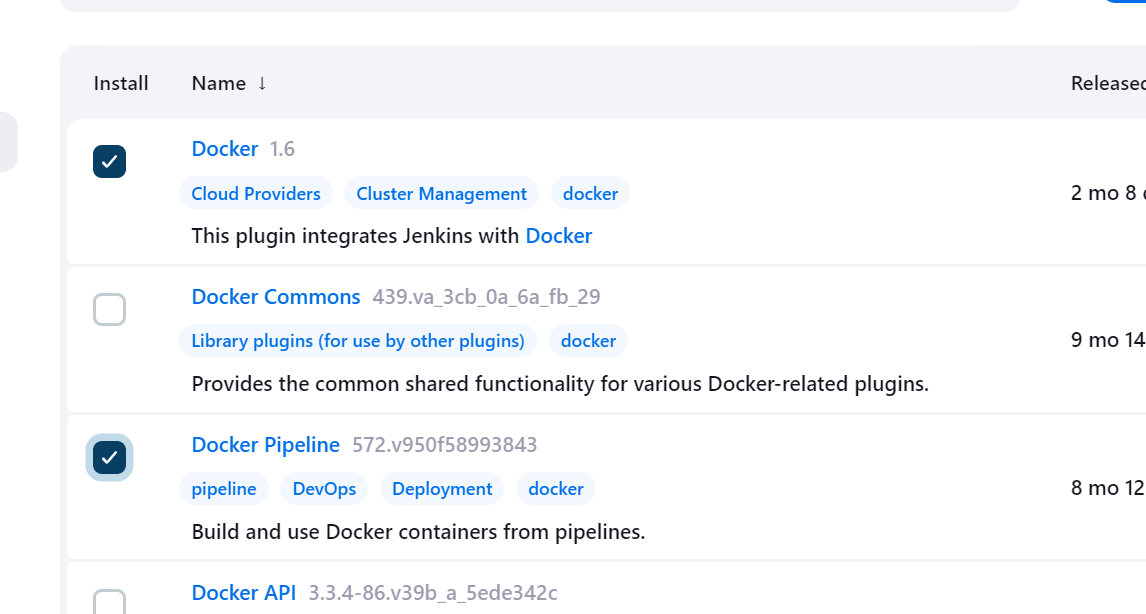
**Microservices Deployment On EKS With Jenkins**

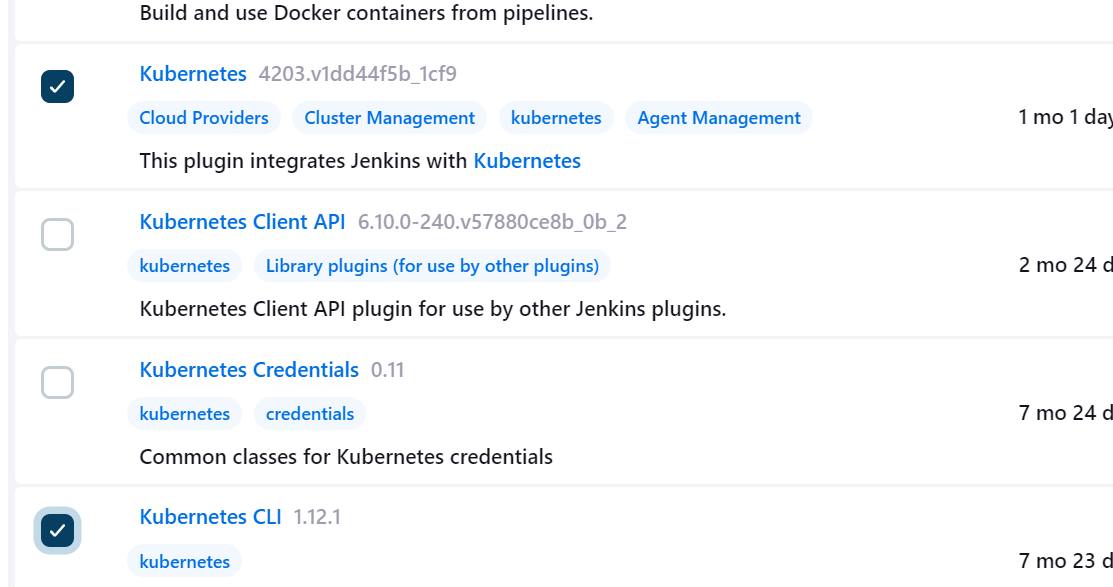
**Github Repo -** [**https://github.com/sudhanshuvlog/MicroservicesApp**](https://github.com/sudhanshuvlog/MicroservicesApp)

