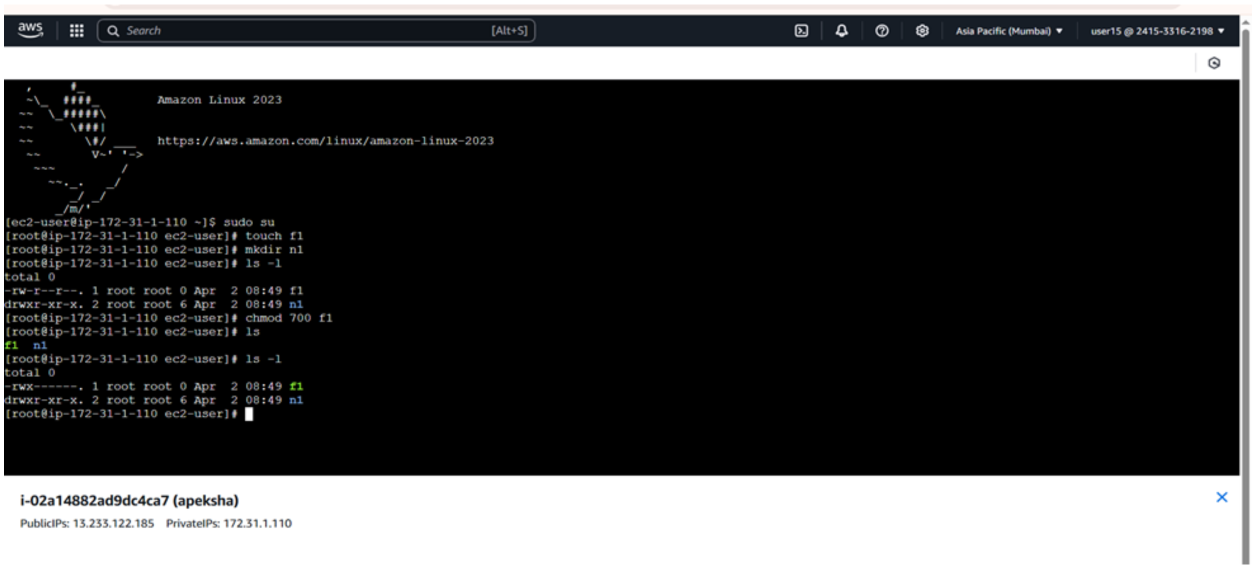
Day-5:

Step 1: Launch EC2 Instances (Master and Worker Node)

1. **Login to AWS Console**
   - Visit AWS Console and sign in using your AWS credentials.
2. **Launch EC2 Instances:**
   - Navigate to **EC2** → **Instances** → **Launch Instances**.
3. **Create the Master Node:**
   - Choose **Amazon Linux 2 AMI** or **Ubuntu**.
   - Select instance type **t2.medium** or higher.
   - Configure the instance to allow:
     - **SSH (port 22)**
     - **HTTP (port 80)**
     - **HTTPS (port 443)**
   - Launch the instance, and take note of the public IP address (e.g., 13.203.202.3).
4. **Create the Worker Node:**
   - Follow the same steps to launch a second instance as a worker node. Again, choose **Amazon Linux 2** or **Ubuntu**.
   - Configure security groups with the same ports open (SSH, HTTP, HTTPS).
5. **Connect to both instances:**
   - After both instances are launched, select each instance and use the **Connect** button to SSH into them via terminal. The public IP addresses will be used to connect to each node.



