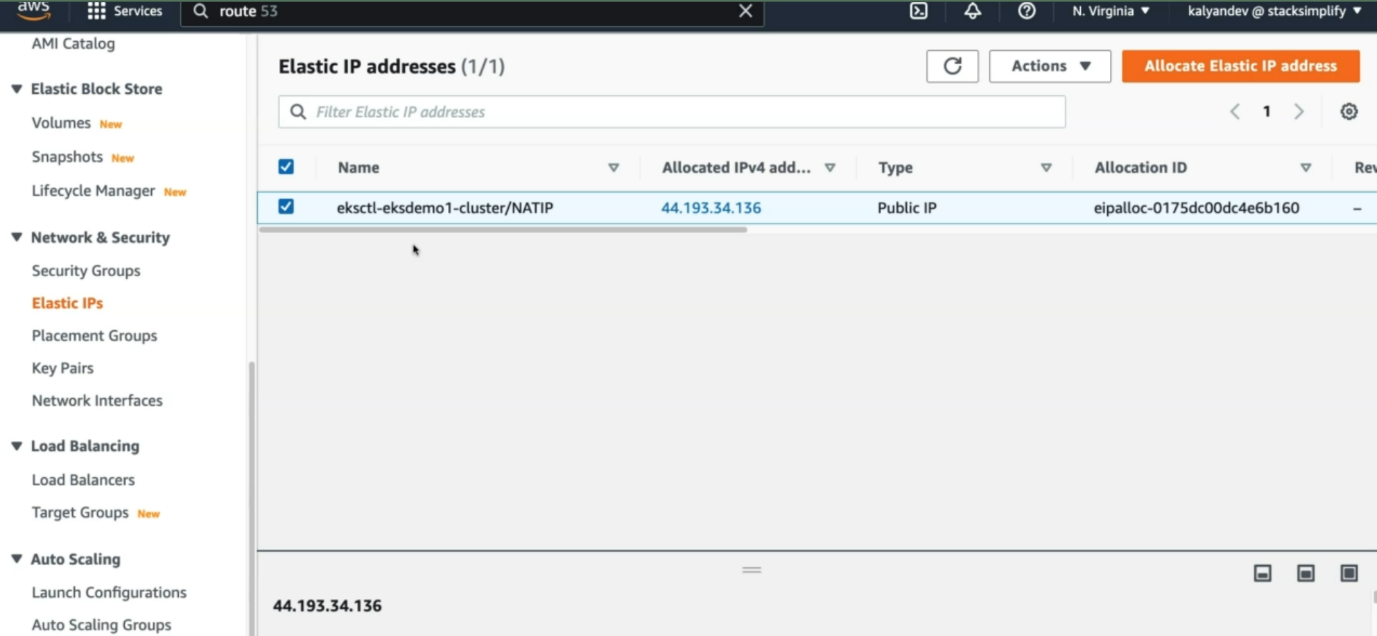
--- NLB must be internet-facing

**# Elastic IP Allocation IDs**

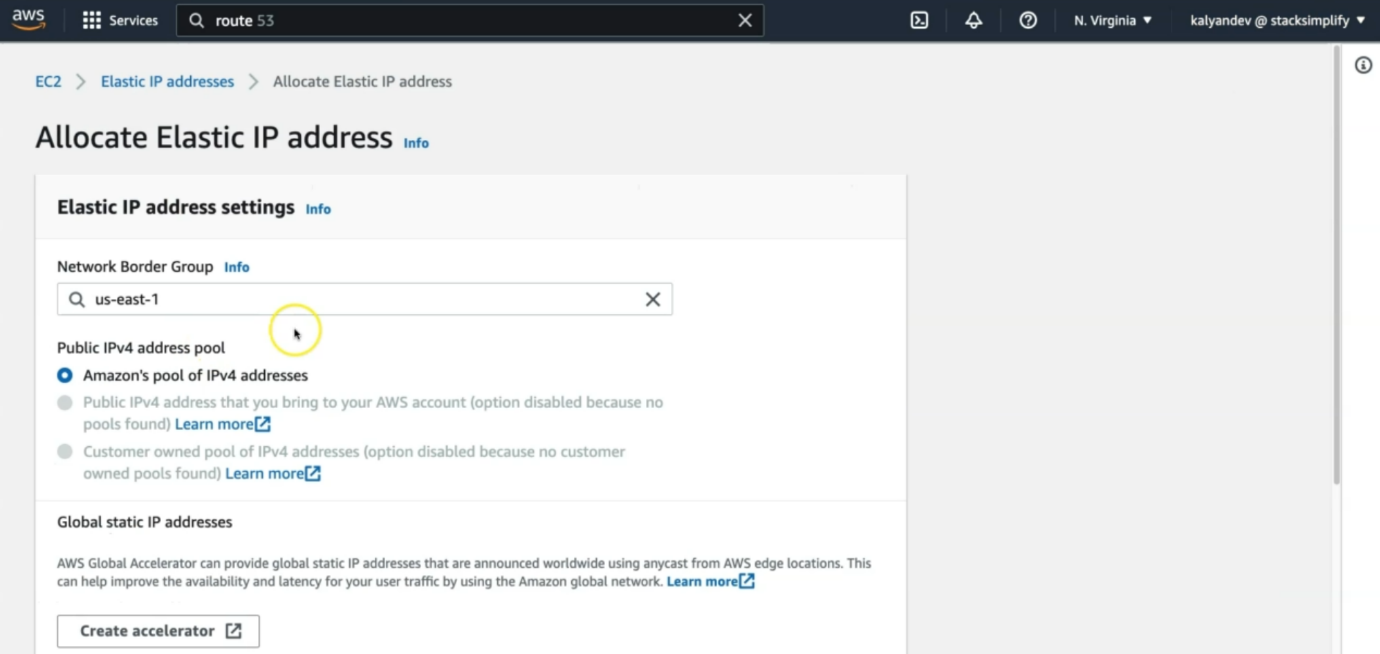
eipalloc-07daf60991cfd21f0

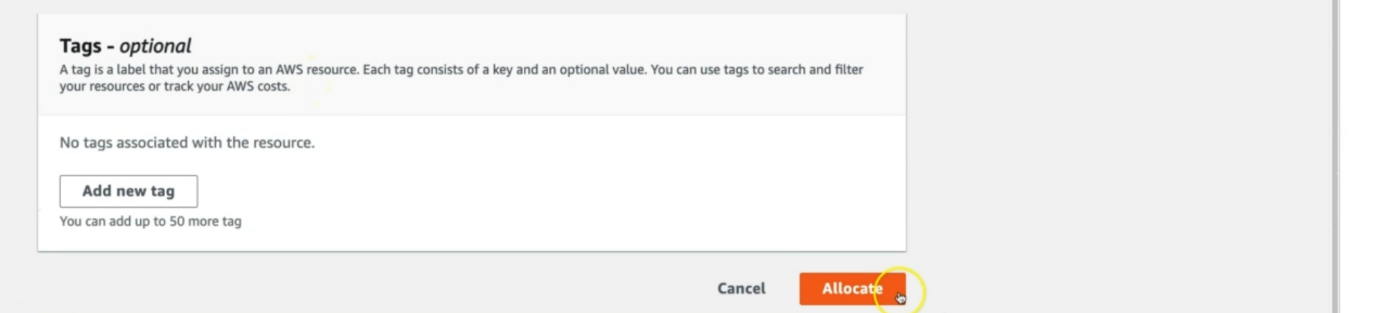
eipalloc-0a8e8f70a6c735d16

--- steps to create elastic ip.



--- click on allocate Elastic IP address.





--- this is how, you should create a elastic IP.

**Review Elastic IP Annotations**

--- **File Name: kube-manifests\02-LBC-NLB-LoadBalancer-Service.yml**

**# Elastic IPs**

**service.beta.kubernetes.io/aws-load-balancer-eip-allocations: eipalloc-07daf60991cfd21f0, eipalloc-0a8e8f70a6c735d16**

apiVersion: v1

kind: Service

metadata:

  name: elasticip-lbc-network-lb

  annotations:

    # Traffic Routing

    service.beta.kubernetes.io/aws-load-balancer-name: elasticip-lbc-network-lb

    service.beta.kubernetes.io/aws-load-balancer-type: external

    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: instance

    #service.beta.kubernetes.io/aws-load-balancer-subnets: subnet-xxxx, mySubnet ## Subnets are auto-discovered if this annotation is not specified, see Subnet Discovery for further details.

    # Health Check Settings

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-protocol: http

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-port: traffic-port

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-path: /index.html

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-healthy-threshold: "3"

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-unhealthy-threshold: "3"

    service.beta.kubernetes.io/aws-load-balancer-healthcheck-interval: "10" # The controller currently ignores the timeout configuration due to the limitations on the AWS NLB. The default timeout for TCP is 10s and HTTP is 6s.

