**05. Create STS Assume IAM Role for CodeBuild to interact with AWS EKS**

--- Reference - <https://github.com/stacksimplify/aws-eks-kubernetes-masterclass/tree/master/11-DevOps-with-AWS-Developer-Tools>

--- we are going to create an STS role for Codebuild with to interact with aws EKS. The Codebuild will be a service which will be doing over the docker image build and then also doing the respective deployment to our EKS cluster.

--- it needs some additional permission. let's see, what are those? in AWS code pipeline, we are going to use Codebuild to deploy changes to kubernetes manifest.

--- this requires an AWS IAM role capable of interacting with EKS cluster.

--- In this step, we are going to create an IAM role and add an inline policy **EKS:Describe** that we will use in the CodeBuild stage to interact with the EKS cluster via kubectl.

**Create STS Assume IAM Role for CodeBuild to interact with AWS EKS**

--- In an AWS CodePipeline, we are going to use AWS CodeBuild to deploy changes to our Kubernetes manifests.

--- This requires an AWS IAM role capable of interacting with the EKS cluster.

--- In this step, we are going to create an IAM role and add an inline policy EKS:Describe that we will use in the CodeBuild stage to interact with the EKS cluster via kubectl.

**# Export your Account ID**

--- export ACCOUNT\_ID=180789647333

**# Set Trust Policy**

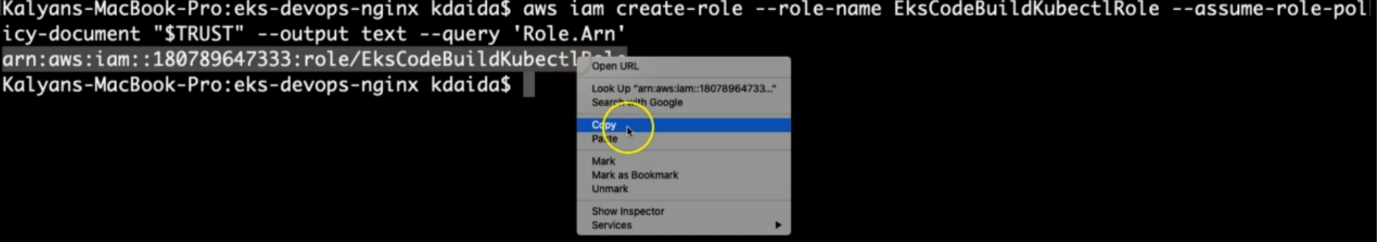
--- TRUST="{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\", \"Principal\": { \"AWS\": \"arn:aws:iam::${ACCOUNT\_ID}:root\" }, \"Action\": \"sts:AssumeRole\" } ] }"

**# Verify inside Trust policy, your account id got replacd**

--- echo $TRUST

**# Create IAM Role for CodeBuild to Interact with EKS**

--- aws iam create-role --role-name **EksCodeBuildKubectlRole** --assume-role-policy-document "$TRUST" --output text --query 'Role.Arn'



--- **note** – if you want then make a note of that arn.

--- **EksCodeBuildKubectlRole –** this is the role name.

