# Siddhant Sathe D15A/51

# **ADVANCE DEVOPS EXP-3**

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

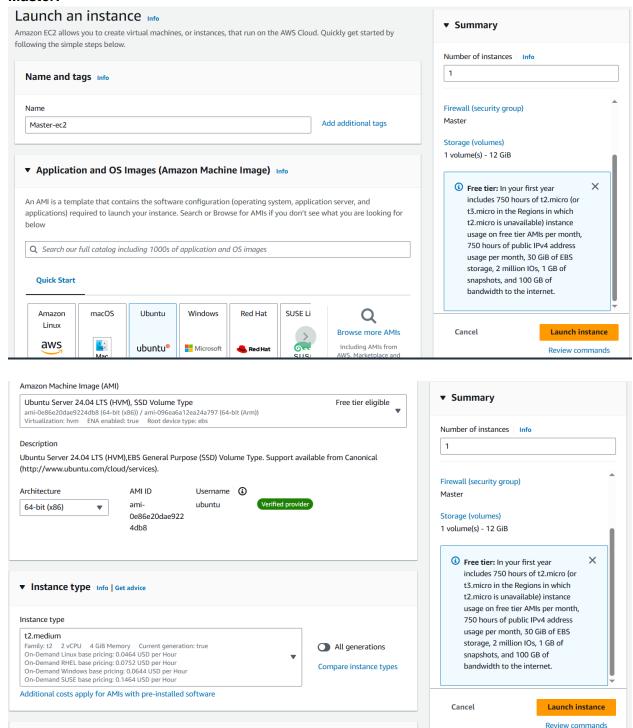
**Step 1**: Create 2 Security Groups for Master and Nodes and add the following inbound rules in those groups

#### Master: Inbound rules Info Protocol Info Port range Info Description - optional Info Security group rule ID sgr-0f10d6d8ca9898f4e SSH 22 Q Delete 0.0.0.0/0 × sgr-0a9b6b212dac59277 Custom TCP Delete ▼ TCP 10250 Custom Q 0.0.0.0/0 🗙 sgr-0a063e24fced60ee5 ▼ TCP 6443 Custom TCP Custom Q Delete 0.0.0.0/0 🗙 sgr-0ea153b52157b37ab ▼ TCP 10252 Custom TCP Custom Q Delete 0.0.0.0/0 🗙 sgr-08a6217572696188c HTTP 80 Custom Q Delete 0.0.0.0/0 × sgr-0a0bd17f3a5c22b46 Custom TCP Delete ▼ TCP 10251 Custom w Q 0.0.0.0/0 🗙 sgr-0c13dda434e17628f All TCP ▼ TCP 0 - 65535 Custom Q Delete 0.0.0.0/0 🗙 sgr-09d8b493e78aa2e80 Delete All traffic Custom Q Node: Inbound rules Info Security group rule ID sgr-0be6b7289168883a8 Custom TCP 30000 - 32767 Custom Q Delete 0.0.0.0/0 🗙 sgr-035e8c1dae322fa83 SSH 22 Custom Q Delete 0.0.0.0/0 🗙 sgr-0b011ea3327732231 All TCP ▼ TCP 0 - 65535 Custom ▼ Delete 0.0.0.0/0 🗙 sgr-0087387292cceaa9d All traffic Custom Delete 0.0.0.0/0 🗙 sgr-0a2a26d63b63c3bb1 Custom TCP 10250 Custom Delete 0.0.0.0/0 🗙 sgr-0dc9223a90b1037d1 HTTP 0.0.0.0/0 × Add rule

**Step 2:** Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances(1 for Master and 2 for Node). Select Ubuntu as AMI and t2.medium as Instance Type and create a key

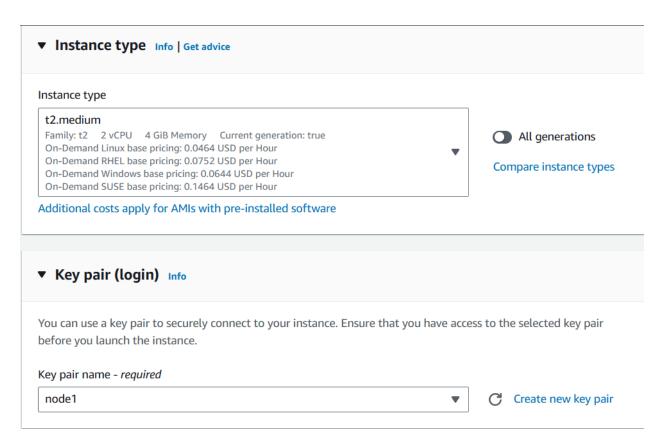
of type RSA with .pem extension and move the downloaded key to the new folder.We can use 2 Different keys, 1 for Master and 1 for Node. Also Select Security Groups from the existing.

