**7: Create EKS Managed Node Group & IAM OIDC Provider**

**Create & Associate IAM OIDC Provider for our EKS Cluster**

--- To enable and use AWS IAM roles for Kubernetes service accounts on our EKS cluster, we must create & associate OIDC identity provider.

--- We can do this by using 2 methos. 1st one is we can do this by using below command. 2nd method is by going to aws account, in this method we have to do multiple procedures. Some times we will mess up something. That is why use 1st method because it will take care of everything for you.

--- To do so using eksctl we can use the below command.

--- **note** - Use latest eksctl version (as on today the latest version is 0.21.0)

**# Template**

--- **eksctl utils associate-iam-oidc-provider \**

**--region <region-code> \**

**--cluster <cluter-name> \**

**--approve**

**# Replace with region & cluster name**

--- **eksctl utils associate-iam-oidc-provider \**

**--region us-east-1 \**

**--cluster eksdemo1 \**

**--approve**

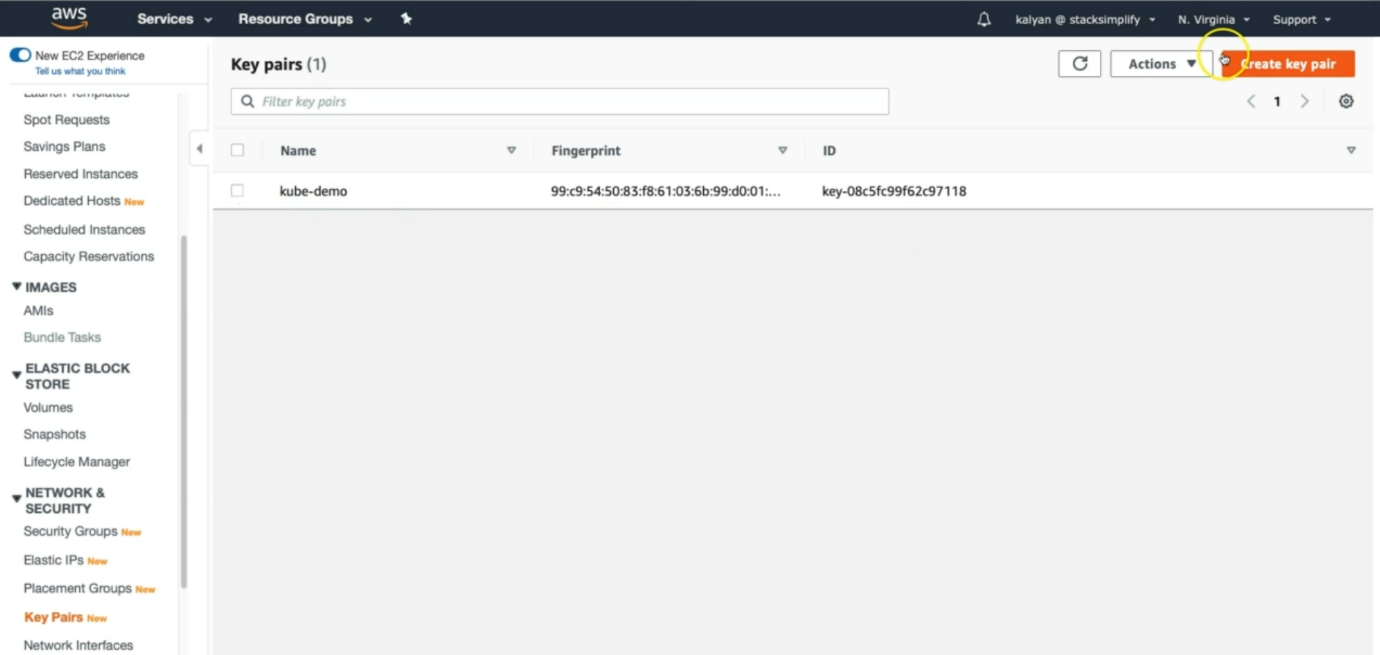
**Create EC2 Keypair on AWS**

--- Create a new EC2 Keypair with name as kube-demo

--- This keypair we will use it when creating the EKS Nodegroup.

--- This will help us to login to the EKS Worker Nodes using Terminal.

