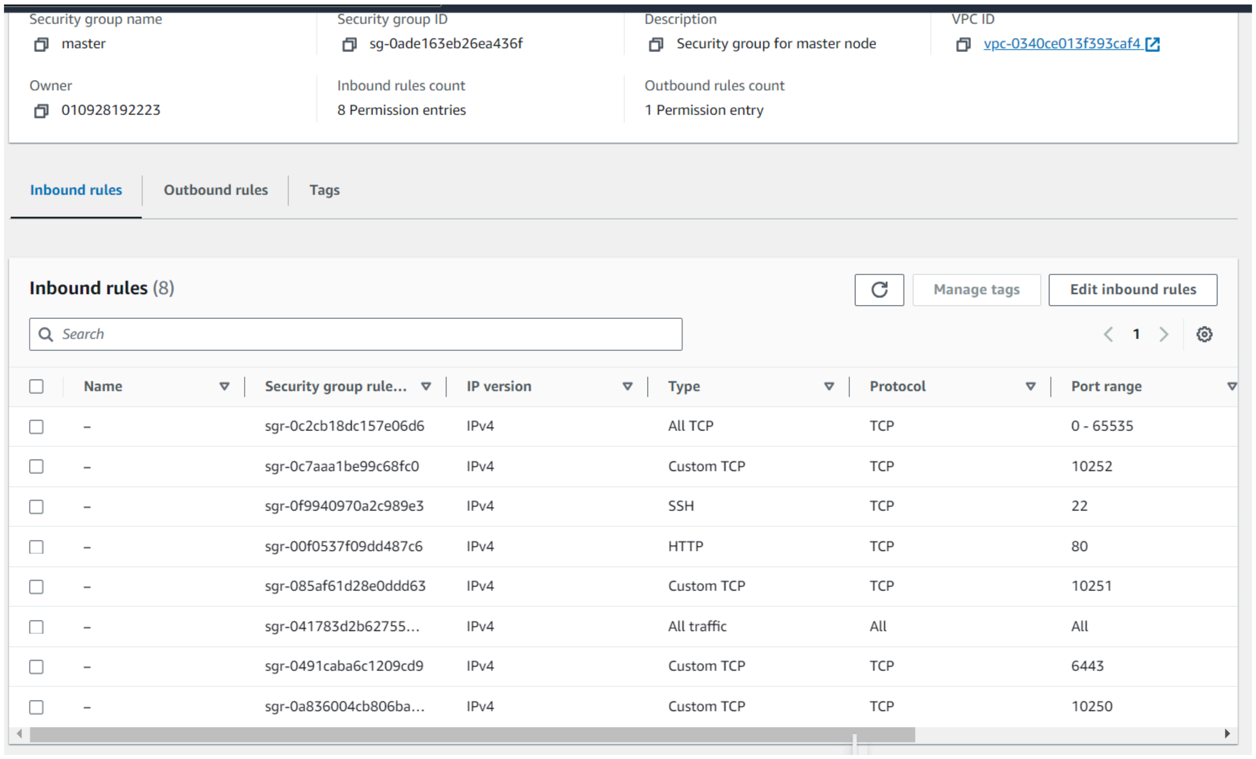
Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Prerequisites :
Create 2 Security Groups for Master and Nodes and add the following rules inbound rules in those Groups.

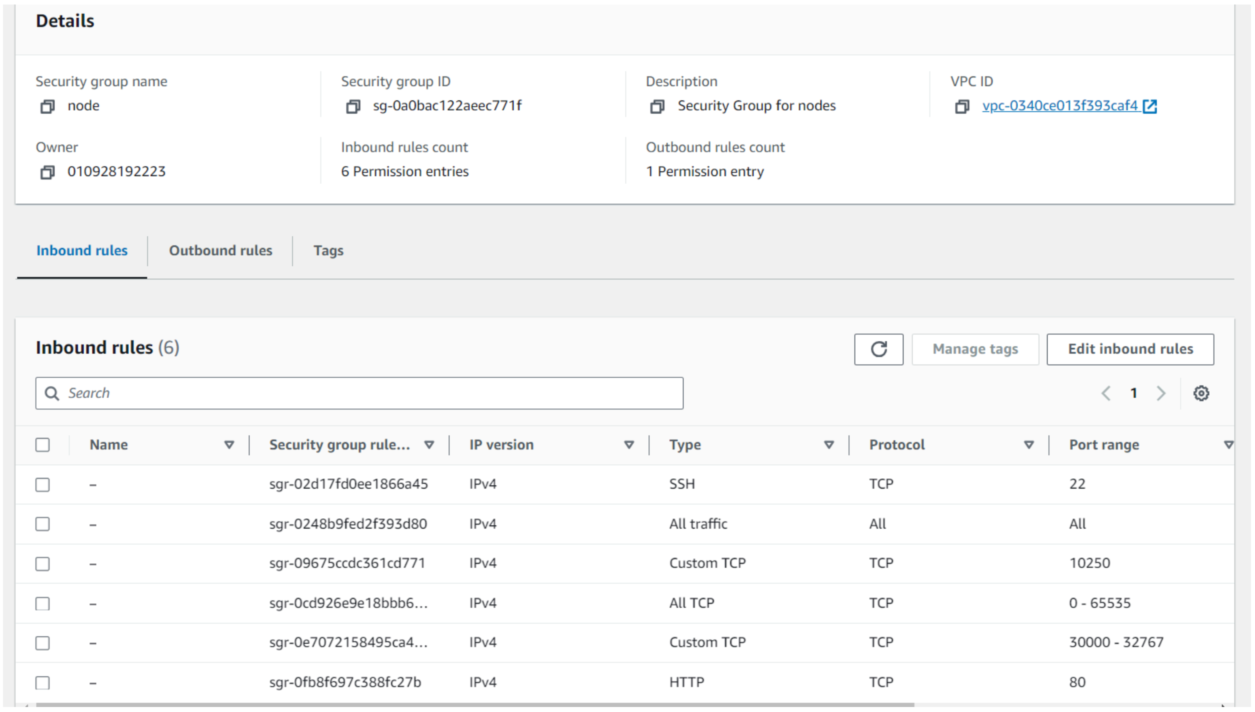


Security group name	Security group ID	Description	VPC ID
master	sg-0ade163eb26ea436f	Security group for master node	vpc-0340ce013f393caf4

Owner	Inbound rules count	Outbound rules count
010928192223	8 Permission entries	1 Permission entry

Inbound rules (8)

Name	Security group rule...	IP version	Type	Protocol	Port range
–	sgr-0c2cb18dc157e06d6	IPv4	All TCP	TCP	0 - 65535
–	sgr-0c7aaa1be99c68fc0	IPv4	Custom TCP	TCP	10252
–	sgr-0f9940970a2c989e3	IPv4	SSH	TCP	22
–	sgr-00f0537f09dd487c6	IPv4	HTTP	TCP	80
–	sgr-085af61d28e0ddd63	IPv4	Custom TCP	TCP	10251
–	sgr-041783d2b62755...	IPv4	All traffic	All	All
–	sgr-0491caba6c1209cd9	IPv4	Custom TCP	TCP	6443
–	sgr-0a836004cb806ba...	IPv4	Custom TCP	TCP	10250



Details

Security group name	Security group ID	Description	VPC ID
node	sg-0a0bac122aeec771f	Security Group for nodes	vpc-0340ce013f393caf4

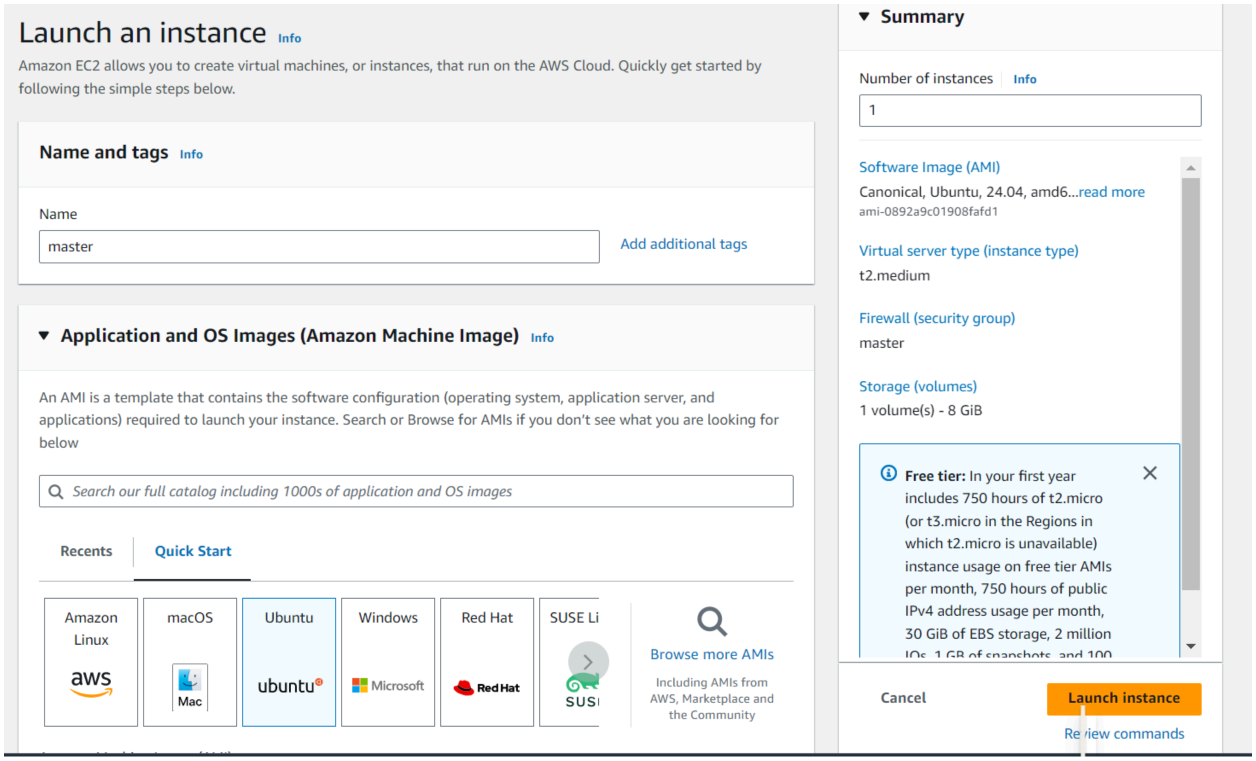
Owner	Inbound rules count	Outbound rules count
010928192223	6 Permission entries	1 Permission entry

Inbound rules (6)

Name	Security group rule...	IP version	Type	Protocol	Port range
–	sgr-02d17fd0ee1866a45	IPv4	SSH	TCP	22
–	sgr-0248b9fed2f393d80	IPv4	All traffic	All	All
–	sgr-09675ccdc361cd771	IPv4	Custom TCP	TCP	10250
–	sgr-0cd926e9e18bbb6...	IPv4	All TCP	TCP	0 - 65535
–	sgr-0e7072158495ca4...	IPv4	Custom TCP	TCP	30000 - 32767
–	sgr-0fb8f697c388fc27b	IPv4	HTTP	TCP	80

1. Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances. Select Ubuntu as AMI and t2.medium as Instance Type and create a key of type RSA with .pem extension and move the downloaded key to the new folder.We can use 3 Different keys or 1 common key also.

