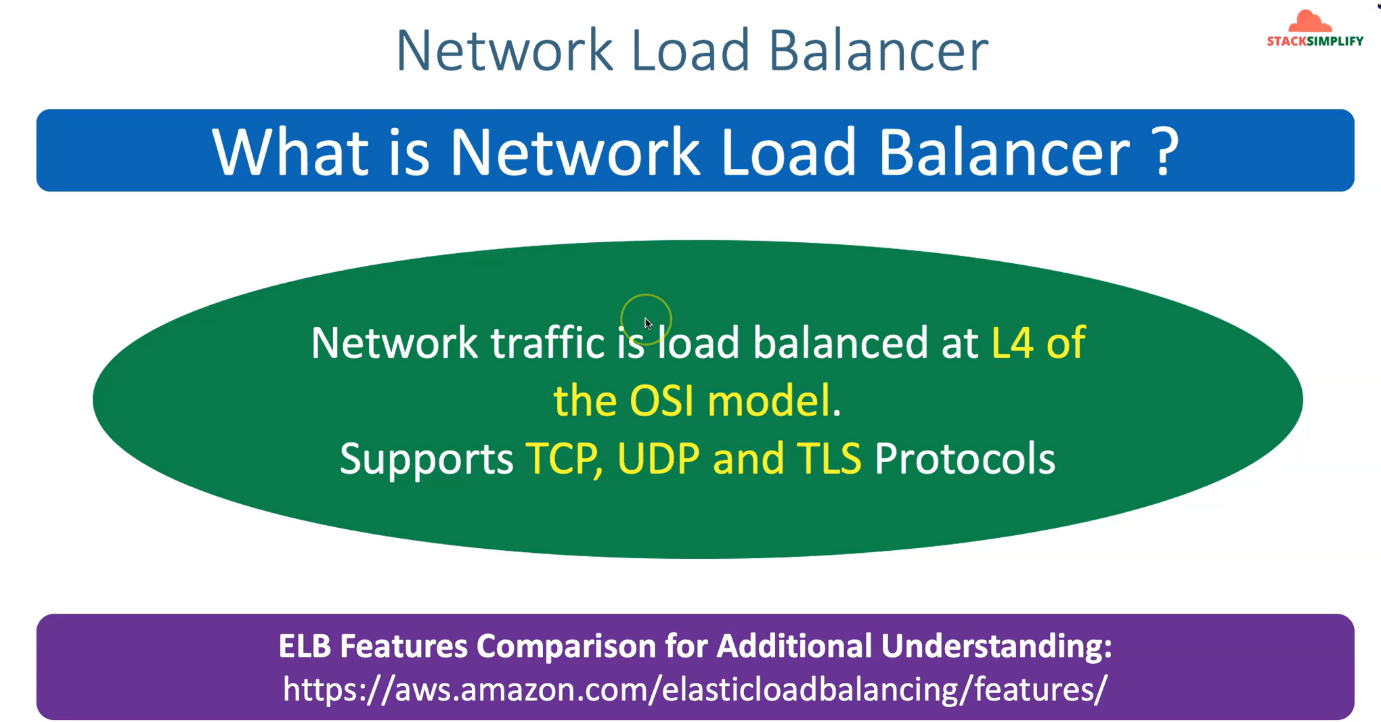
**01. Introduction to Network Load Balancer with k8s Service**

--- in this section, we are going to implement AWS network load balancer using Kubernetes service and aws EKS load Balancer controller.

--- whenever we have implemented it in Section seven, it is with aws cloud provider load balancer controller. Which is legacy one. Aws has provided a dedicated load balancer controller, where in, you will be able to create application load balancer and network load balancer.