Create an ec2 instance with t2.medium and setup Jenkins

Use this command to launch jenkins container - **#docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts-jdk17**

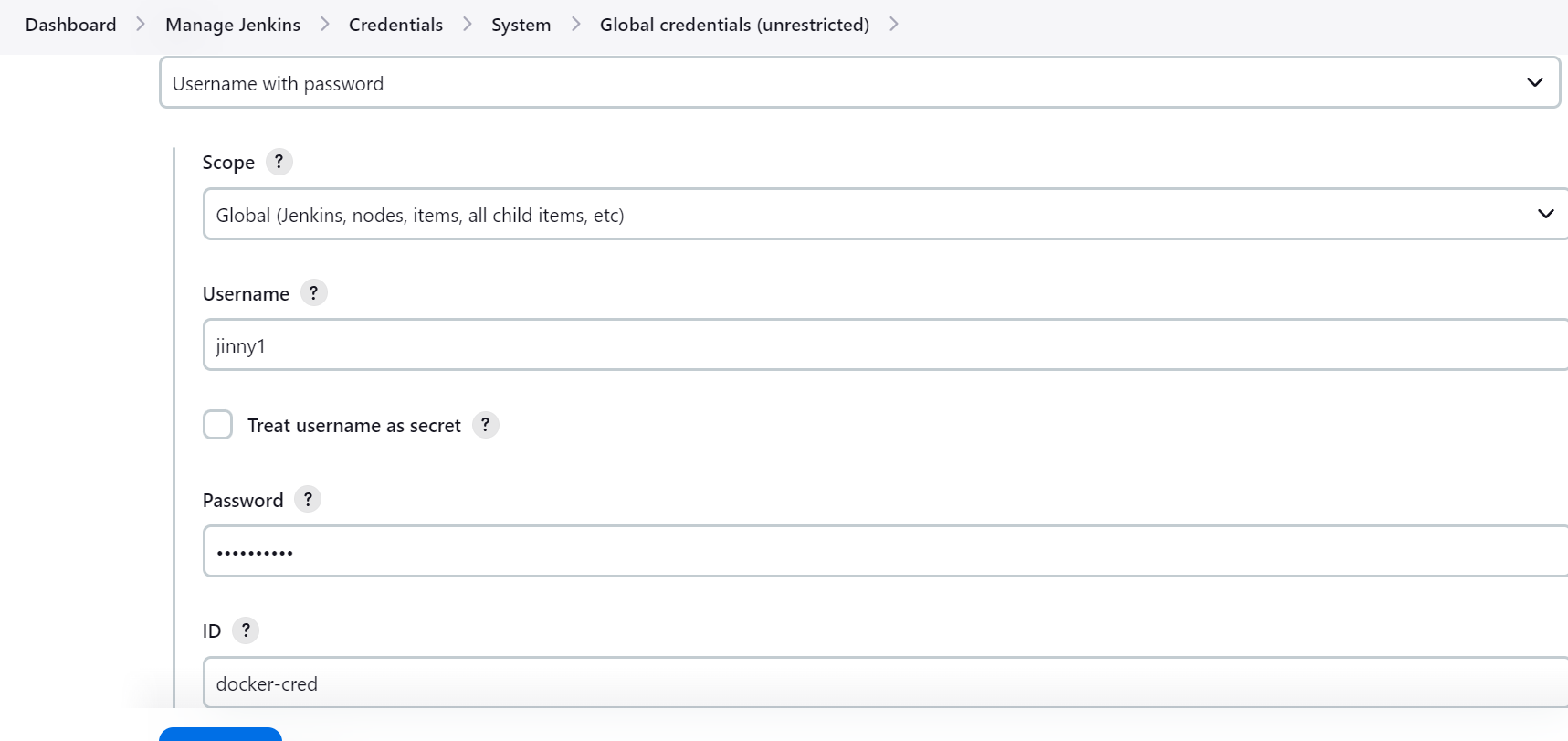
Now Once the Jenkins server is running, Install the below plugins