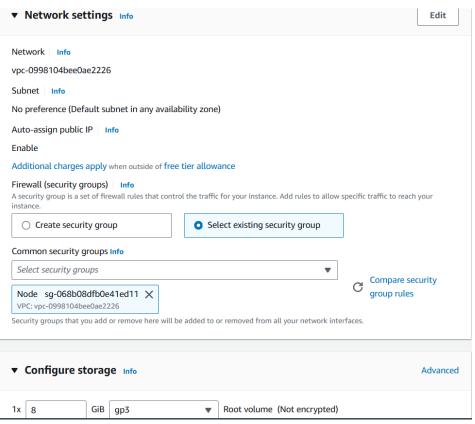
## Master:



# Node:

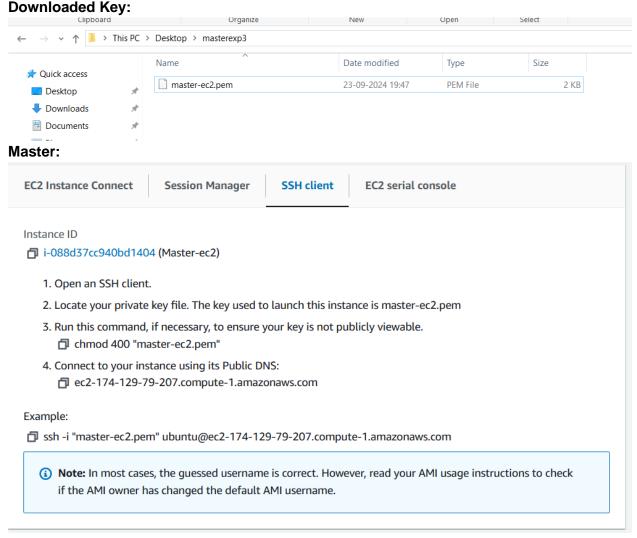
Name										
node1		Add additional tags								
▼ Application and OS Images (Amazon Machine Image) Info										
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below										
Q Search our full catalog including 1000s of application and OS images										
Recents Quick Start										
Amazon	macOS	Ubuntu	Windows	Red Hat	SUSE Li	Q				
aws	Mac	ubuntu <sup>®</sup>	Microsoft	🦀 Red Hat	SUSI	Browse more AMIs Including AMIs from AWS, Marketplace and the Community				
Amazon Machine Image (AMI)										
ami-0e86e20da	er <b>24.04 LTS (HV</b> ) e9224db8 (64-bit ( ovm ENA enabled	(x86)) / ami-096ea	a6a12ea24a797 (6	4-bit (Arm))		Free tier eligible				





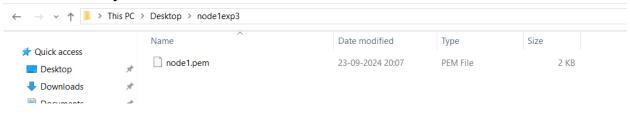


**Step 3:** Connect the instance and navigate to SSH client and copy the example command. Now open the folder in the terminal 3 times for Master, Node1 & Node 2 where our .pem key is stored and paste the Example command from ssh client (starting with ssh -i .....) in the terminal.



```
PS C:\Users\sathe> cd Desktop
PS C:\Users\sathe\Desktop> cd .\masterexp3\
PS C:\Users\sathe\Desktop\masterexp3> <mark>ssh</mark> -i "master-ec2.pem" ubuntu@ec2-174-129-79-207.compute-1.amazonaws.com
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 Mon Sep 23 15:00:49 UTC 2024
 System load: 0.0 Processes: Usage of /: 14.5% of 10.58GB Users logged in:
 Memory usage: 5%
                                   IPv4 address for enX0: 172.31.90.179
 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
Last login: Mon Sep 23 14:58:49 2024 from 58.146.120.240
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

# **Downloaded Key Node 1:**



## Node 1:

EC2 Instance Connect Session Manager SSH client EC2 serial console

#### Instance ID

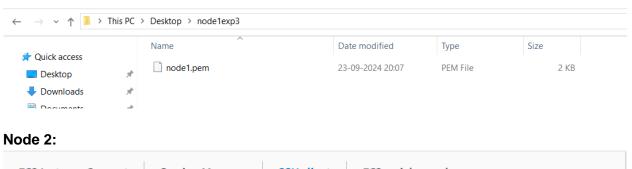
- i-08cbf923421fdc6ad (node1)
  - 1. Open an SSH client.
  - 2. Locate your private key file. The key used to launch this instance is node1.pem
  - 3. Run this command, if necessary, to ensure your key is not publicly viewable.
    - d chmod 400 "node1.pem"
  - 4. Connect to your instance using its Public DNS:
    - d ec2-35-170-201-60.compute-1.amazonaws.com

#### Example:

- ssh -i "node1.pem" ubuntu@ec2-35-170-201-60.compute-1.amazonaws.com
  - (i) **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.