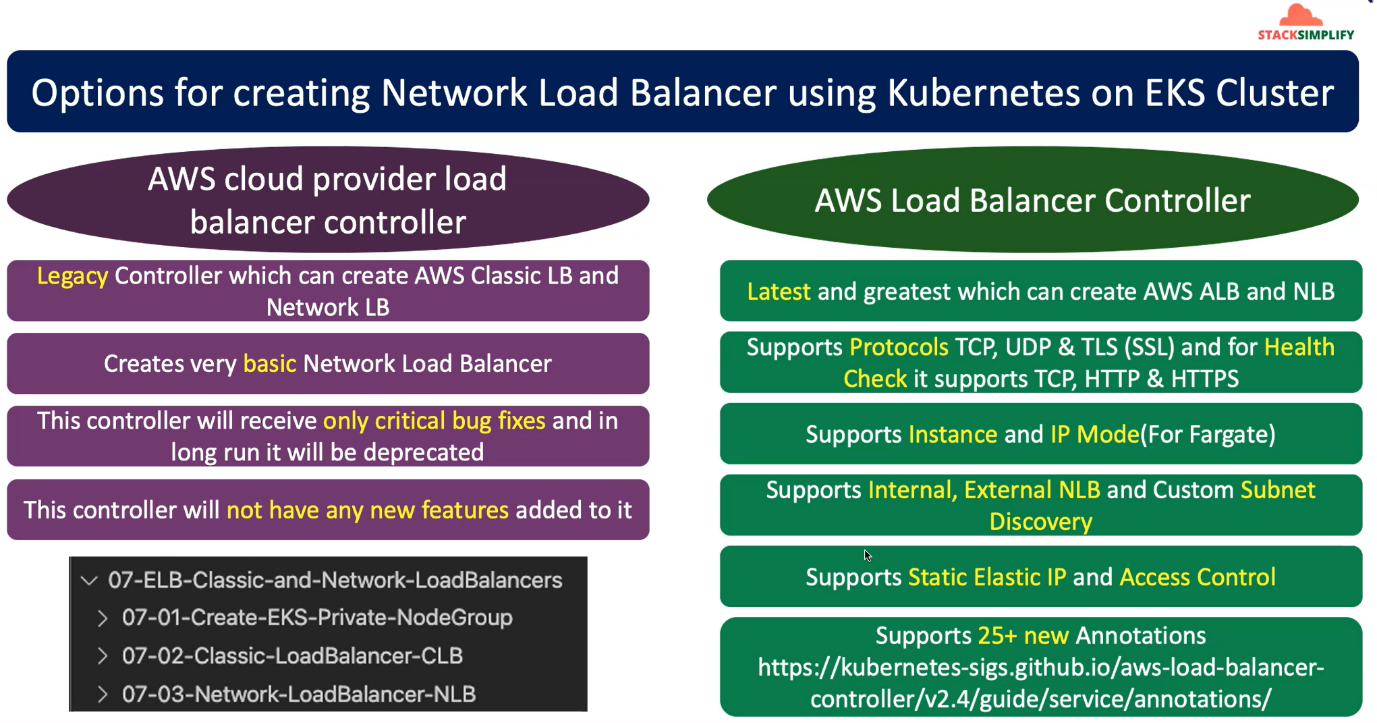
--- let's see how to create this network load balancer and a little background about all these things in detail before going into practical implementation.

**What is this network load balancer?**



**options for creating network load balancer using Kubernetes on EKS**

**cluster**



--- So, options for creating network load balancer using Kubernetes on EKS cluster. what are the options? we have two options.

--- one is aws cloud provider load balancer controllers and another controller is aws load balancer controller and if you see this aws cloud provider load balancer controller, it is a legacy controller which can create a classic LB and network lb.

--- this is a load balancer controller can create latest. which can create latest ALB and NLB and this cloud provider load balancer controller.

--- **NOTE** - AWS cloud provider load balancer controller installed automatically whenever the EKS cluster control plane is getting created.

--- when coming to aws Load Balancer controller, so you need to specifically go ahead and then install it. we have done that when we have implemented the application load balancer ingress concepts. now in our cluster, we already have the aws load balancer controller and also the externally DNS related pods and all those things.

--- So, this network load balancer support’s TCP, UDP & TLS(SSL) and for health checks it supports TCP, HTTP and HTTPS related protocols.