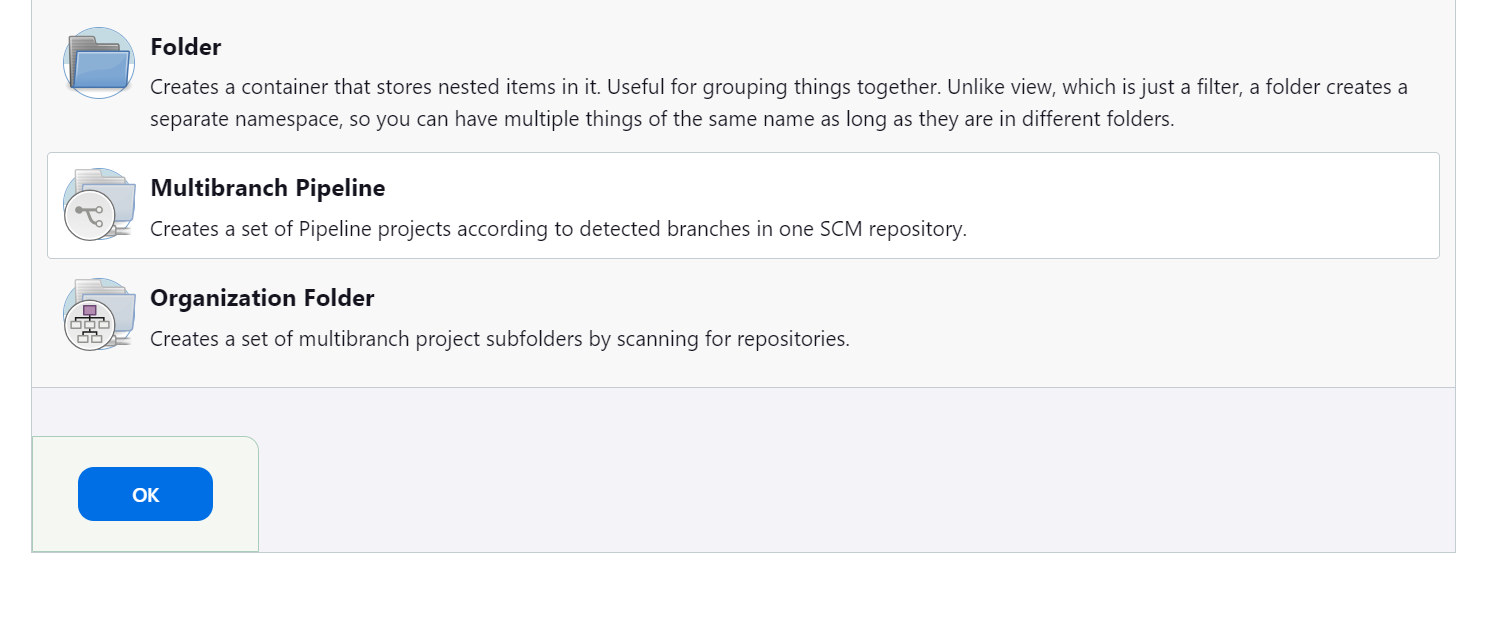
Once the plugin is installed, Please set the ec2 instance as a slave node and also Install git in the slave node, it will be required later.

Set docker hub credentials in Jenkins - Go to Manage Jenkins and Credentials



You can set git-cred as well for private repo, in the same way as shown above.