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-1-110 ~]$ sudo su
[root@ip-172-31-1-110 ec2-user]# touch f1
[root@ip-172-31-1-110 ec2-user]# mkdir n1
[root@ip-172-31-1-110 ec2-user]# ls -l
total 0
-rw-r--r--. 1 root root 0 Apr  2 08:49 f1
drwxr-xr-x. 2 root root 6 Apr  2 08:49 n1
[root@ip-172-31-1-110 ec2-user]# chmod 700 f1
[root@ip-172-31-1-110 ec2-user]# ls
f1  n1
[root@ip-172-31-1-110 ec2-user]# ls -l
total 0
-rwx------. 1 root root 0 Apr  2 08:49 f1
drwxr-xr-x. 2 root root 6 Apr  2 08:49 n1
[root@ip-172-31-1-110 ec2-user]#
```

i-02a14882ad9dc4ca7 (apeksha)
PublicIPs: 13.233.122.185 PrivateIPs: 172.31.1.110

Step 2: Configure EC2 Instances

On both the **Master Node** and **Worker Node**, execute the following commands to set up your environment.

On EC2 Instances (Master and Worker Nodes):

Get Root Access:
sudo su

Configure AWS CLI:
aws configure

1.Enter your **AWS Access Key**, **Secret Key**, **Region** (e.g., ap-south-1), and **Output format** (default json).

Update Kubernetes Cluster Configuration: To set up your **Master Node** and **Worker Node** for Kubernetes, run:
Commands in terminal:
aws eks update-kubeconfig --name cluster-1 --region ap-south-1

aws eks update-kubeconfig --name cluster-2 --region ap-south-1

Verify Node Setup: Run the following to verify that the nodes are correctly connected:
kubectl get nodes

This command will display a list of nodes in your Kubernetes cluster.

Step 3: Set Up Jenkins on EC2 Master Node

Install Jenkins on Master Node: Follow these steps to install Jenkins on the **Master Node**:

For Amazon Linux 2 (Master Node):

Commands:

```
sudo yum update -y

sudo amazon-linux-extras install java-openjdk11 -y

sudo yum install jenkins -y

sudo systemctl start jenkins

sudo systemctl enable jenkins
```

For Ubuntu (Master Node):

```
sudo apt update

sudo apt install openjdk-11-jdk -y

sudo wget -q -O - https://pkg.jenkins.io/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb http://pkg.jenkins.io/debian/ stable main > /etc/apt/sources.list.d/jenkins.list'