```
ubuntu@ec2-35-170-201-60.compute-1.amazonaws.com
PS C:\Users\sathe\Desktop\node1exp3> ssh -i
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 Mon Sep 23 15:04:19 UTC 2024
                                                        113
 System load: 0.0
                                 Processes:
 Usage of /: 22.9% of 6.71GB Users logged in:
 Memory usage: 5%
                                 IPv4 address for enX0: 172.31.90.171
 Swap usage:
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
Last login: Mon Sep 23 14:41:42 2024 from 18.206.107.27
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo root" for details.
```

# **Downloaded Key:**



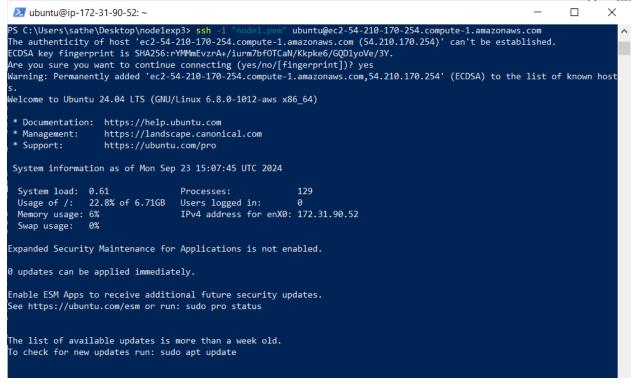


## Instance ID

- - 1. Open an SSH client.
  - 2. Locate your private key file. The key used to launch this instance is node1.pem
  - 3. Run this command, if necessary, to ensure your key is not publicly viewable.
    - d chmod 400 "node1.pem"
  - 4. Connect to your instance using its Public DNS:
    - ec2-54-210-170-254.compute-1.amazonaws.com

# Example:

- ssh -i "node1.pem" ubuntu@ec2-54-210-170-254.compute-1.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.



**Step 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"

```
.79:∘$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
wnload.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))
buntu@ip-172-31-90-179:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
 ----BEGIN PGP PUBLIC KEY BLOCK-----
mQINBFit2ioBEADhWpZ8/wvZ6hUTiXOwQHXMAlaFHcPH9hAtr4F1y2+OYdbtMuth
lqqwp028AqyY+PRfVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh
B8UuLa+z077PxyxQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOXbnq
_4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlANyEV1Zzeqo7XK17
JrwV5inawTSzWNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N
mhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht
ku4bIQN1X5P07fNWzlgaRL5Z4POXDDZT1IQ/E158j9kp4bnWRCJW01ya+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtl/94VFYVJ01eAv8W92KdgDkhTcTD
G7c0tIkVEKNUq48b3aQ64NOZQW7fVjfoKwEZdOqPE72Pa45jrZzvUFxSpdiNk2tZ
(YukHjlxxEgBdC/J3cMMNRE1F4NCA3ApfV1Y7/hTeOnmDuDYwr9/obA8t016Yljj
q5rdkywPf4JF8mXUW5eCN1vAFHxeg9ZWemhBtQmGxXnw9M+z6hWwc6ahmwARAQAB
tCtEb2NrZXIgUmVsZWFzZSAoQ0UgZGViKSA8ZG9ja2VyQGRvY2tlci5jb20+iQI3
BBMBCgAhBQJYrefAAhsvBQsJCAcDBRUKCQgLBRYCAwEAAh4BAheAAAoJEI2BgDwO
v82IsskP/iQZo68f1DQmNvn8X5XTd6RRaUH33kXYXquT6NkHJciS7E2gTJmqvMqd
```

```
179: $ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
> $(lsb_release -cs) stable"
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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
[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 4s (7570 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 <code>DEPRECATION</code> section in <code>apt-key(8)</code> for <code>details.</code>
```

- sudo apt-get update
- sudo apt-get install -y docker-ce
- sudo mkdir -p /etc/docker
   cat <<EOF | sudo tee /etc/docker/daemon.json</li>
   "exec-opts": ["native.cgroupdriver=systemd"]
   EOF