Now Create a MultiBranch Pipeline -

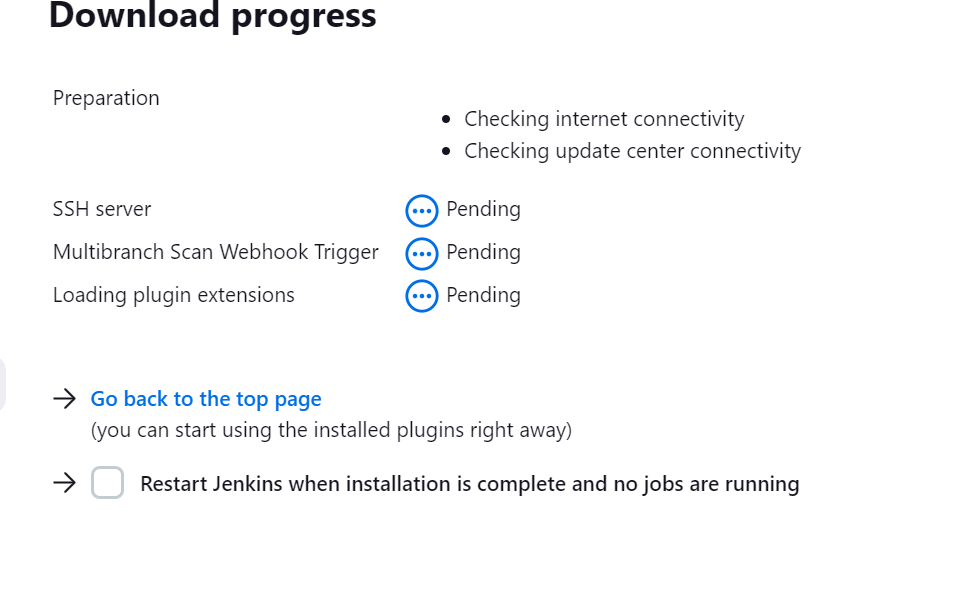


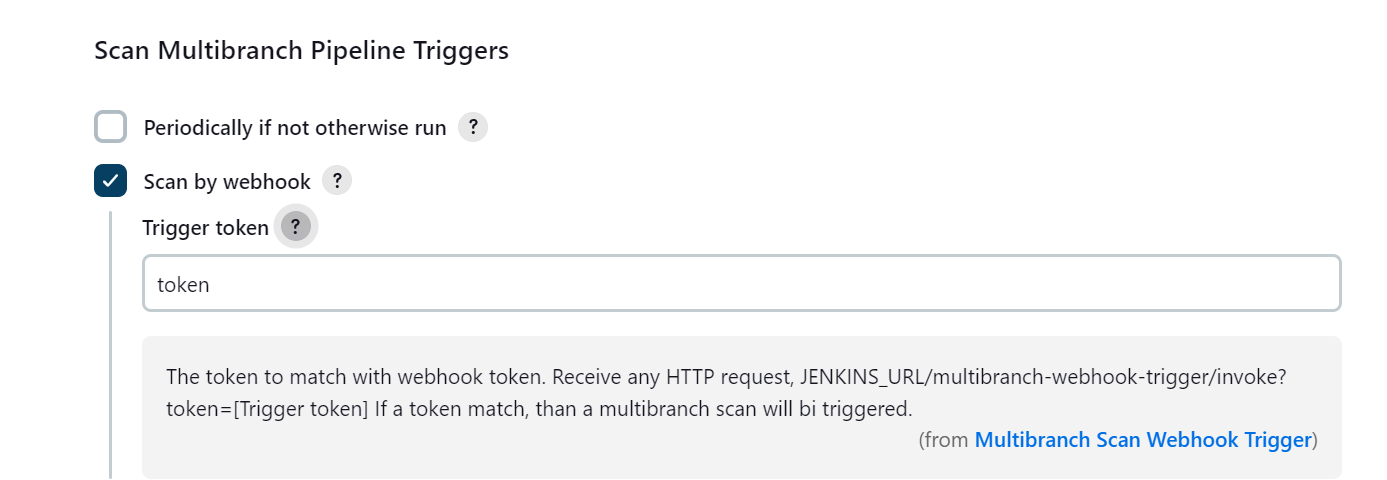
In MultiBranch Pipeline, Please set the git repo



In trigger you will not find a multibranch trigger hence go to the plugin and install this