sudo apt update

sudo apt install jenkins -y

sudo systemctl start jenkins

sudo systemctl enable jenkins
```

1. Open Jenkins Web Interface:
After Jenkins is installed and running, navigate to your **Master Node**'s public IP in a browser. For example:
http://<Master_Node_Public_IP>:8080
sudo cat /var/lib/jenkins/secrets/initialAdminPassword →(To retrieve)

1. **Complete Jenkins Setup:**
   Follow the Jenkins setup wizard to install recommended plugins and create the admin user.
2. **Install Kubernetes Plugin for Jenkins:**

   From the Jenkins dashboard, go to **Manage Jenkins** → **Manage Plugins**.

   Search for **Kubernetes** plugin, install it, and restart Jenkins.

Step 4: Configure Worker Node for Jenkins Jobs

On the **Worker Node**, you need to configure it as a **Kubernetes Node** where Jenkins can run jobs.

1. **Install Docker and Kubernetes on Worker Node:**
   - Follow similar steps as the **Master Node** to install Docker and Kubernetes on the **Worker Node**.
2. **Configure Worker Node in Jenkins:**
   - Go to the **Jenkins Dashboard** → **Manage Jenkins** → **Configure System**.
   - Under **Cloud** section, add **Kubernetes Cloud**.
   - Enter the **Kubernetes URL** (e.g., http://<Master_Node_Public_IP>:8080).
   - Configure the **Kubernetes plugin** with the necessary credentials and settings to allow the worker node to run Jenkins jobs.

Step 5: Set Up Jenkins Jobs and Kubernetes Workloads

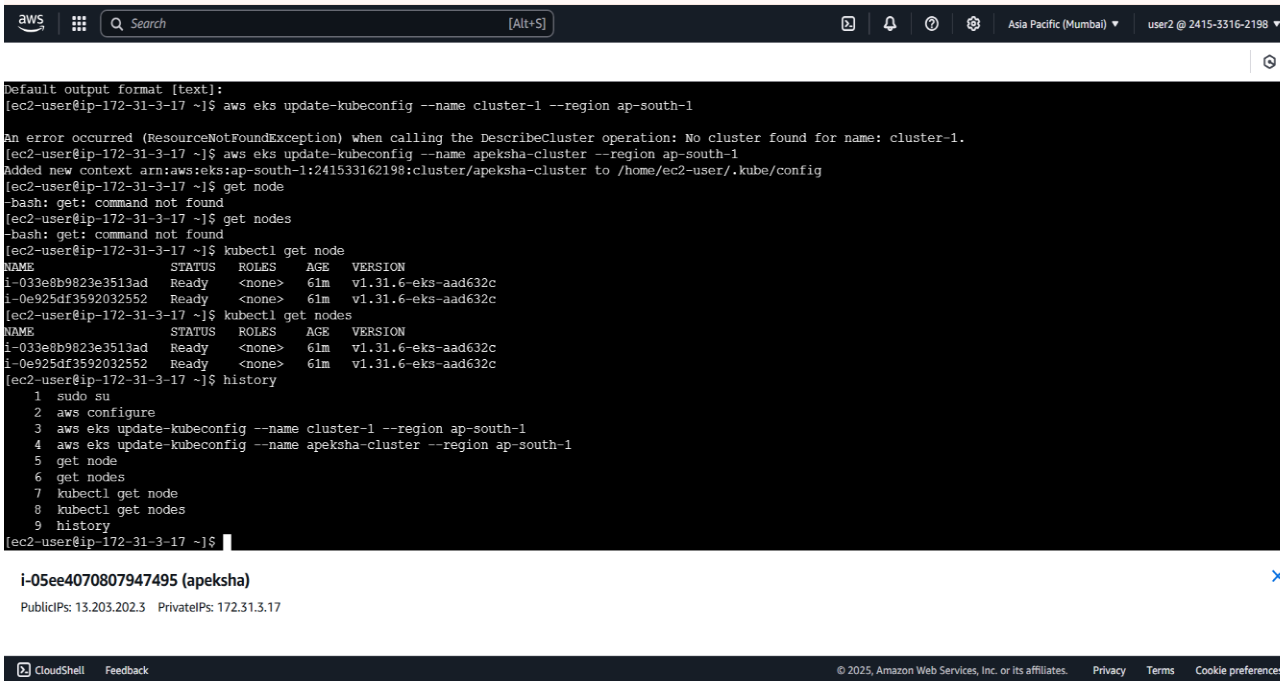
1. **Create Jenkins Pipeline Jobs:**
   - Create Jenkins jobs that will deploy containers, manage Kubernetes pods, and execute tasks on the Worker Node.
2. **Configure Kubernetes Executor:**
   - For Jenkins to run tasks on the Kubernetes Worker Node, configure the executor as **Kubernetes Pod**. This allows Jenkins to spawn Kubernetes pods for job execution.
3. **Test the Setup:**
   - Run a Jenkins job, and check if it is executed on the Worker Node by viewing the job logs and Kubernetes dashboard.

Step 6: Access Jenkins Web Interface

After everything is set up, you can access Jenkins using the **Master Node's public IP**:

http://<Master_Node_Public_IP>:8080

- Enter your credentials to log in.
- Jenkins will now manage and distribute tasks to your Worker Node based on the Kubernetes cluster configuration.



```
Default output format [text]:
[ec2-user@ip-172-31-3-17 ~]$ aws eks update-kubeconfig --name cluster-1 --region ap-south-1