    # Access Control

    service.beta.kubernetes.io/load-balancer-source-ranges: 0.0.0.0/0

    service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"

    # AWS Resource Tags

    service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags: Environment=dev,Team=test

    # TLS

    service.beta.kubernetes.io/aws-load-balancer-ssl-cert: arn:aws:acm:us-east-1:180789647333:certificate/d86de939-8ffd-410f-adce-0ce1f5be6e0d

    service.beta.kubernetes.io/aws-load-balancer-ssl-ports: 443, # Specify this annotation if you need both TLS and non-TLS listeners on the same load balancer

    service.beta.kubernetes.io/aws-load-balancer-ssl-negotiation-policy: ELBSecurityPolicy-TLS13-1-2-2021-06

    service.beta.kubernetes.io/aws-load-balancer-backend-protocol: tcp

    # External DNS - For creating a Record Set in Route53

    external-dns.alpha.kubernetes.io/hostname: nlbeip201.stacksimplify.com

    # Elastic IPs

    service.beta.kubernetes.io/aws-load-balancer-eip-allocations: eipalloc-068b65c8e0df2b53e, eipalloc-022d66b51f98706c6

spec:

  type: LoadBalancer

  selector:

    app: app3-nginx

  ports:

    - name: http

      port: 80

      targetPort: 80

    - name: https

      port: 443

      targetPort: 80

--- **01-Nginx-App3-Deployment.yml**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: app3-nginx-deployment

  labels:

    app: app3-nginx

spec:

  replicas: 1

  selector:

    matchLabels:

      app: app3-nginx

  template:

    metadata:

      labels:

        app: app3-nginx

    spec:

      containers:

        - name: app2-nginx

          image: stacksimplify/kubenginx:1.0.0

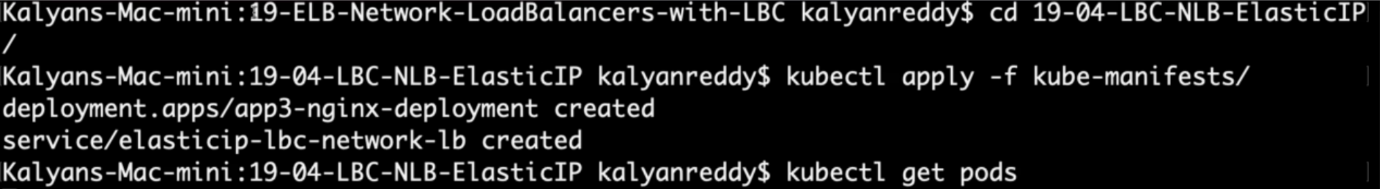
          ports:

            - containerPort: 80

**Deploy all kube-manifests**

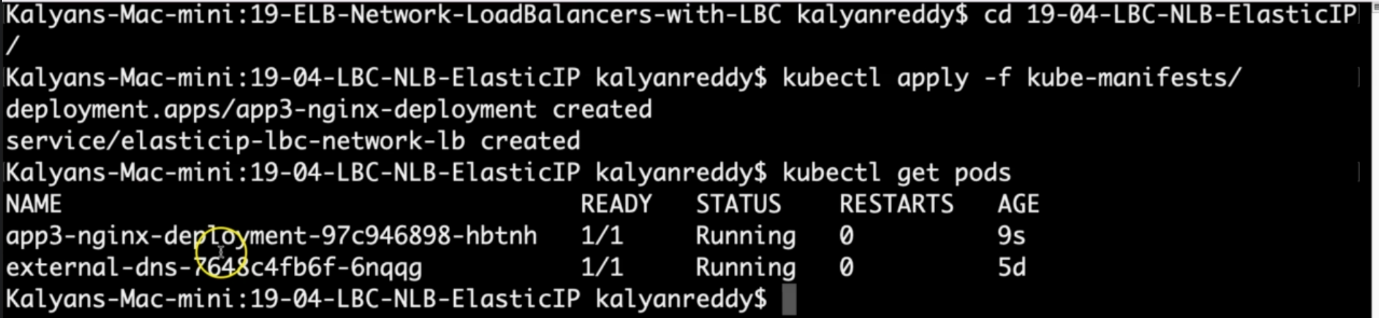
**# Deploy kube-manifests**

--- **kubectl apply -f kube-manifests/**



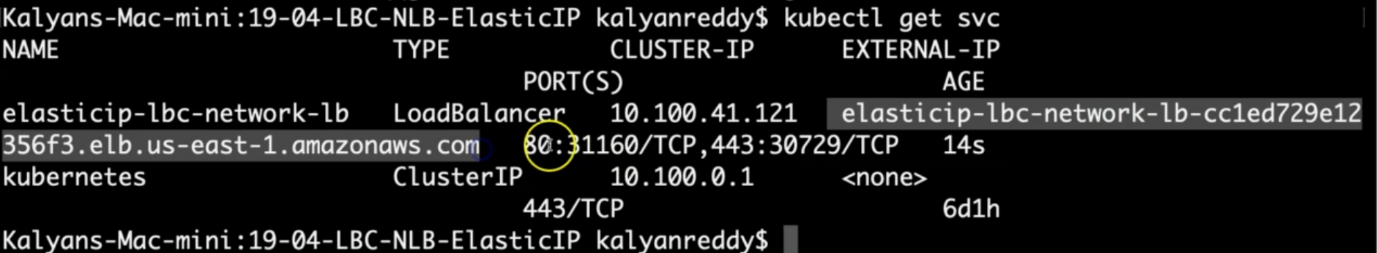
**# Verify Pods**

--- **kubectl get pods**



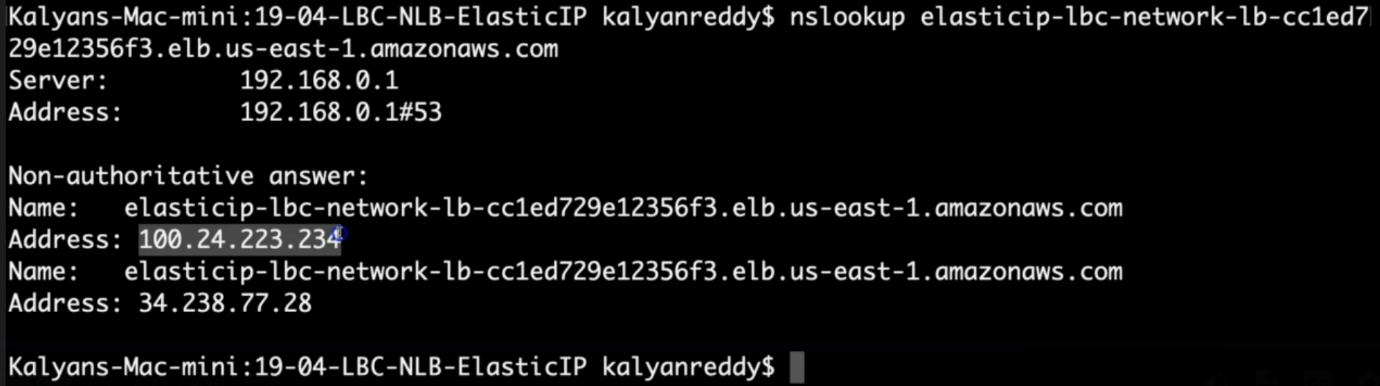
**# Verify Services**

--- **kubectl get svc**



Observation:

1. Verify the network lb DNS name



--- **note** - the nslookup is resolved to elastic ip addresses.

**# Verify AWS Load Balancer Controller pod logs**

--- **kubectl -n kube-system get pods**

--- **kubectl -n kube-system logs -f <aws-load-balancer-controller-POD-NAME>**

**# Verify using AWS Mgmt Console**

Go to Services -> EC2 -> Load Balancing -> Load Balancers

1. Verify Description Tab - DNS Name matching output of "kubectl get svc" External IP

2. Verify Listeners Tab

Observation: Should see two listeners Port 80 and 443

Go to Services -> EC2 -> Load Balancing -> Target Groups

1. Verify Registered targets

2. Verify Health Check path

Observation: Should see two target groups. 1 Target group for 1 listener

**# Perform nslookup Test**

--- nslookup nlbeip201.stacksimplify.com

Observation:

1. Verify the IP Address matches our Elastic IPs we created in Step-02

**# Access Application**

**# Test HTTP URL**

--- <http://nlbeip201.stacksimplify.com>

**# Test HTTPS URL**

--- <https://nlbeip201.stacksimplify.com>

**Clean-Up**

**# Delete or Undeploy kube-manifests**

--- **kubectl delete -f kube-manifests/**

**# Delete Elastic IPs created**

In AWS Mgmt Console,

Go to Services -> EC2 -> Network & Security -> Elastic IPs

Delete two EIPs created

**# Verify if NLB deleted**

In AWS Mgmt Console,

Go to Services -> EC2 -> Load Balancing -> Load Balancers

**References**

--- Network Load Balancer - <https://docs.aws.amazon.com/eks/latest/userguide/network-load-balancing.html>

--- NLB Service - <https://kubernetes-sigs.github.io/aws-load-balancer-controller/v2.4/guide/service/nlb/>

--- NLB Service Annotations - <https://kubernetes-sigs.github.io/aws-load-balancer-controller/v2.4/guide/service/annotations/>