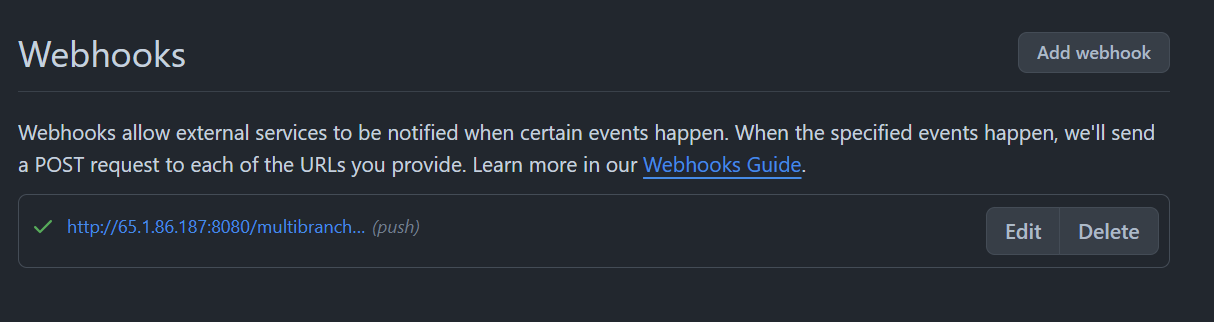
“MultiBranch Scan Webhook trigger”



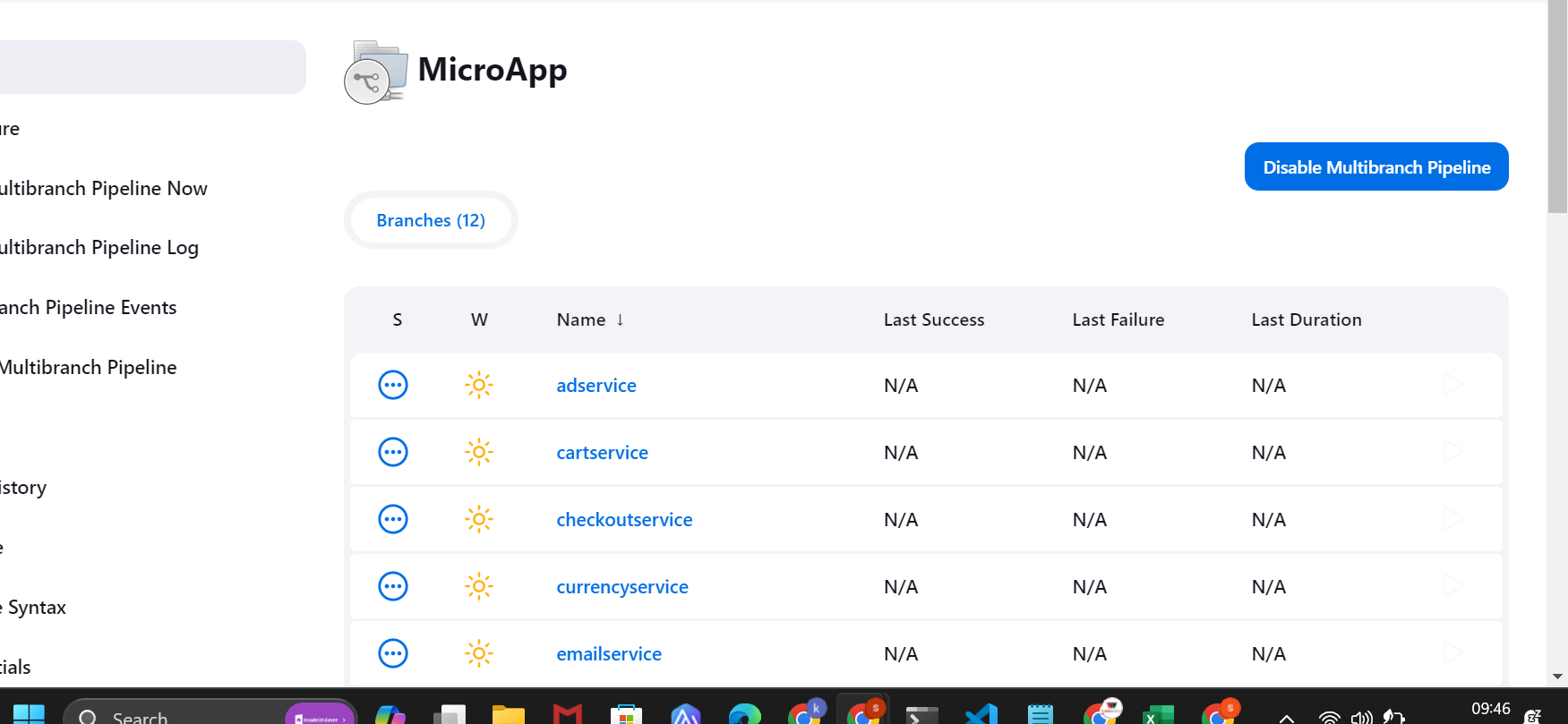


<http://65.1.86.187:8080/multibranch-webhook-trigger/invoke?token=token>

Now mention this token to the GitHub webhook



Once this webhook will be added, You will see all the 11 pipelines will be started