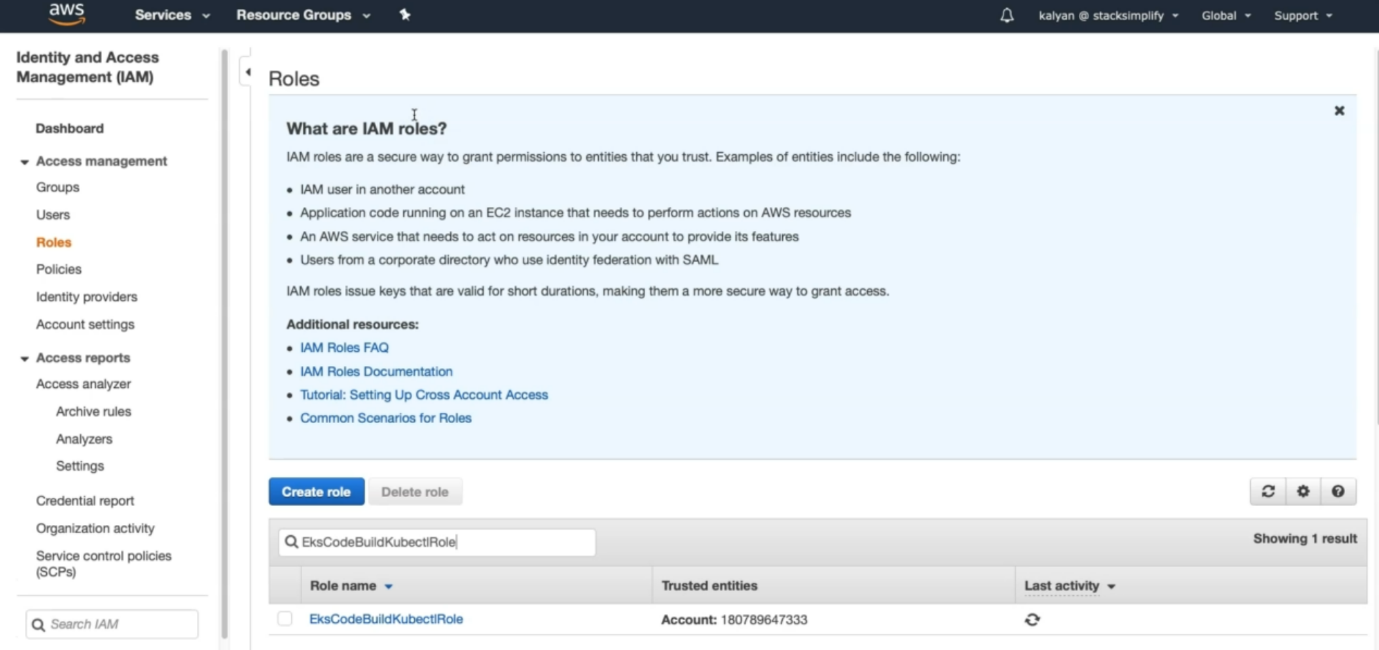
**# Define Inline Policy with eks Describe permission in a file iam-eks-describe-policy**

--- echo '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "eks:Describe\*", "Resource": "\*" } ] }' > /tmp/iam-eks-describe-policy

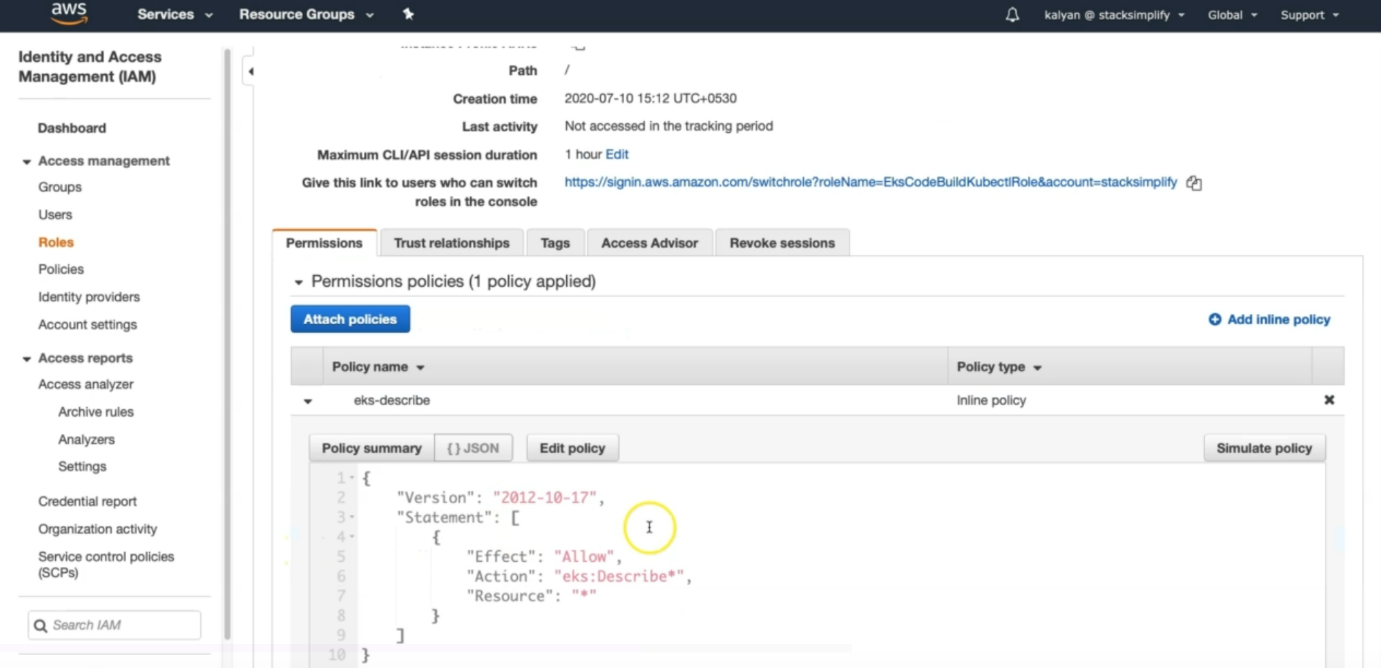
**# Associate Inline Policy to our newly created IAM Role**