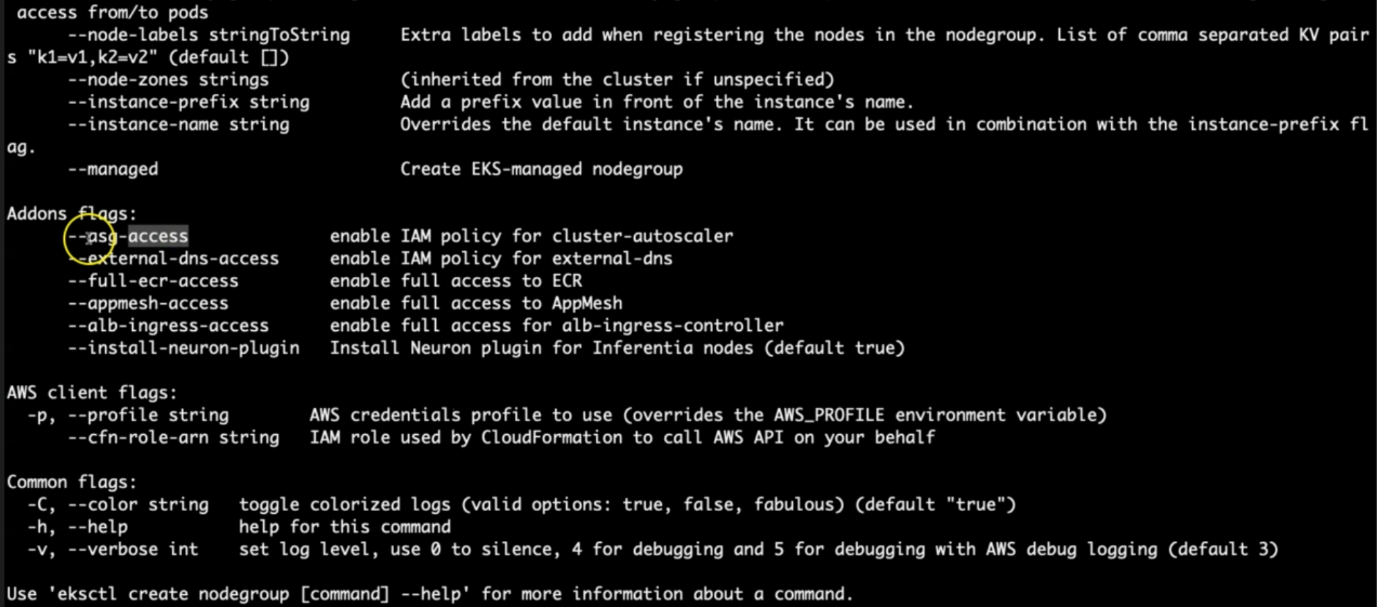
--- **ec2 -> keypair**



**Create Node Group with additional Add-Ons in Public Subnets**

--- These add-ons will create the respective IAM policies for us automatically within our Node Group role.

--- **eksctl create nodegroup --help** – it will give you the more options. Like how you can crate nodegroups and what options you can use it.



--- **asg-access** – it enables the IAM policy for cluster autoscaler in the worker nodes.

--- **external dns access** – whenever you crate ingress controller or load balancer automatically your route53 dns also register, that related IAM policy will get created.

--- **full-ecr-access** – elastic container register related policy will get created.

---

**# Create Public Node Group**

--- **eksctl create nodegroup --cluster=eksdemo1 \**

**--region=us-east-1 \**

**--name=eksdemo1-ng-public1 \**

**--node-type=t3.medium \**

**--nodes=2 \**

**--nodes-min=2 \**

**--nodes-max=4 \**

**--node-volume-size=20 \**

**--ssh-access \**

**--ssh-public-key=kube-demo \**

**--managed \**

**--asg-access \**

**--external-dns-access \**

**--full-ecr-access \**

**--appmesh-access \**

**--alb-ingress-access**

--- **--cluster=eksdemo1** – the name of the cluster.

--- **--name=eksdemo1-ng-public1** – the name of the node group.

--- **ssh-access** – we are enabling ssh access on the worker nodes.

--- **ssh-public-key=kube.demo** – while creating node groups, we are giving public key, this key created in the previous step. We can connect the worker node using private key.

--- **managed** – earlier aws provided un managed worker nodes but now aws providing managed worker nodes, in future everything going to be managed by aws so we will use this option.

---