Automatic 11 pipelines are triggered



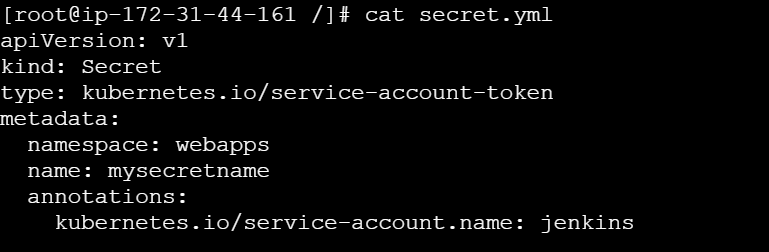
Now that all the microservices have been built successfully, So we can proceed with the deployment.

But we will have to authenticate Jenkins to be a k8s client so that it can connect to the EKS cluster and deploy the microservices.

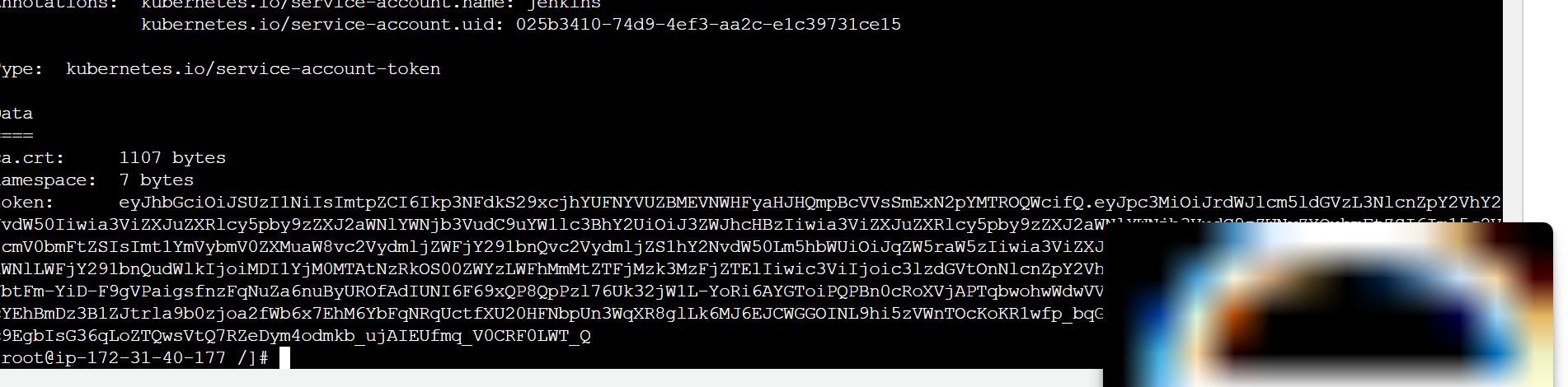
Follow the next steps from here - <https://github.com/sudhanshuvlog/MicroservicesApp/blob/Infra-Steps/setup-infra.md>

From the above document, you will find the service account, role configuration file.