--- aws iam put-role-policy --role-name EksCodeBuildKubectlRole --policy-name eks-describe --policy-document file:///tmp/iam-eks-describe-policy

**# Verify the same on Management Console**



--- click on the role



--- one important thing is path / and see the permissions.

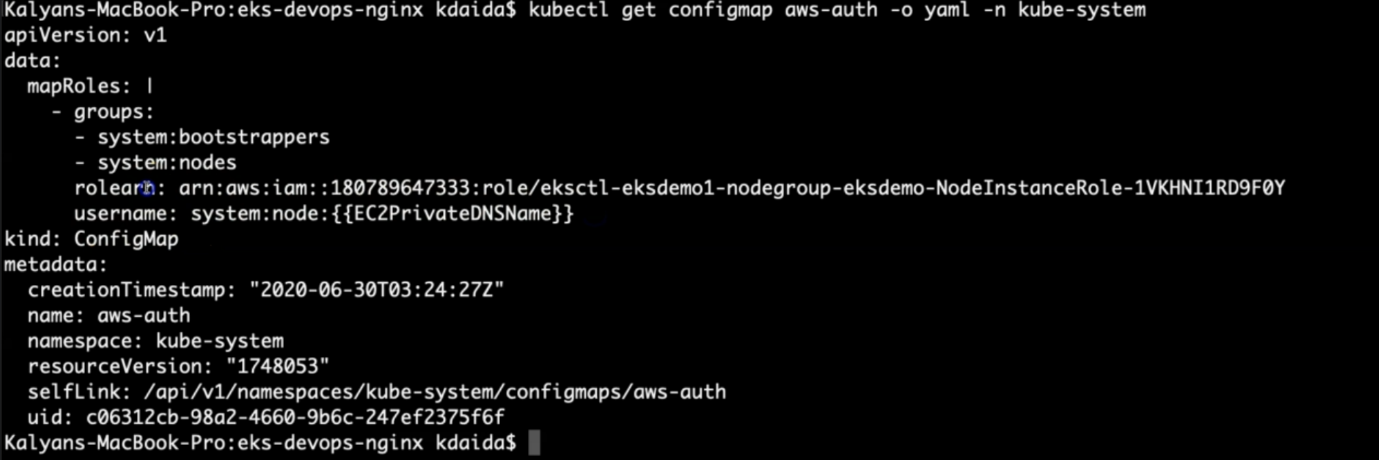
**Update EKS Cluster aws-auth ConfigMap with new role created in previous step**

--- We are going to add the role to the aws-auth ConfigMap for the EKS cluster. This aws-auth will be present inside of EKS cluster.

--- Once the EKS aws-auth ConfigMap includes this new role, kubectl in the CodeBuild stage of the pipeline will be able to interact with the EKS cluster via the IAM role.

**# Verify what is present in aws-auth configmap before change**

--- **kubectl get configmap aws-auth -o yaml -n kube-system**



**# Export your Account ID**

--- **export ACCOUNT\_ID=180789647333**

**# Set ROLE value**

--- **ROLE=" - rolearn: arn:aws:iam::$ACCOUNT\_ID:role/EksCodeBuildKubectlRole\n username: build\n groups:\n - system:masters"**

--- **note** - this role is for system masters.

**# Get current aws-auth configMap data and attach new role info to it**

--- **kubectl get -n kube-system configmap/aws-auth -o yaml | awk "/mapRoles: \|/{print;print \"$ROLE\";next}1" > /tmp/aws-auth-patch.yml**

**# Patch the aws-auth configmap with new role**

--- **kubectl patch configmap/aws-auth -n kube-system --patch "$(cat /tmp/aws-auth-patch.yml)"**

**# Verify what is updated in aws-auth configmap after change**

--- **kubectl get configmap aws-auth -o yaml -n kube-system**