An error occurred (ResourceNotFoundException) when calling the DescribeCluster operation: No cluster found for name: cluster-1.
[ec2-user@ip-172-31-3-17 ~]$ aws eks update-kubeconfig --name apeksha-cluster --region ap-south-1
Added new context arn:aws:eks:ap-south-1:241533162198:cluster/apeksha-cluster to /home/ec2-user/.kube/config
[ec2-user@ip-172-31-3-17 ~]$ get node
-bash: get: command not found
[ec2-user@ip-172-31-3-17 ~]$ get nodes
-bash: get: command not found
[ec2-user@ip-172-31-3-17 ~]$ kubectl get node
NAME                  STATUS   ROLES    AGE   VERSION
i-033e8b9823e3513ad   Ready    <none>   61m   v1.31.6-eks-aad632c
i-0e925df3592032552   Ready    <none>   61m   v1.31.6-eks-aad632c
[ec2-user@ip-172-31-3-17 ~]$ kubectl get nodes
NAME                  STATUS   ROLES    AGE   VERSION
i-033e8b9823e3513ad   Ready    <none>   61m   v1.31.6-eks-aad632c
i-0e925df3592032552   Ready    <none>   61m   v1.31.6-eks-aad632c
[ec2-user@ip-172-31-3-17 ~]$ history
    1  sudo su
    2  aws configure
    3  aws eks update-kubeconfig --name cluster-1 --region ap-south-1
    4  aws eks update-kubeconfig --name apeksha-cluster --region ap-south-1
    5  get node
    6  get nodes
    7  kubectl get node
    8  kubectl get nodes
    9  history
[ec2-user@ip-172-31-3-17 ~]$
```

i-05ee4070807947495 (apeksha)

PublicIPs: 13.203.202.3 PrivateIPs: 172.31.3.17

Step 2: Commands

sudo su →To get into Ec2 rootaws configure→for the aws configuration.

aws eks update-kubeconfig —name cluster-1 –region ap-south-1—>To update kubernet region

Aws eks update-kubeconfig —name cluster-2 –region ap-south-1

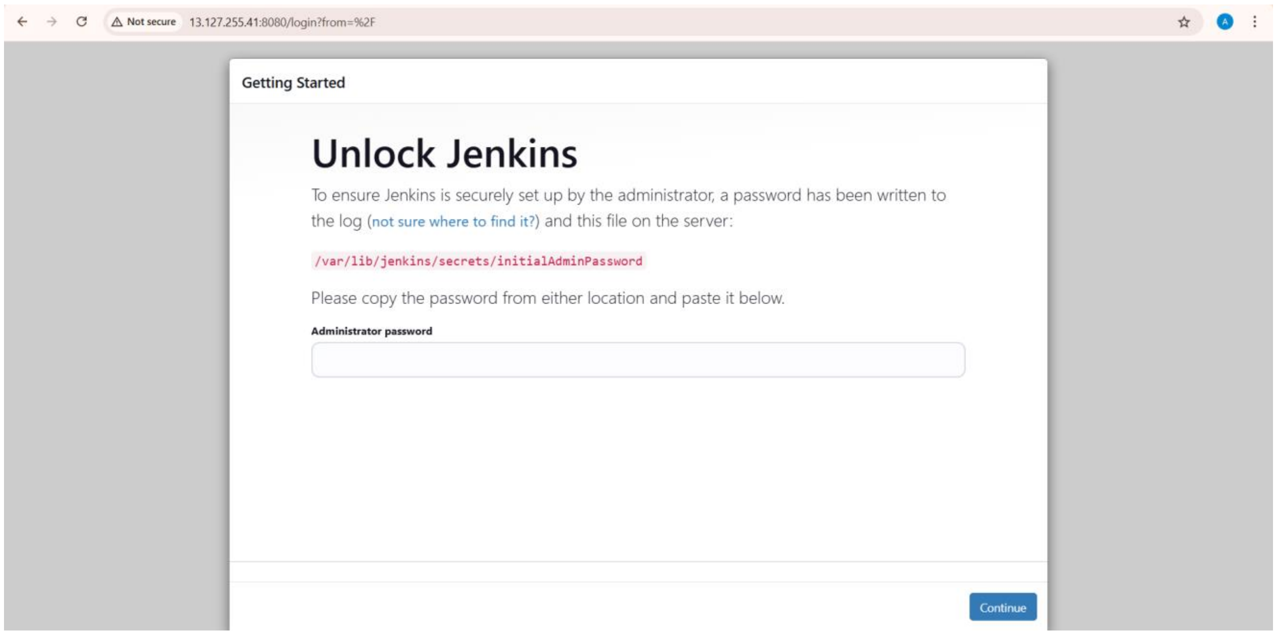
get node

get nodes

Kubectl get node

Kubectl get nodes

After getting the output with those image commands then paste that public ip address(like 13.203.202.3) with in web and we should get the image as below:



Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

Step 3: 1. EC2 Master Node: will run Jenkins and manage the Kubernete cluster.

2. EC2 Worker Nodes: will run Jenkins jobs or Kubernetes workloads.