You can create those resources with “kubectl apply” command

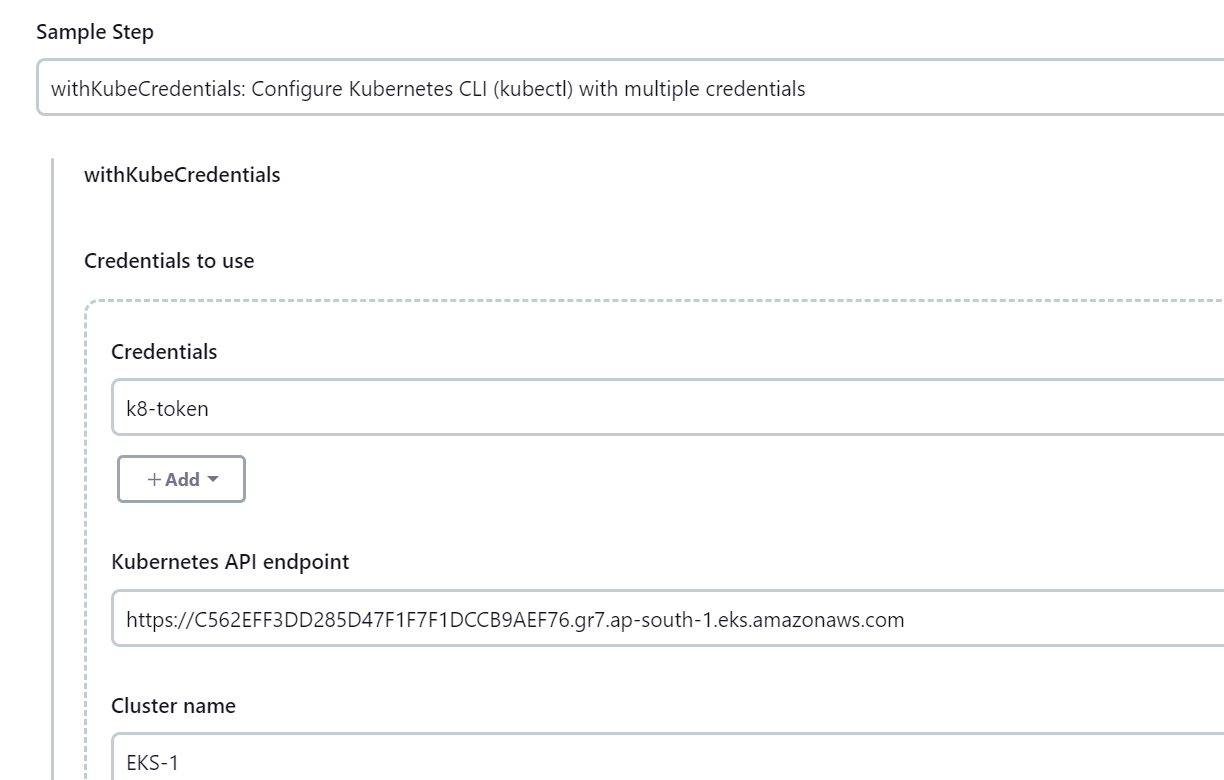


Once done, get the secret, as shown below





Now we can go to Jenkins and create the Groovy script, for authenticating to EKS cluster



Use this command to delete your entire cluster - **eksctl delete cluster --name EKS-1 --region ap-south-1**

Command History

1 docker run -p 8080:8080 -p 50000:50000 --restart=on-failure –name jenkins -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts-jdk17

2 docker run -it -p 8080:8080 -p 50000:50000 --restart=on-failure –-name jenkins -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts-jdk17

3 cls

4 clear

5 docker run -p 8080:8080 -p 50000:50000 --name jenkins --restart=on-failure -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts-jdk17

6 cd /

7 wget https://download.oracle.com/java/17/archive/jdk-17.0.10\_linux-x64\_bin.rpm

8 yum install jdk-17.0.10\_linux-x64\_bin.rpm -y

9 mkdir /slave

10 curl -sO http://15.206.66.20:8080/jnlpJars/agent.jar

11 java -jar agent.jar -url http://15.206.66.20:8080/ -secret 84c347988c2ee7da4c24da150a97e31821d51b535edecd904b205ed957330836 -name ec2 -workDir "/slave" &

12 yum install git -y

13 vi service-account.yml

14 vi role.yml

15 kubectl get pods

16 vi bind-role.yml

17 kubectl apply -f service-account.yml

18 kubectl create namespace webapps

19 kubectl apply -f service-account.yml

20 kubectl apply -f role.yml

21 kubectl apply -f bind-role.yml

22 vi secret.yml

23 kubectl apply -f secret.yml

24 kubectl get secret

25 kubectl get secret -n webapps

26 kubectl describe secret mysecretname -n webapps

27 clear

28 history