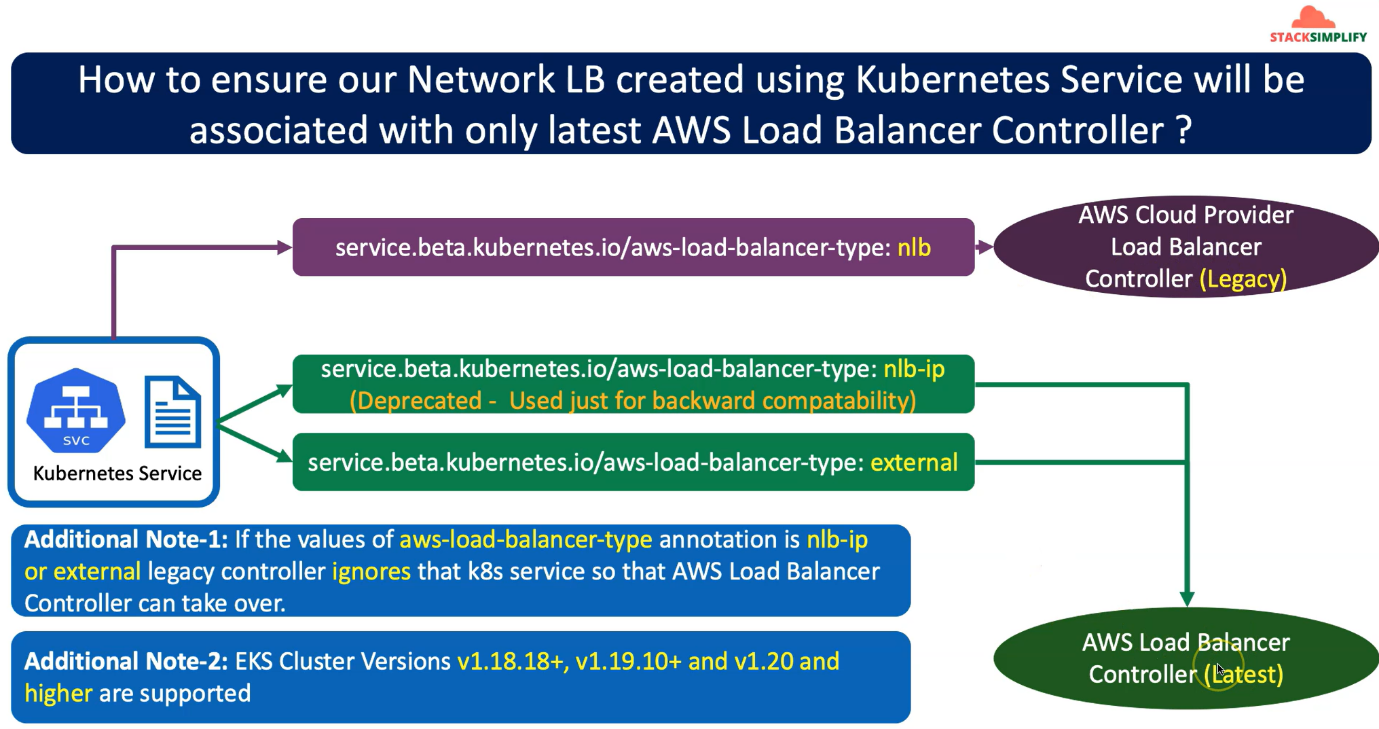
--- this network load balancer In combination with this aws load Balancer controller also supports instance and IP mode and for IP mode is primarily for fargate but you can also use it. when you are using the worker nodes also. that the requests will go directly from your load balancer to your respective pod.

--- this network load balancer also supports internal load balancer and also external load balancer. In addition to that, it also supports custom subnet discovery. which means you can define custom subnets by annotation itself

--- this network load balancer in combination with this aws load Balancer controller also support static elastic IP and also access controls.

--- 25+ new annotations available for us and we can use all those annotations to effectively configure our network load.

**Ensure out NLB created using kubernetes service**



--- So, the question here is you have created a Kubernetes service off type load balancer with an annotation related to Network Load Balancer.

--- So how will we know that or how will we ensure that that network load balancer is associated or is reconciled with aws load balancer controller. the latest one.

--- here comes the important thing. you have created your Kubernetes service manifest and you also have your cloud provider load balancer controller, which is a legacy one, by default whenever you create EKS cluster control plane. that gets installed.

--- now you also install your Load Balancer controller, which is the latest one. now here comes the important thing in the Kubernetes service manifest.