Master:



Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: master

Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Li | Browse more AMIs (Including AMIs from AWS, Marketplace and the Community)

Summary

Number of instances Info: 1

Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more ami-0892a9c01908fafd1

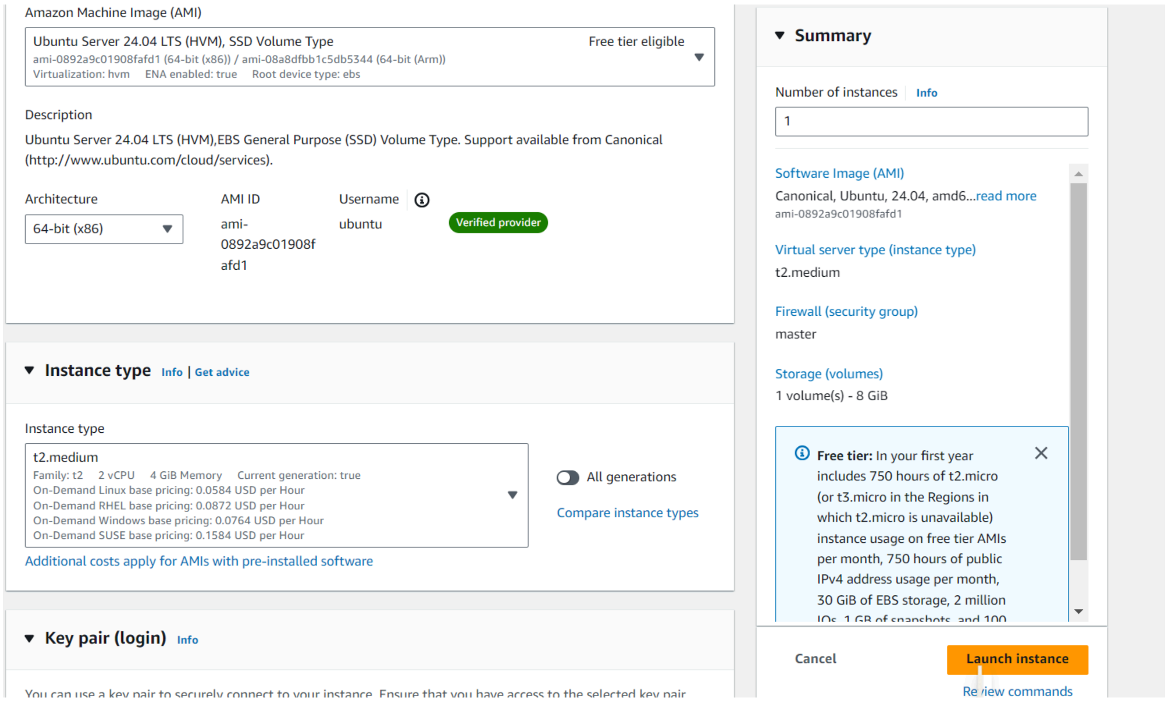
Virtual server type (instance type): t2.medium

Firewall (security group): master

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100

Cancel | Launch instance | Review commands



Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0892a9c01908fafd1 (64-bit (x86)) / ami-08a8dfbb1c5db5344 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Architecture	AMI ID	Username	
64-bit (x86)	ami-0892a9c01908fafd1	ubuntu	Verified provider

Instance type Info | Get advice

Instance type

t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0584 USD per Hour
On-Demand RHEL base pricing: 0.0872 USD per Hour
On-Demand Windows base pricing: 0.0764 USD per Hour
On-Demand SUSE base pricing: 0.1584 USD per Hour

All generations | Compare instance types

Additional costs apply for AMIs with pre-installed software

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair

Summary

Number of instances Info: 1

Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more ami-0892a9c01908fafd1

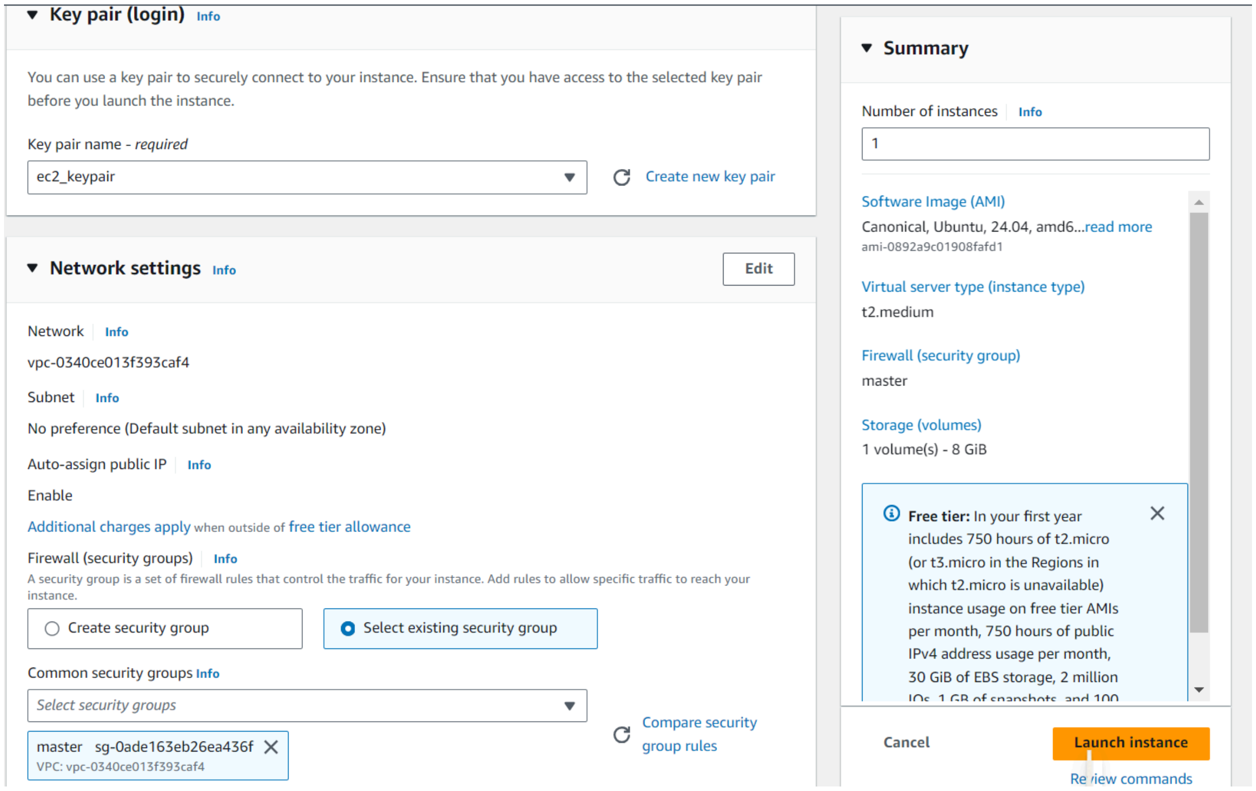
Virtual server type (instance type): t2.medium

Firewall (security group): master

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100

Cancel | Launch instance | Review commands


Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

ec2_keypair

Create new key pair

Network settings Info

Edit

Network Info

vpc-0340ce013f393caf4

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group [✓ Select existing security group]

Common security groups Info

Select security groups

master sg-0ade163eb26ea436f

VPC: vpc-0340ce013f393caf4

Compare security group rules

Summary

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 24.04, amd6...read more

ami-0892a9c01908fafd1

Virtual server type (instance type)

t2.medium

Firewall (security group)

master

Storage (volumes)

1 volume(s) - 8 GiB

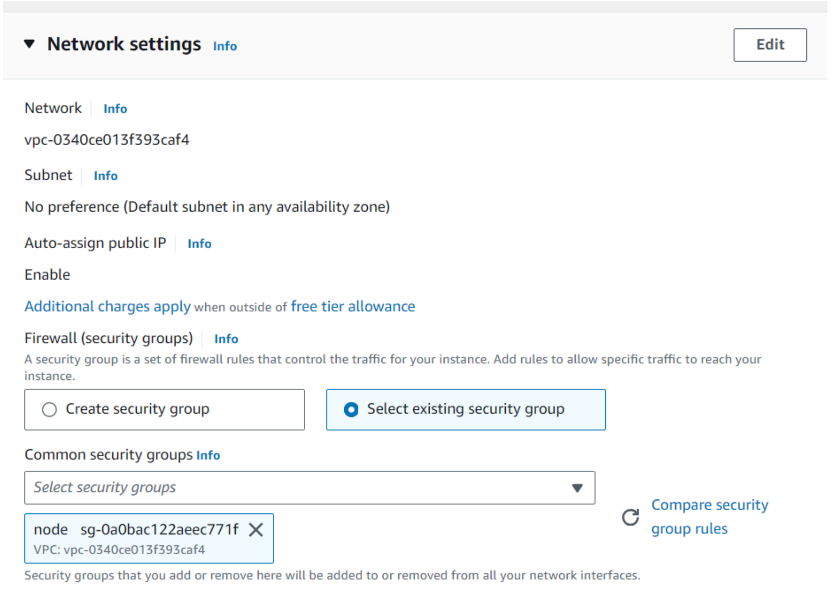
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million

Cancel

Launch instance

Review commands

Do Same for 2 Nodes and use security groups of Node for that.


Network settings Info

Edit

Network Info

vpc-0340ce013f393caf4

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group [✓ Select existing security group]

Common security groups Info

Select security groups

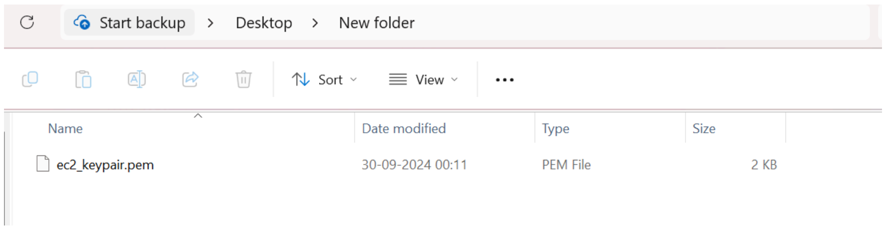
node sg-0a0bac122aeec771f

VPC: vpc-0340ce013f393caf4

Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

2. After creating the instances click on Connect & connect all 3 instances and navigate to SSH Client.



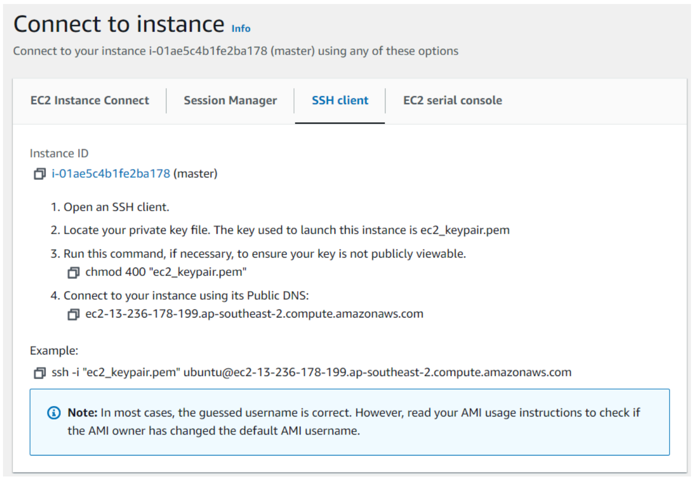
Start backup > Desktop > New folder

Sort View

Name	Date modified	Type	Size
ec2_keypair.pem	30-09-2024 00:11	PEM File	2 KB

3. Now open the folder in the terminal 3 times for Master, Node1& Node 2 where our .pem key is stored. Then execute the ssh command.
   For example: ssh -i "ec2_keypair.pem" ubuntu@ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com

   Master:



Connect to instance Info

Connect to your instance i-01ae5c4b1fe2ba178 (master) using any of these options

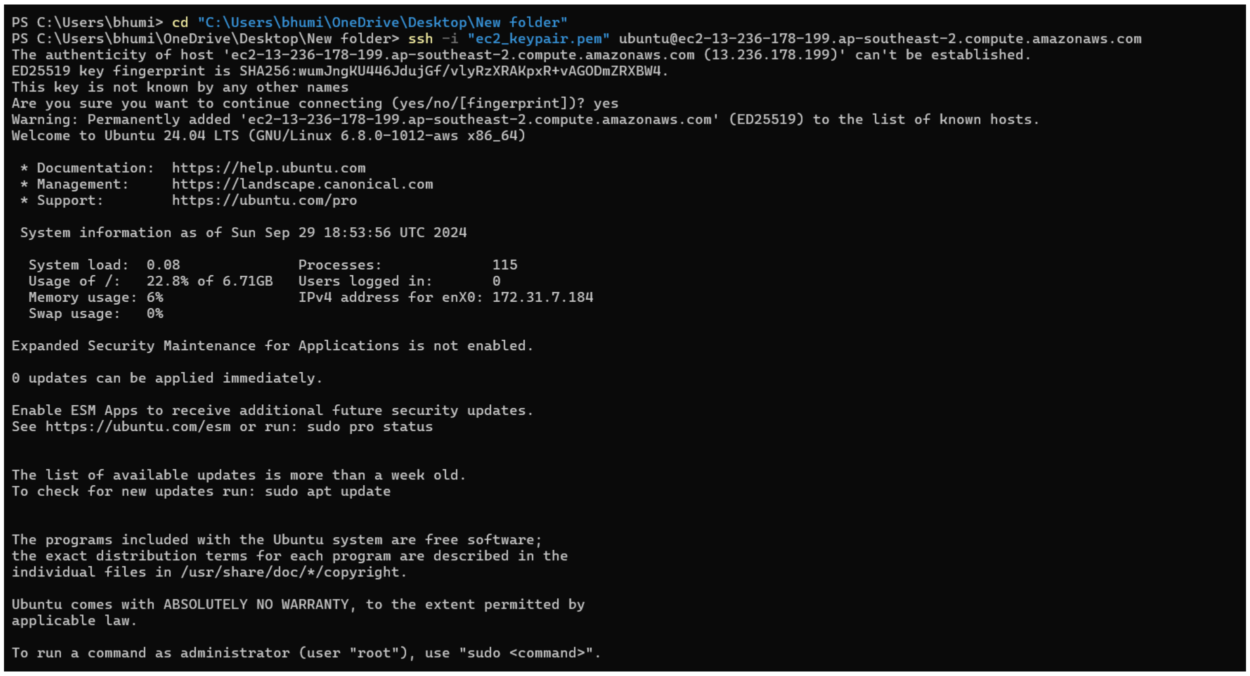
EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID
i-01ae5c4b1fe2ba178 (master)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ec2_keypair.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "ec2_keypair.pem"
4. Connect to your instance using its Public DNS:
   ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com

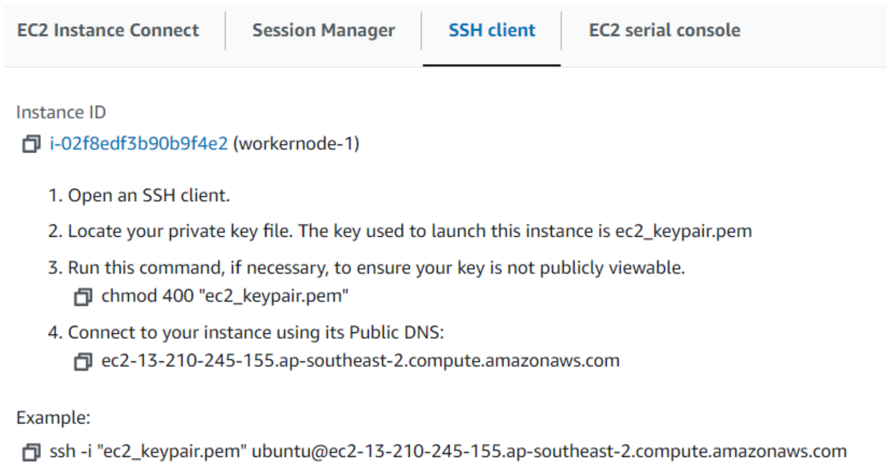
Example:
ssh -i "ec2_keypair.pem" ubuntu@ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.



```
PS C:\Users\bhumi> cd "C:\Users\bhumi\OneDrive\Desktop\New folder"
PS C:\Users\bhumi\OneDrive\Desktop\New folder> ssh -i "ec2_keypair.pem" ubuntu@ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com
The authenticity of host 'ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com (13.236.178.199)' can't be established.
ED25519 key fingerprint is SHA256:wumJngKU446JdujGf/vlyRzXRAKpxR+vAGODmZRXBW4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sun Sep 29 18:53:56 UTC 2024

  System load:  0.08               Processes:             115
  Usage of /:   22.8% of 6.71GB    Users logged in:       0
  Memory usage: 6%                 IPv4 address for enX0: 172.31.7.184
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
```

Node 1:



EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

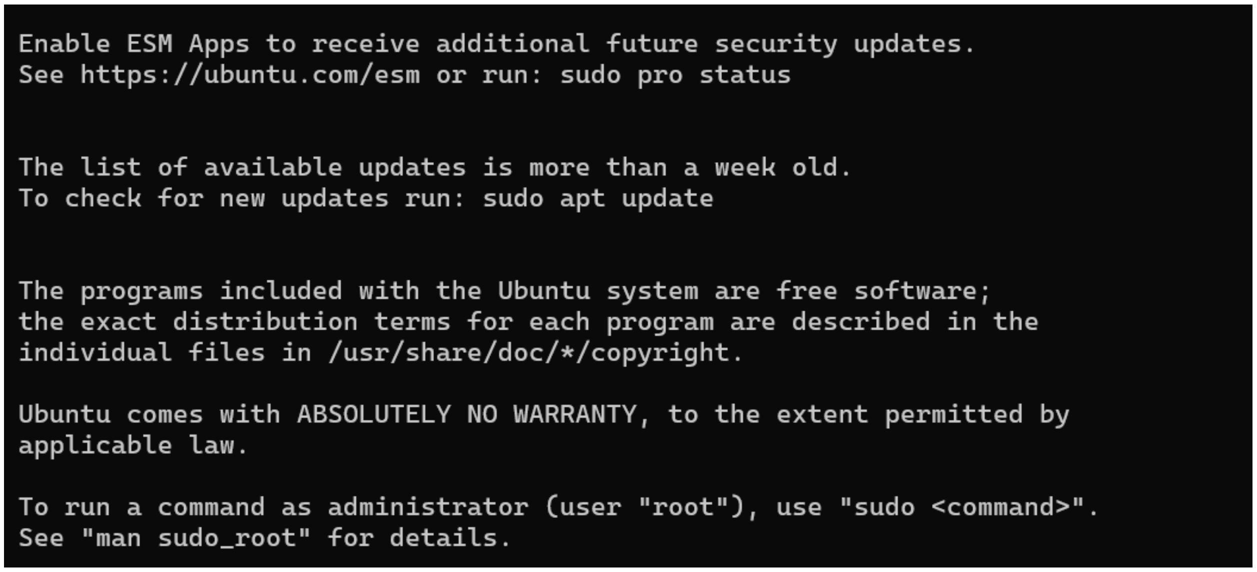
Instance ID

i-02f8edf3b90b9f4e2 (workernode-1)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ec2_keypair.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "ec2_keypair.pem"
4. Connect to your instance using its Public DNS:
   ec2-13-210-245-155.ap-southeast-2.compute.amazonaws.com

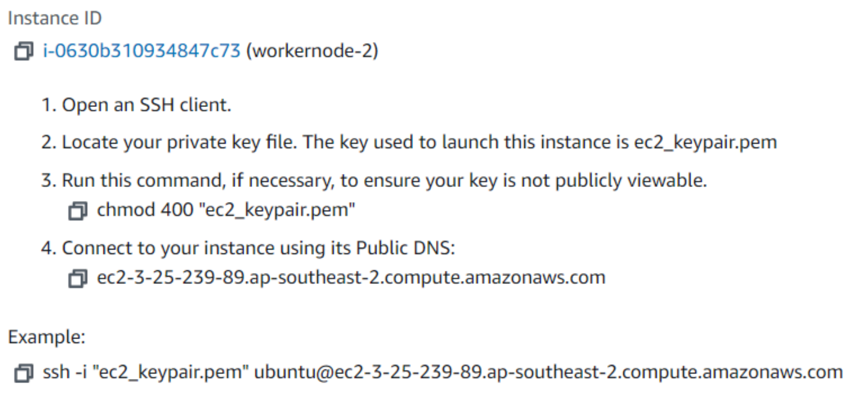
Example:

ssh -i "ec2_keypair.pem" ubuntu@ec2-13-210-245-155.ap-southeast-2.compute.amazonaws.com



```
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Node 2:



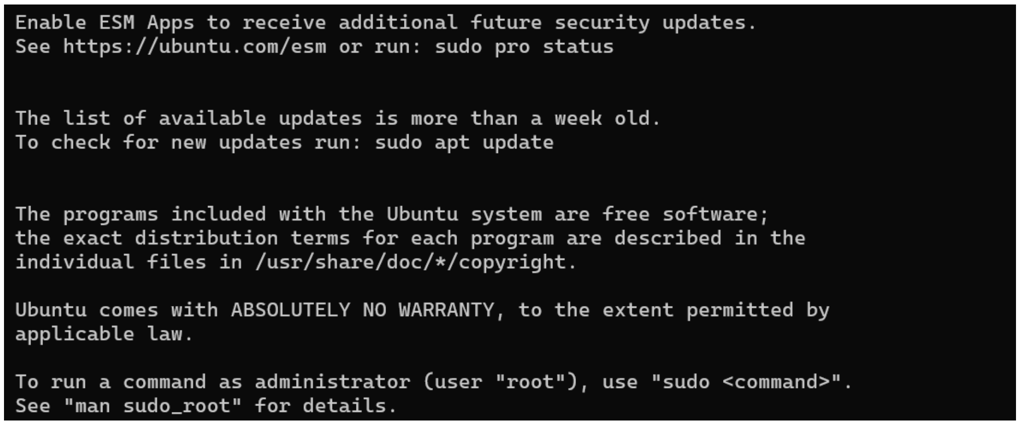
Instance ID

i-0630b310934847c73 (workernode-2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ec2_keypair.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "ec2_keypair.pem"
4. Connect to your instance using its Public DNS:
   ec2-3-25-239-89.ap-southeast-2.compute.amazonaws.com

Example:

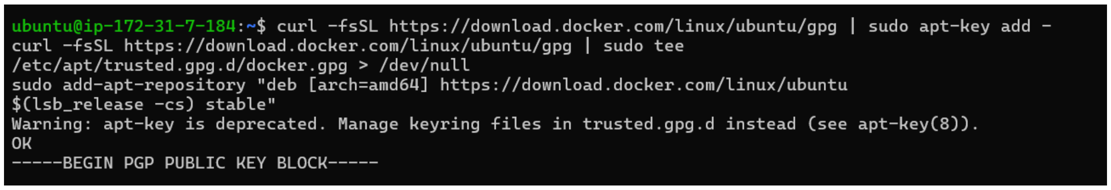
ssh -i "ec2_keypair.pem" ubuntu@ec2-3-25-239-89.ap-southeast-2.compute.amazonaws.com



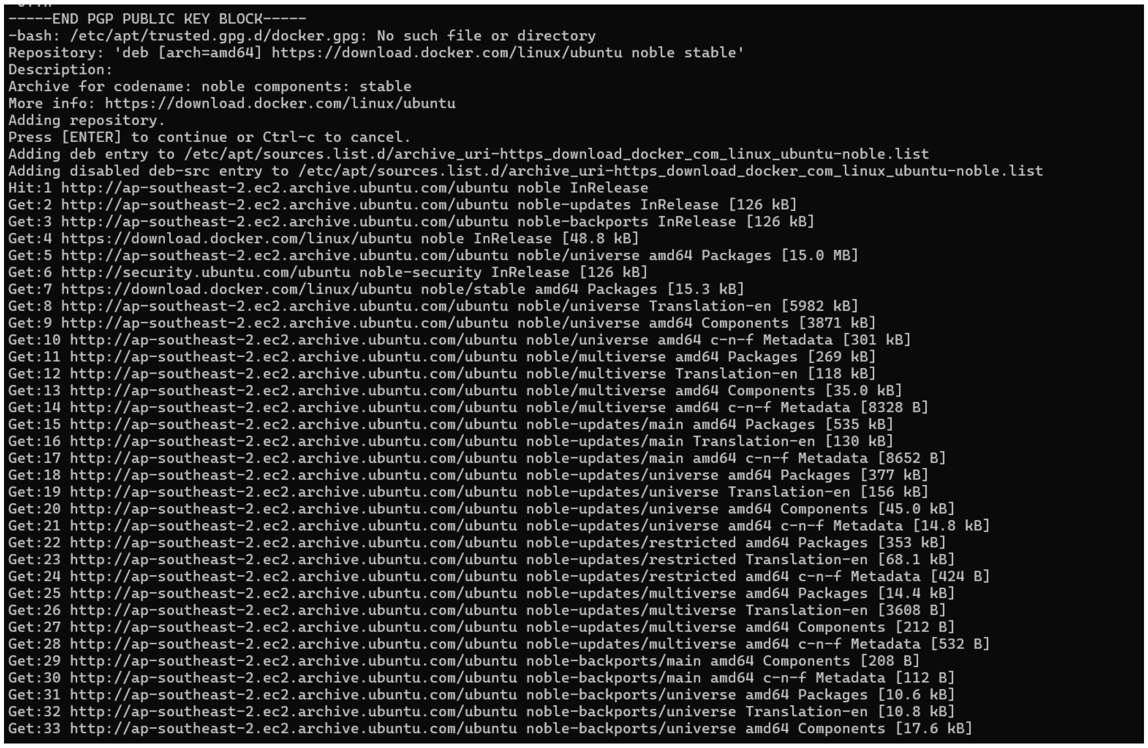
```
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Thus the connection is successful.

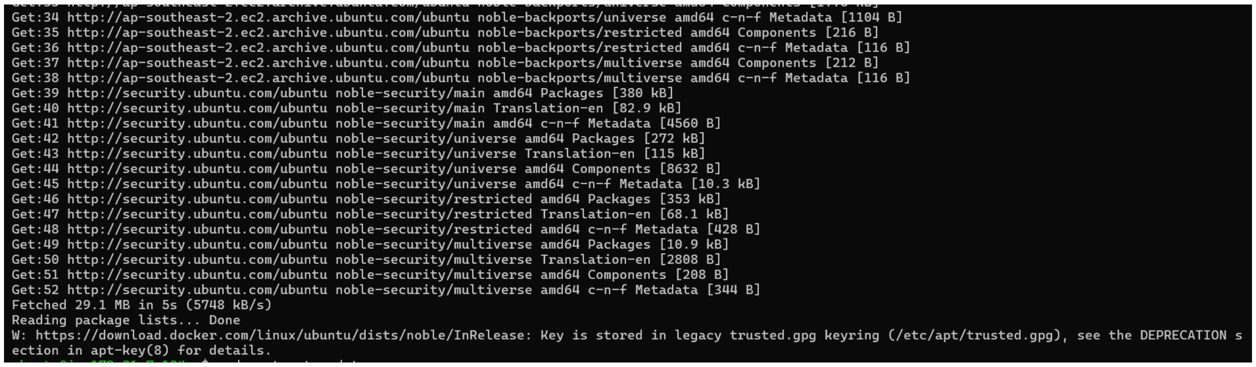
4. Run on Master,Node 1,and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.
   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
   sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable”



```
ubuntu@ip-172-31-7-184:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

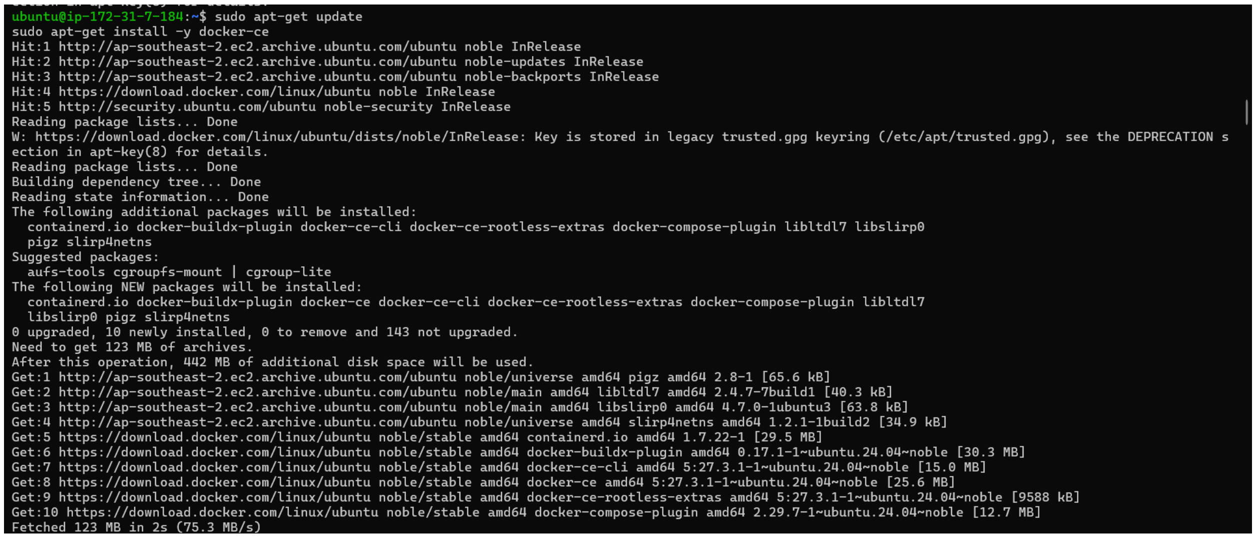


```
-----END PGP PUBLIC KEY BLOCK-----
-bash: /etc/apt/trusted.gpg.d/docker.gpg: No such file or directory
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:11 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:12 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:13 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:14 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:15 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:16 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [130 kB]
Get:17 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8652 B]
Get:18 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [377 kB]
Get:19 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [156 kB]
Get:20 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.8 kB]
Get:22 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [353 kB]
Get:23 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [68.1 kB]
Get:24 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:25 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:26 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:27 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:28 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:29 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:32 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:33 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
```

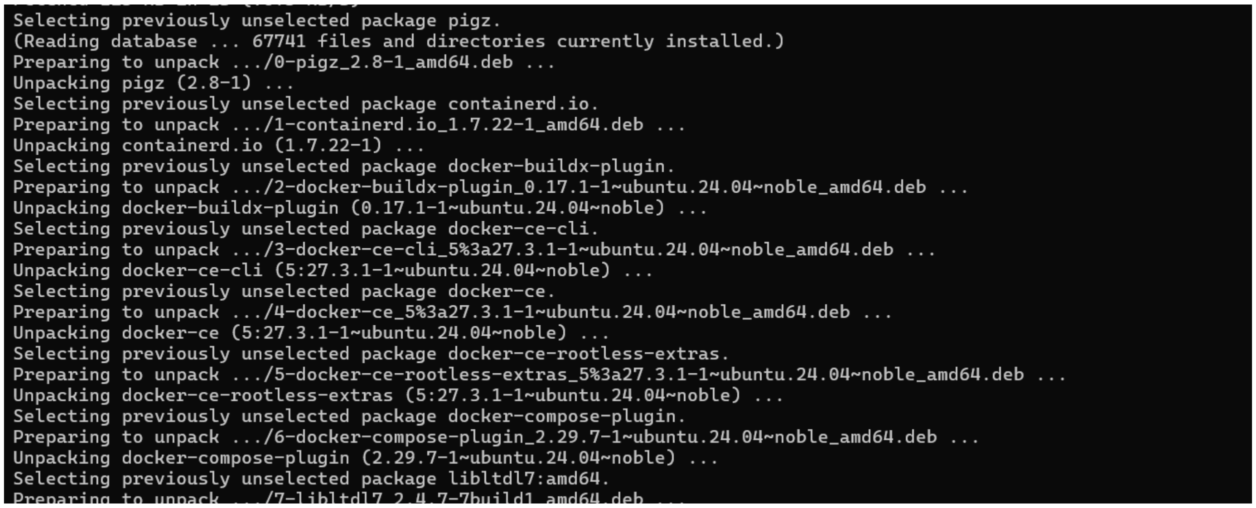


```
Get:34 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 5s (5748 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
```

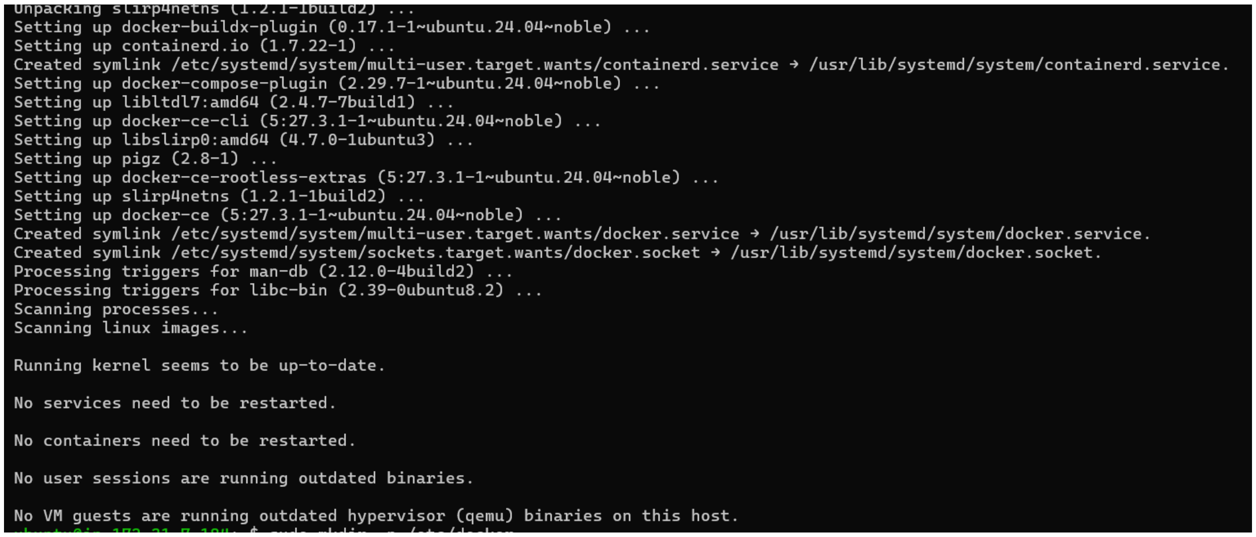
sudo apt-get update
sudo apt-get install -y docker-ce