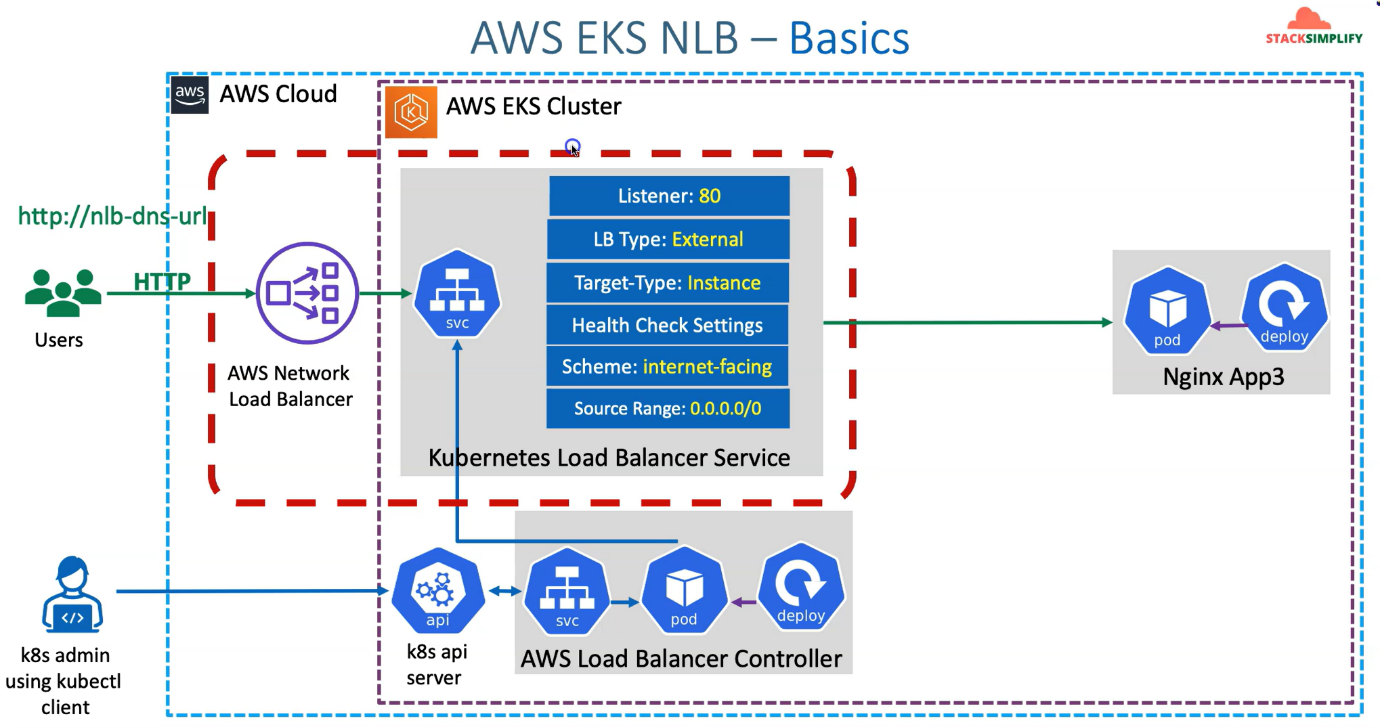
--- whenever you define your annotation for AWS load balancer type as NLB then Kubernetes service will be associated with this legacy cloud provider load balancer controller.

--- whenever you define aws load balancer type NLB-IP or external. When you define these two things, it will be associated with AWS Load Balancer controller **LATEST**.

**Kubernetes service manifest**



**Architectural diagram of AWS EKS NLB (kubernetes perspective)**



--- In AWS cloud. We have created a EKS cluster and we already installed the aws Load Balancer controller, which means its related service accounts and IAM roles all are created as part of our Ingress first section.

--- whenever you deploy your Kubernetes manifests of type service. automatically initially it will go and then check against your aws cloud provider load balancer which is legacy controller and there it finds that the load balancer type is external and at that point of time, this load aws balancer controller will pick up.

--- this Kubernetes service will be created and reconciled or associated with aws load balancer controller.

--- As part of deploying the Kubernetes service Manifest, will also deploy our nginx app3 related deployment manifest, so it will create a nginx related pod objects in EKS cluster.

--- now from Settings perspective in Kubernetes service, listeners port 80 is going to be by default.

We don't need to define any annotation for that, so it is a default one and LB type is external or load

balancer type external.

--- this is the one which is deciding that this Kubernetes load balancer service should be associated

with aws load balancer controller instead of Legacy one.

--- the next annotation you are going to define is target type is instance. either you can use instance or IP mode, and we will do with instance mode. Fargate mode we will do with IP, but you can use any of those thing’s instance or ip mode.