```
ubuntu@ip-172-31-7-184:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 143 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.17.1-1~ubuntu.24.04~noble [30.3 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:27.3.1-1~ubuntu.24.04~noble [15.0 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:27.3.1-1~ubuntu.24.04~noble [25.6 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:27.3.1-1~ubuntu.24.04~noble [9588 kB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.29.7-1~ubuntu.24.04~noble [12.7 MB]
Fetched 123 MB in 2s (75.3 MB/s)
```

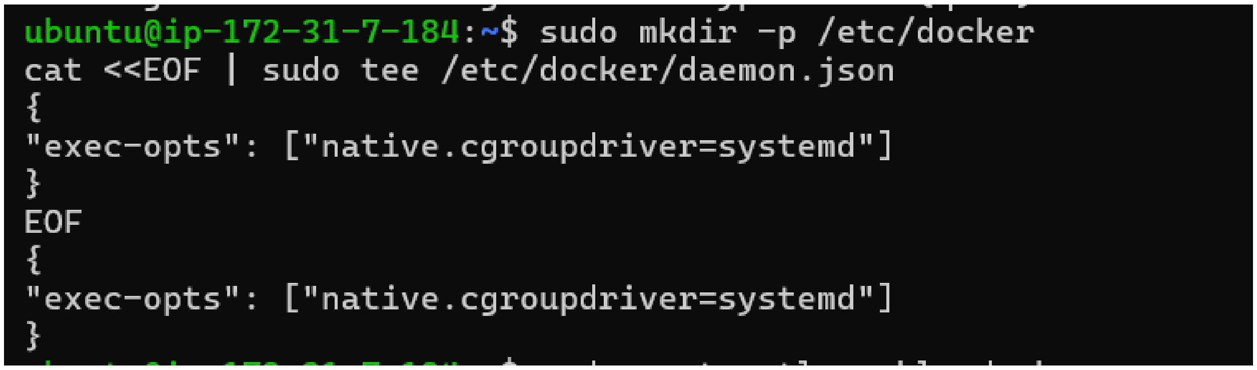


```
Selecting previously unselected package pigz.
(Reading database ... 67741 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../1-containerd.io_1.7.22-1_amd64.deb ...
Unpacking containerd.io (1.7.22-1) ...
Selecting previously unselected package docker-buildx-plugin.
Preparing to unpack .../2-docker-buildx-plugin_0.17.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-buildx-plugin (0.17.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../3-docker-ce-cli_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-cli (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../4-docker-ce_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-rootless-extras.
Preparing to unpack .../5-docker-ce-rootless-extras_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-rootless-extras (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-compose-plugin.
Preparing to unpack .../6-docker-compose-plugin_2.29.7-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-compose-plugin (2.29.7-1~ubuntu.24.04~noble) ...
Selecting previously unselected package libltdl7:amd64.
Preparing to unpack .../7-libltdl7_2.4.7-7build1_amd64.deb ...
```



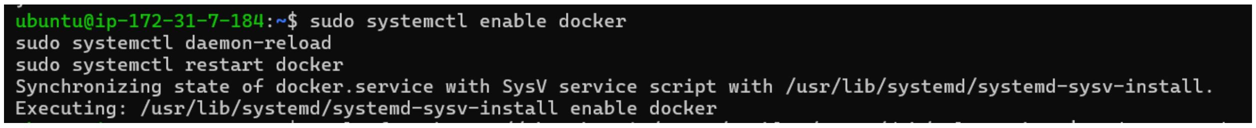
```
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up docker-buildx-plugin (0.17.1-1~ubuntu.24.04~noble) ...
Setting up containerd.io (1.7.22-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.29.7-1~ubuntu.24.04~noble) ...
Setting up libltdl7:amd64 (2.4.7-7build1) ...
Setting up docker-ce-cli (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

```
sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
```



```
ubuntu@ip-172-31-7-184:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
```

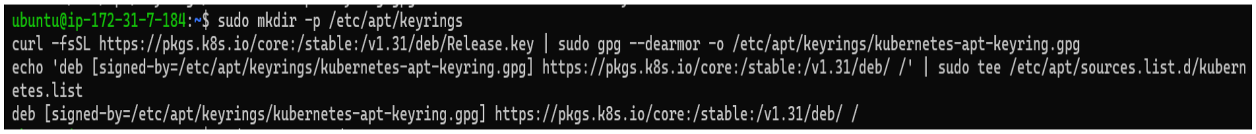
```
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
```



```
ubuntu@ip-172-31-7-184:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

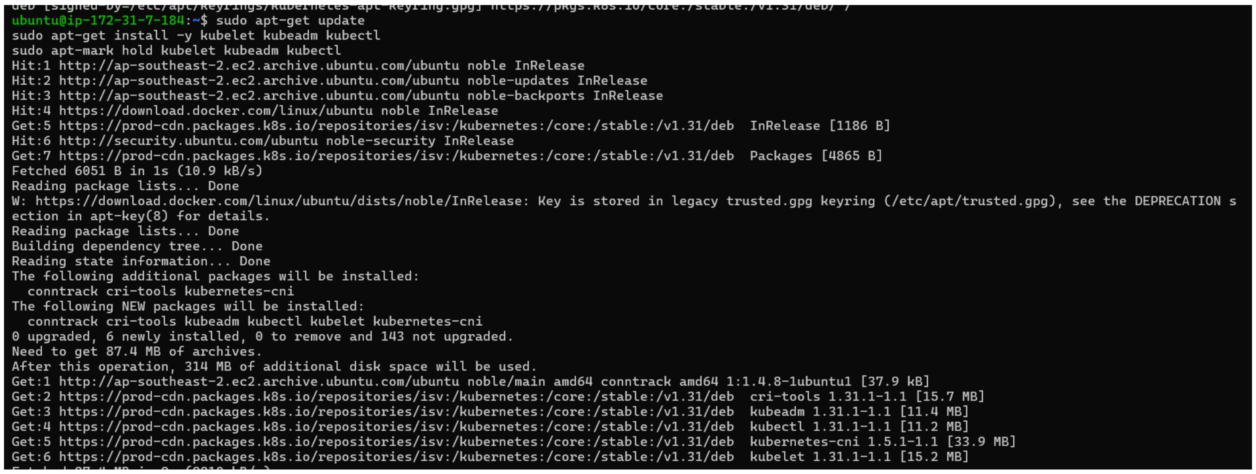
5. Run the below command to install Kubernets.
   curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

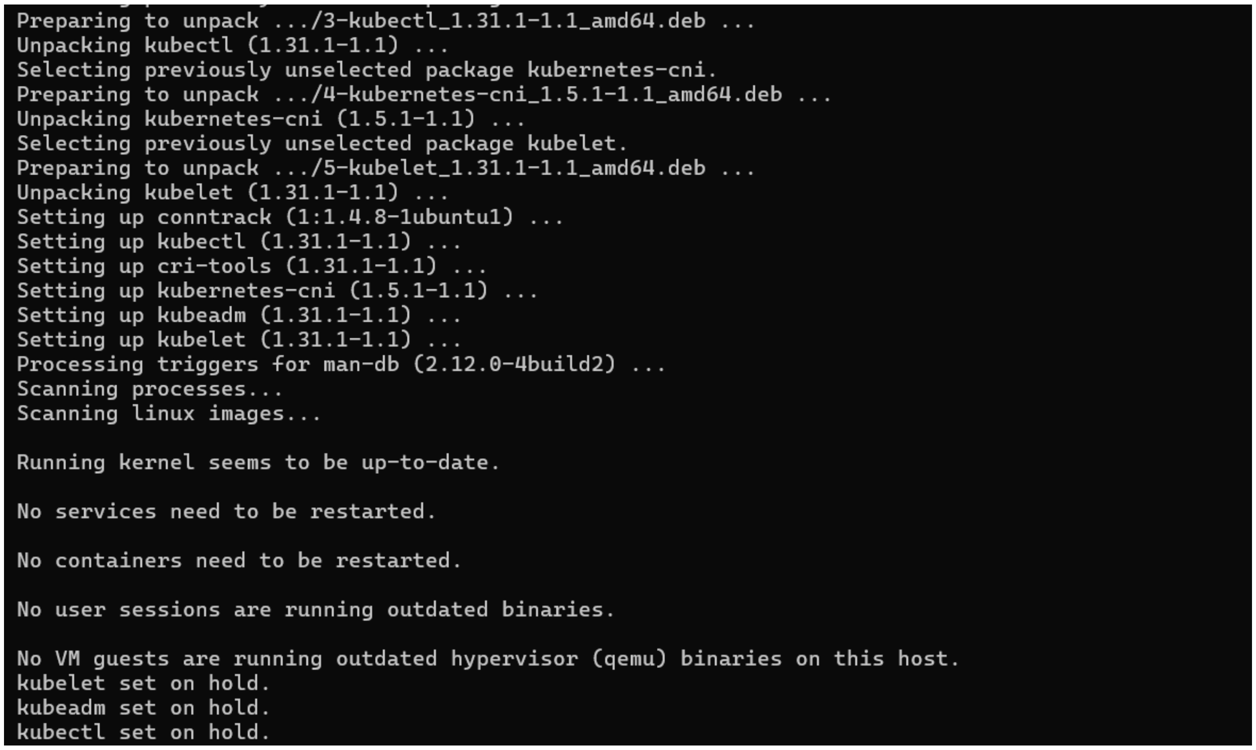


```
ubuntu@ip-172-31-7-184:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubern
etes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