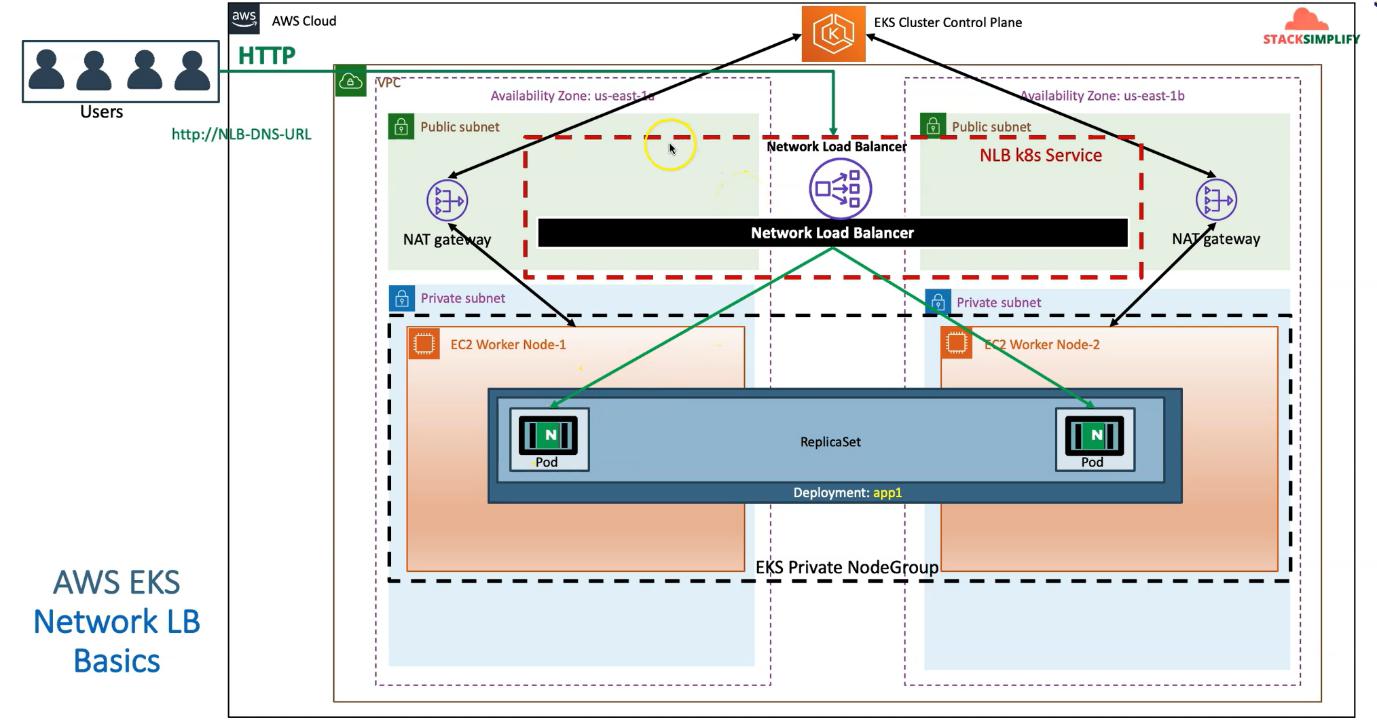
--- instance mode – the worker nodes with the node port services, which means worker nodes related node port will be added as it targets in your target group of your load balancer, so which we already discussed.

--- you can also define health check settings for your respective network load balancer from the Kubernetes service itself and the important option we have is it supports both TCP, HTTP AND HTTPS Load Balancer related health settings Even though this is a Layer four network load balancer and it operates at the layer four of the OSI model.

--- Scheme – internet facing, this load balancer is going to be internet facing.

--- source Range – 0.0.0.0/0, this load balancer can access form anywhere.

**Architectural diagram of AWS EKS NLB (AWS perspective)**



--- whenever we have created EKS cluster control plane, so it has created public subnets, private subnets and also you VPC for us in two availability zones.

--- We have also created EKS Private Node Group And we also have nat gateway automatically created for outbound communication for our worker nodes, which are present in the private subnet with EKS cluster control plane.

--- we also deployed our nginx app3, this should be changed to app3 here, but that's fine. it will create a deployment which creates the replicas and their pods inside of them and we'll create a network load balancer, which means you have deployed a Kubernetes service manifest and it created a network load balancer for us.

--- whenever you access your <http://NLB-DNS-URL>. it will come to your network load balancer and via worker node group, it will reach your pod.