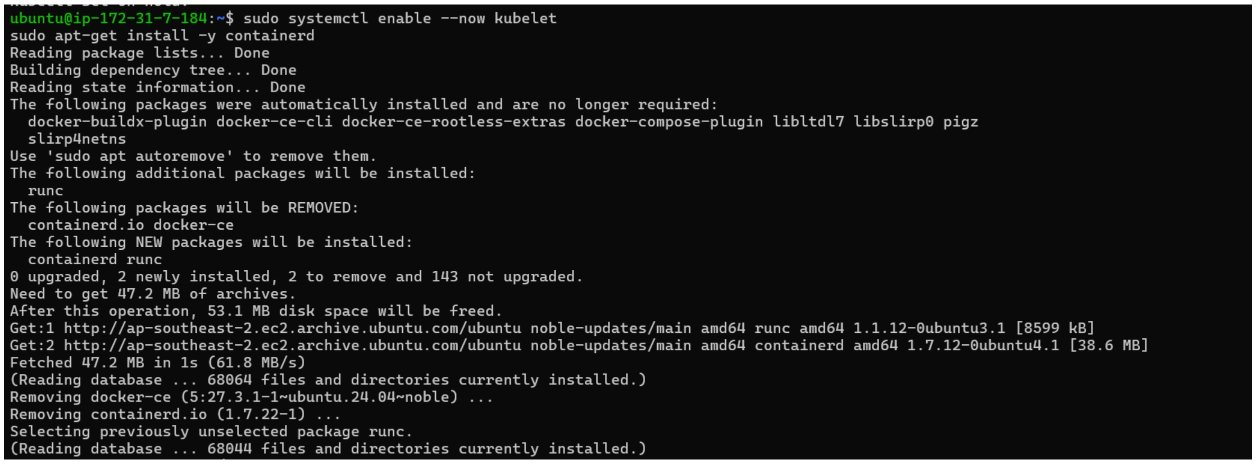


```
ubuntu@ip-172-31-7-184:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 1s (10.9 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 143 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
```

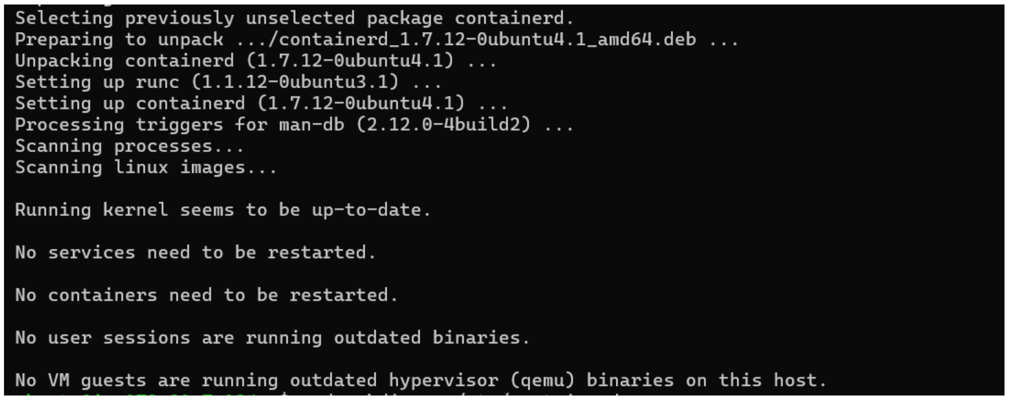


```
Preparing to unpack .../3-kubectl_1.31.1-1.1_amd64.deb ...
Unpacking kubectl (1.31.1-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.5.1-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.5.1-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../5-kubelet_1.31.1-1.1_amd64.deb ...
Unpacking kubelet (1.31.1-1.1) ...
Setting up conntrack (1:1.4.8-1ubuntu1) ...
Setting up kubectl (1.31.1-1.1) ...
Setting up cri-tools (1.31.1-1.1) ...
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
```

sudo systemctl enable --now kubelet
sudo apt-get install -y containerd

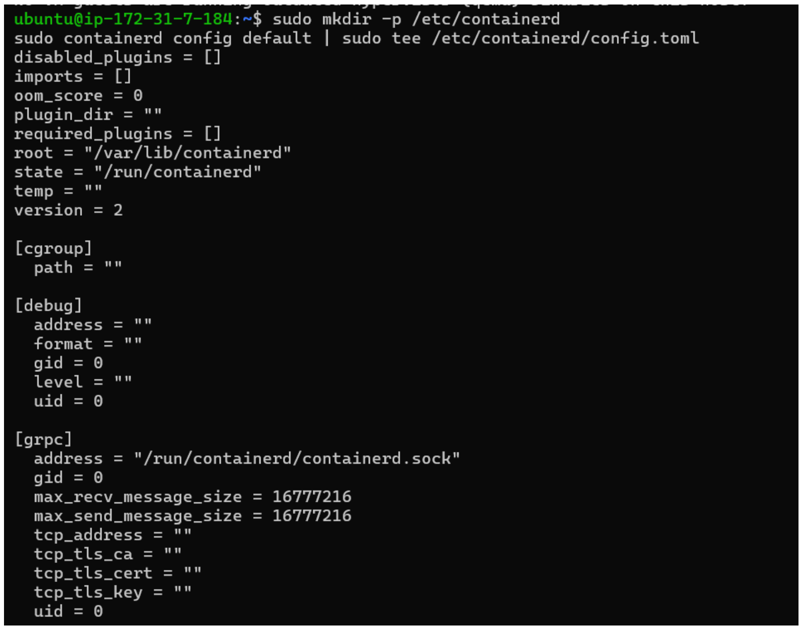


```
ubuntu@ip-172-31-7-184:~$ sudo systemctl enable --now kubelet
sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 143 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (61.8 MB/s)
(Reading database ... 68064 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68044 files and directories currently installed.)
```

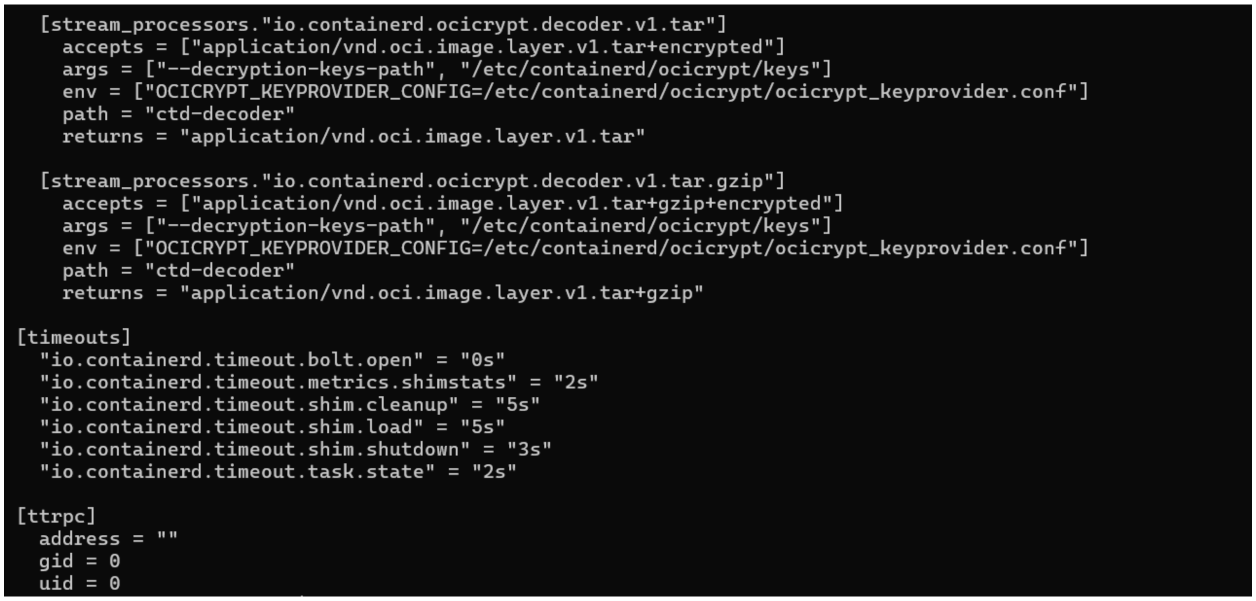


```
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

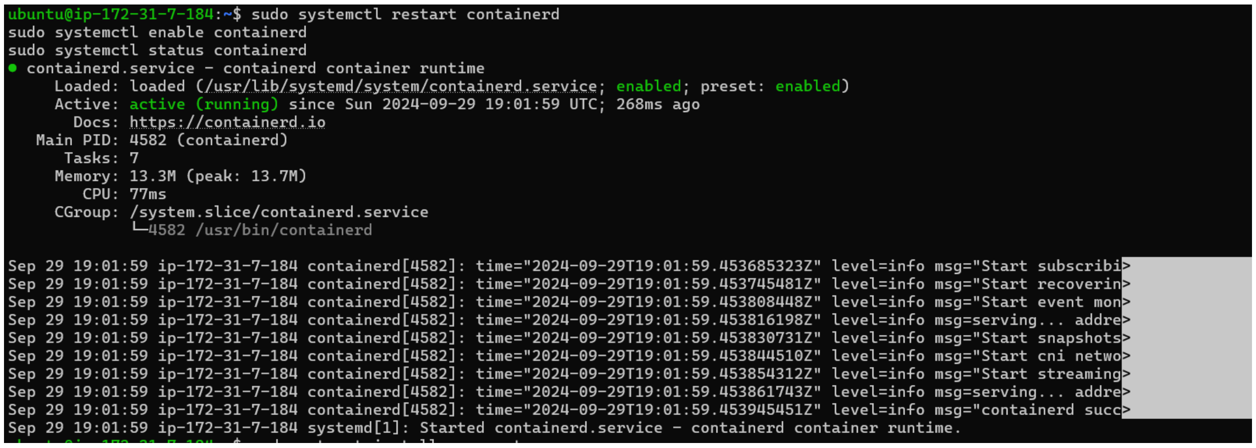


```
ubuntu@ip-172-31-7-184:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0
```



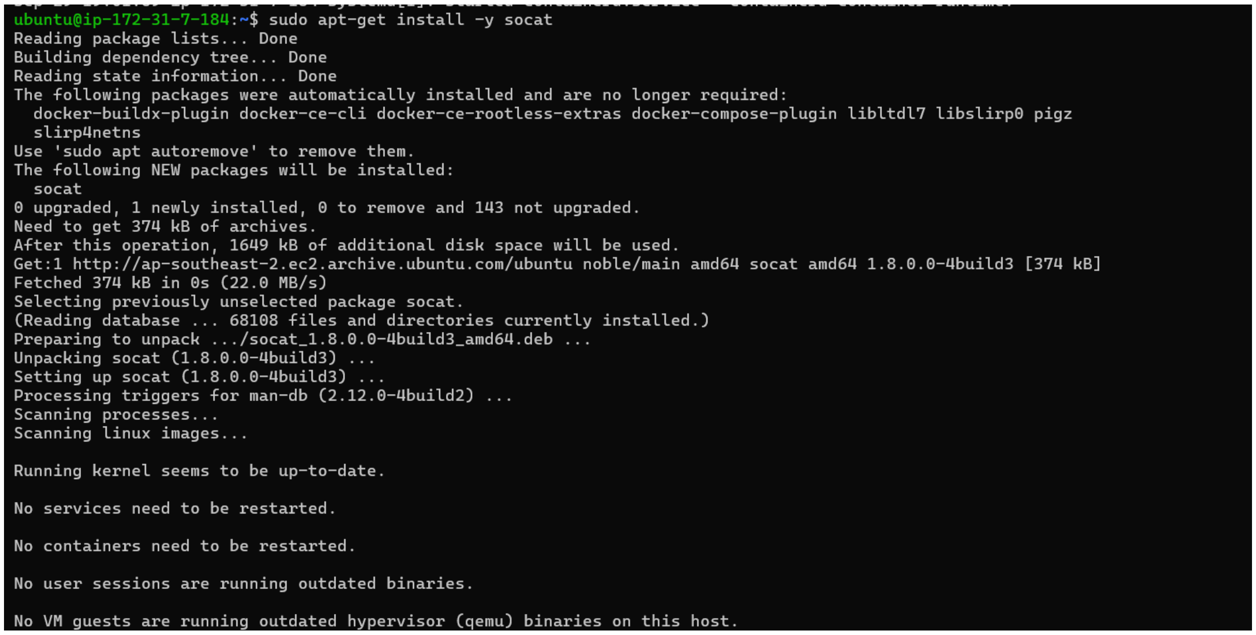
```
  [stream_processors."io.containerd.ocicrypt.decoder.v1.tar"]
    accepts = ["application/vnd.oci.image.layer.v1.tar+encrypted"]
    args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
    env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
    path = "ctd-decoder"
    returns = "application/vnd.oci.image.layer.v1.tar"

  [stream_processors."io.containerd.ocicrypt.decoder.v1.tar.gzip"]
    accepts = ["application/vnd.oci.image.layer.v1.tar+gzip+encrypted"]
    args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
    env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
    path = "ctd-decoder"
    returns = "application/vnd.oci.image.layer.v1.tar+gzip"

[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shimstats" = "2s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"

[ttrpc]
  address = ""
  gid = 0
  uid = 0
```

sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd



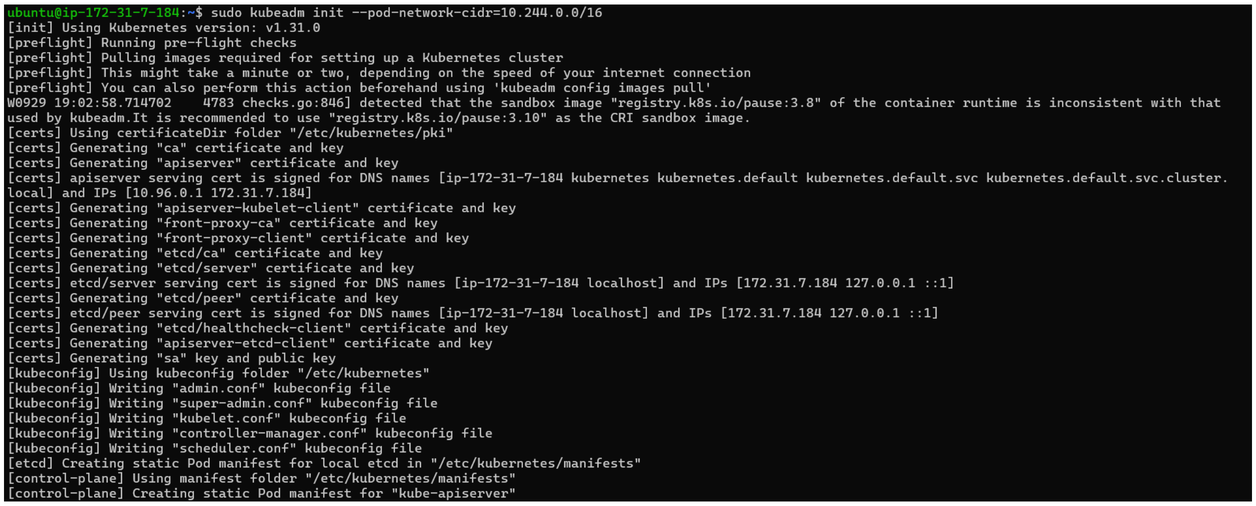
```
ubuntu@ip-172-31-7-184:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Sun 2024-09-29 19:01:59 UTC; 268ms ago
       Docs: https://containerd.io
   Main PID: 4582 (containerd)
      Tasks: 7
     Memory: 13.3M (peak: 13.7M)
        CPU: 77ms
     CGroup: /system.slice/containerd.service
             └─4582 /usr/bin/containerd

Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453685323Z" level=info msg="Start subscribi>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453745481Z" level=info msg="Start recoverin>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453808448Z" level=info msg="Start event mon>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453816198Z" level=info msg=serving... addre>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453830731Z" level=info msg="Start snapshots>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453844510Z" level=info msg="Start cni netwo>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453854312Z" level=info msg="Start streaming>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453861743Z" level=info msg=serving... addre>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453945451Z" level=info msg="containerd succ>
Sep 29 19:01:59 ip-172-31-7-184 systemd[1]: Started containerd.service - containerd container runtime.
```

sudo apt-get install -y socat

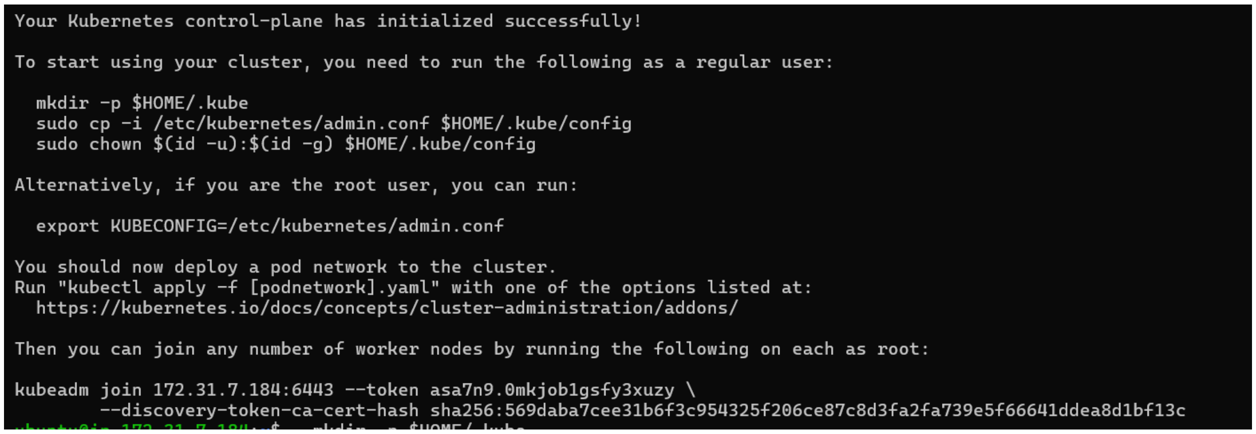


```
ubuntu@ip-172-31-7-184:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 143 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (22.0 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

6. Initialize the Kubecluster .Now Perform this Command only for Master.
   sudo kubeadm init --pod-network-cidr=10.244.0.0/16

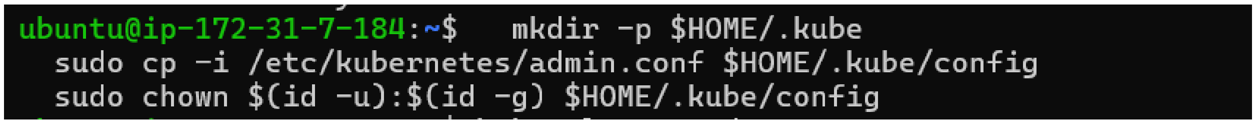


```
ubuntu@ip-172-31-7-184:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0929 19:02:58.714702    4783 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that
used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-7-184 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.
local] and IPs [10.96.0.1 172.31.7.184]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-7-184 localhost] and IPs [172.31.7.184 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-7-184 localhost] and IPs [172.31.7.184 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
```



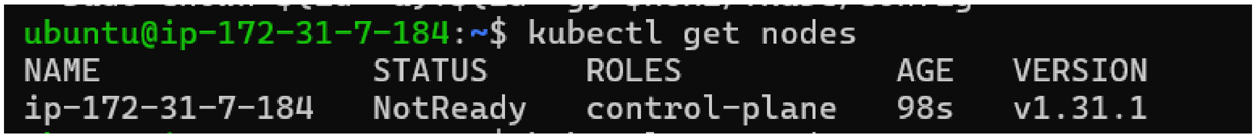
```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
```

Run this command on master and also copy and save the Join command from above.
mkdir -p $HOME/.kube
 sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
 sudo chown $(id -u):$(id -g) $HOME/.kube/config



```
ubuntu@ip-172-31-7-184:~$   mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

7. Now Run the command kubectl get nodes to see the nodes before executing Join command on nodes.

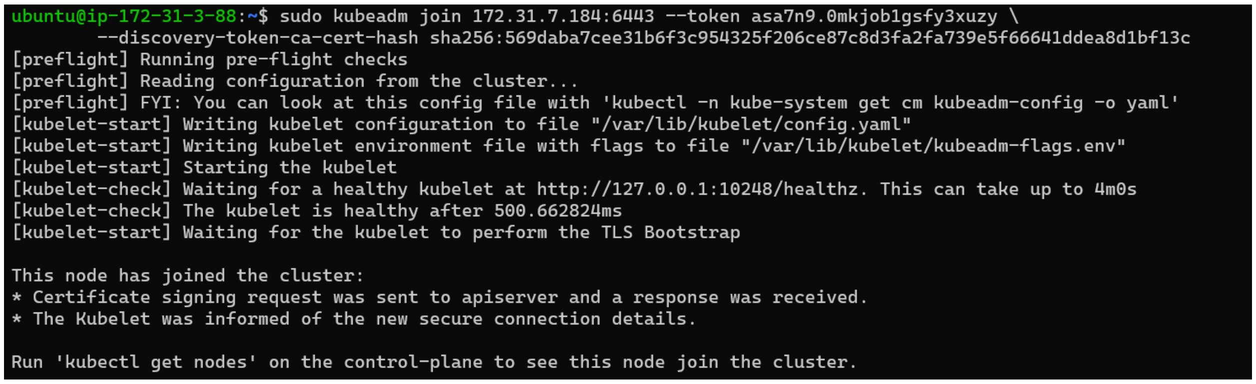


```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME               STATUS     ROLES           AGE   VERSION
ip-172-31-7-184    NotReady   control-plane   98s   v1.31.1
```

8. Now Run the following command on Node 1 and Node 2 to Join to master

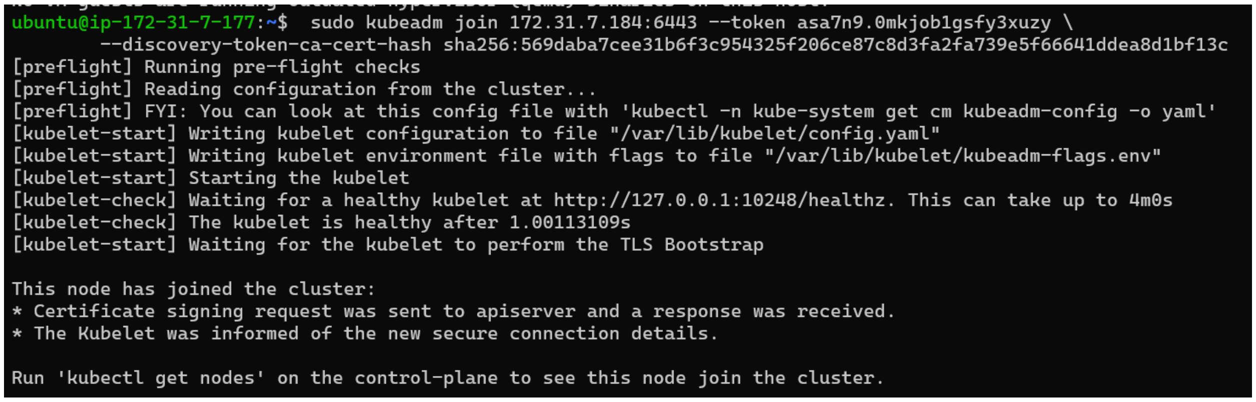
sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
  --discovery-token-ca-cert-hash
sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c

Node 1:



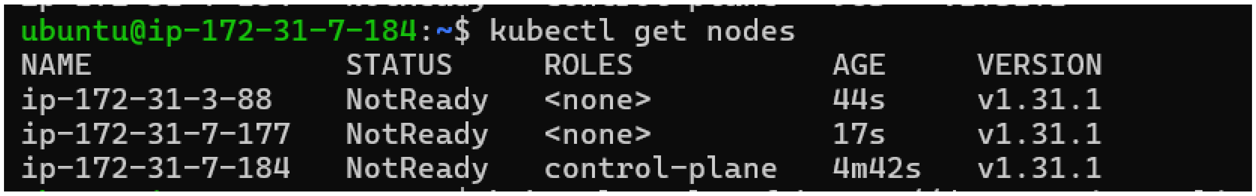
```
ubuntu@ip-172-31-3-88:~$ sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 500.662824ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Node 2:



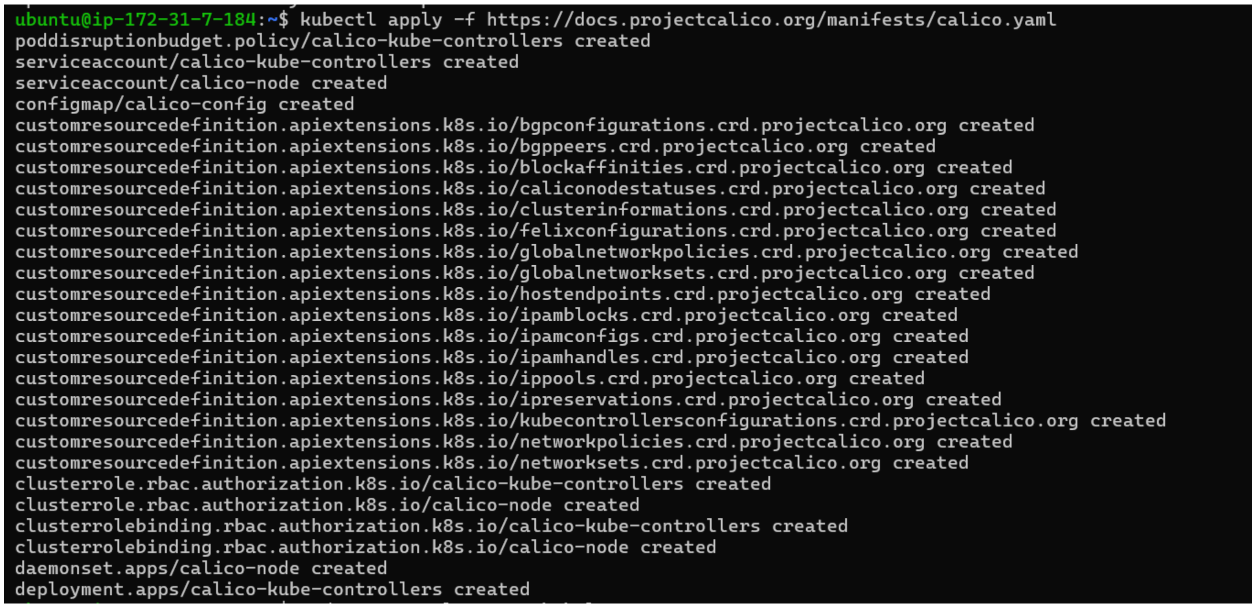
```
ubuntu@ip-172-31-7-177:~$  sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.00113109s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

9. Now Run the command kubectl get nodes to see the nodes after executing Join command on nodes.



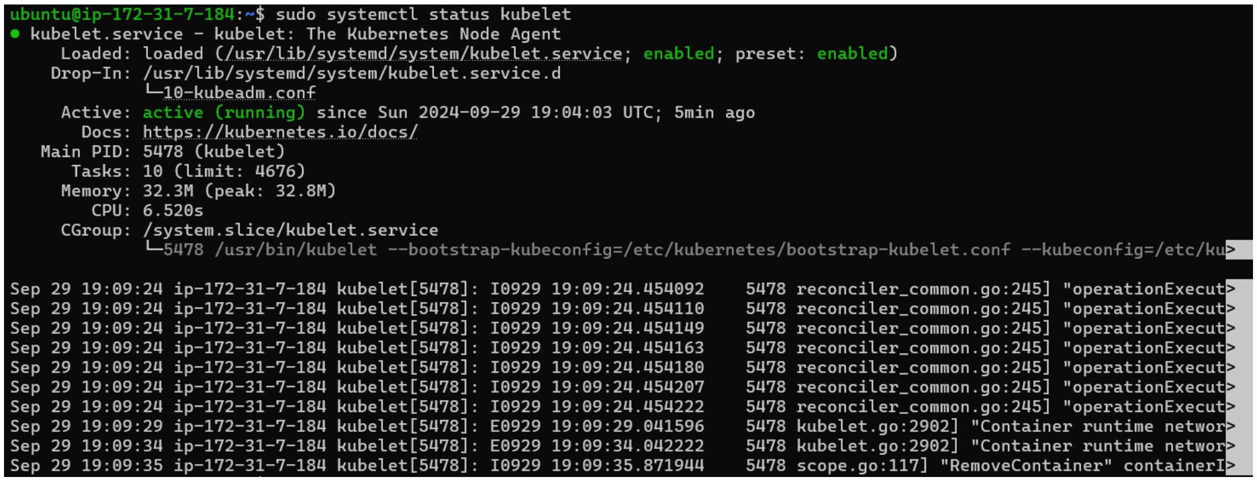
```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME               STATUS     ROLES           AGE     VERSION
ip-172-31-3-88     NotReady   <none>          44s     v1.31.1
ip-172-31-7-177    NotReady   <none>          17s     v1.31.1
ip-172-31-7-184    NotReady   control-plane   4m42s   v1.31.1
```

10. Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.
    kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml



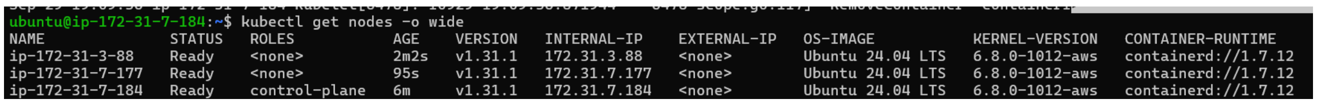
```
ubuntu@ip-172-31-7-184:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

sudo systemctl status kubelet



```
ubuntu@ip-172-31-7-184:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Sun 2024-09-29 19:04:03 UTC; 5min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 5478 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 32.3M (peak: 32.8M)
        CPU: 6.520s
     CGroup: /system.slice/kubelet.service
             └─5478 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/ku>

Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454092    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454110    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454149    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454163    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454180    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454207    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454222    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:29 ip-172-31-7-184 kubelet[5478]: E0929 19:09:29.041596    5478 kubelet.go:2902] "Container runtime networ>
Sep 29 19:09:34 ip-172-31-7-184 kubelet[5478]: E0929 19:09:34.042222    5478 kubelet.go:2902] "Container runtime networ>
Sep 29 19:09:35 ip-172-31-7-184 kubelet[5478]: I0929 19:09:35.871944    5478 scope.go:117] "RemoveContainer" containerI>
```

Now Run command kubectl get nodes -o wide we can see Status is ready.

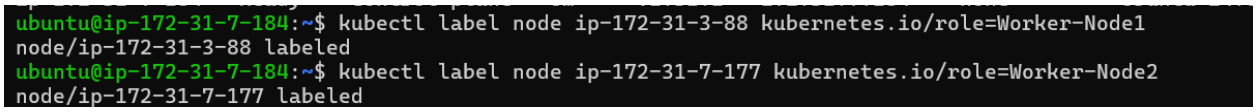


```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes -o wide
NAME               STATUS   ROLES           AGE    VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION    CONTAINER-RUNTIME
ip-172-31-3-88     Ready    <none>          2m2s   v1.31.1   172.31.3.88    <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws    containerd://1.7.12
ip-172-31-7-177    Ready    <none>          95s    v1.31.1   172.31.7.177   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws    containerd://1.7.12
ip-172-31-7-184    Ready    control-plane   6m     v1.31.1   172.31.7.184   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws    containerd://1.7.12
```

Now to Rename run this command kubectl label node ip-172-31-18-135 kubernetes.io/role=worker

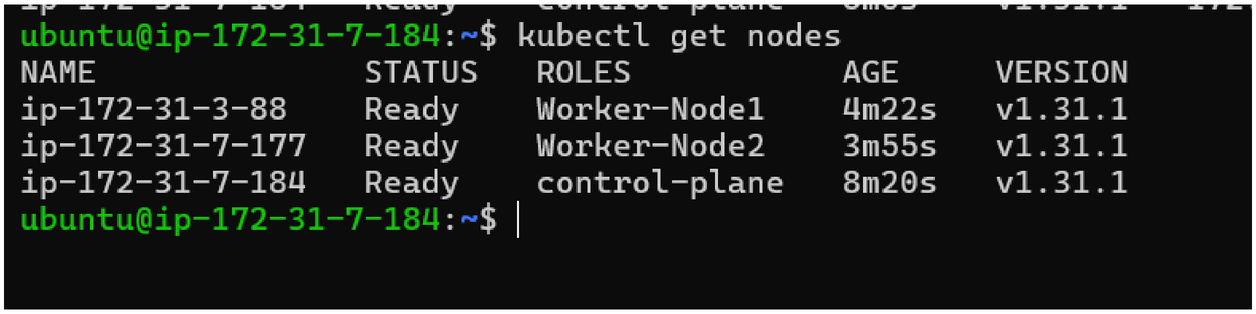
Rename to Node 1:kubectl label node ip-172-31-3-88 kubernetes.io/role=Worker-Node1

Rename to Node 2:kubectl label node ip-172-31-7-177kubernetes.io/role=Worker-Node2



```
ubuntu@ip-172-31-7-184:~$ kubectl label node ip-172-31-3-88 kubernetes.io/role=Worker-Node1
node/ip-172-31-3-88 labeled
ubuntu@ip-172-31-7-184:~$ kubectl label node ip-172-31-7-177 kubernetes.io/role=Worker-Node2
node/ip-172-31-7-177 labeled
```

11. Now run kubectl get nodes



```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE     VERSION
ip-172-31-3-88    Ready    Worker-Node1    4m22s   v1.31.1
ip-172-31-7-177   Ready    Worker-Node2    3m55s   v1.31.1
ip-172-31-7-184   Ready    control-plane   8m20s   v1.31.1
ubuntu@ip-172-31-7-184:~$
```

Hence we can see we have Successfully connected Node 1 and Node 2 to the Master.

Conclusion:

In this experiment, we successfully set up a Kubernetes cluster with one master and two worker nodes on AWS EC2 instances. After installing Docker, Kubernetes tools (kubelet, kubeadm, kubectl), and containerd on all nodes, the master node was initialized and the worker nodes were joined to the cluster. Initially, the nodes were in the NotReady state, which was resolved by installing the Calico network plugin. We also labeled the nodes with appropriate roles (control-plane and worker). The cluster became fully functional with all nodes in the Ready state, demonstrating the successful configuration and orchestration of Kubernetes.