```
ubuntu@ip-172-31-90-171: ~
              -90-171:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o/et 🔨
stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
             31-90-171:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports 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 0s (12.2 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 section in apt-key(8) for details.
           2-31-90-171:~$ sudo apt-get install -y kubelet kubeadm kubectl
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 139 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://us-east-1.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
MB1
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 M
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubectl 1.31.1-1.1 [11.2 M
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubernetes-cni 1.5.1-1.1
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubelet 1.31.1-1.1 [15.2 M
Fetched 87.4 MB in 1s (83.2 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 68007 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3a1.4.8-1ubuntu1_amd64.deb ...
Unpacking conntrack (1:1.4.8-1ubuntu1) ...
Selecting previously unselected package cri-tools.
Preparing to unpack .../1-cri-tools_1.31.1-1.1_amd64.deb ...
Unpacking cri-tools (1.31.1-1.1) ..
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.31.1-1.1_amd64.deb ...
Unpacking kubeadm (1.31.1-1.1) ...
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
ubuntu@ip-172-31-90-179:∾$ sudo mkdir -p /etc/docker
 sudo tee /etc/docker/daemon.json
"exec-opts": ["native.cgroupdriver=systemd"]
EOFcat <<EOF | sudo tee /etc/docker/daemon.json
 exec-opts": ["native.cgroupdriver=systemd"]
EOFubuntu@ip-172-31-90-179:~$ sudo systemctl enable docker
```

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

```
EOFubuntu@ip-172-31-90-179:-$ sudo systemctl enable docker
o systemctl daemon-reload
sudo systemctl restart dockersudo systemctl daemon-reload
sudo systemctl restart dockerSynchronizing state of docker.service with SysV service script with /usr/lib/systemd/system
d-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

# Step 5: Run the below command to install Kubernetes.

- 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-172-31-90-179:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o/et c/apt/keyrings/kubernetes-apt-keyring.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N) y
ubuntu@ip-172-31-90-179:~$ 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
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

```
@ip-172-31-90-179:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Hit:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/ap
/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
    tu@ip-172-31-90-179:~$ sudo apt-get install -y kubelet kubeadm kubectl
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 139 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://us-east-1.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.
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4
```

```
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.

ubuntu@ip-172-31-90-179:~$ sudo apt-mark hold kubelet kubeadm kubectl

kubelet set on hold.

kubeadm set on hold.

kubectl set on hold.
```

- sudo systemctl enable --now kubelet
- sudo apt-get install -y containerd
- sudo mkdir -p /etc/containerd
- sudo containerd config default | sudo tee /etc/containerd/config.toml

```
buntu@ip-172-31-90-179:~$ sudo systemctl enable --now kubelet
 ountu@ip-172-31-90-179:~$ 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
Jse 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
The following packages will be REMOVED:
containerd.io docker-ce
The following NEW packages will be installed:
 containerd runc
doupgraded, 2 newly installed, 2 to remove and 139 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 M
```

```
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.
ubuntu@ip-172-31-90-179:~$ sudo mkdir -p /etc/containerd
ubuntu@ip-172-31-90-179:∼$ 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]
```

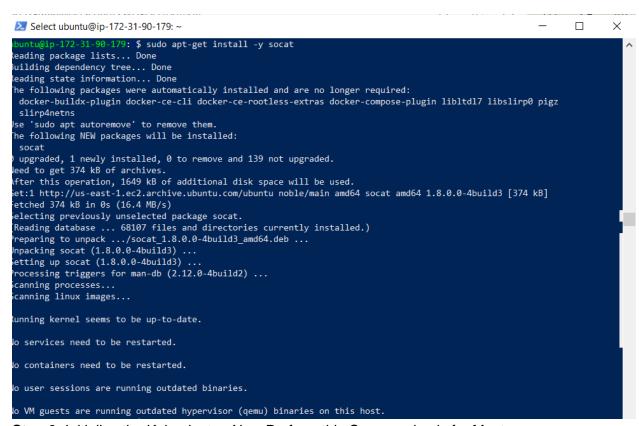
```
[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shimstats" = "2s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"

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

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

```
179:~$ sudo systemctl restart containerd
  buntu@ip-172-31-90-179:~$ sudo systemctl enable containerd
 buntu@ip-172-31-90-179:~$ sudo systemctl status containerd
  containerd.service - containerd container runtime
      Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
      Active: active (running) since Mon 2024-09-23 15:48:22 UTC; 30s ago
        Docs: https://containerd.io
   Main PID: 6350 (containerd)
       Tasks: 8
      Memory: 13.1M (peak: 13.8M)
         CPU: 228ms
      CGroup: /system.slice/containerd.service
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390407424Z" level=info msg="Start subscrib
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390457482Z" level=info msg="Start recoveri
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390457512Z" level=info msg=serving... addr Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390506097Z" level=info msg=serving... addr
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390528794Z" level=info msg="Start event mo
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390542029Z" level=info msg="Start snapshot
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390550998Z" level=info msg="Start cni netw
Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.390557855Z" level=info msg="Start streaming
Sep 23 15:48:22 ip-172-31-90-179 systemd[1]: Started containerd.service - containerd container runtime.
"Sep 23 15:48:22 ip-172-31-90-179 containerd[6350]: time="2024-09-23T15:48:22.394691700Z" level=info msg="containerd suc
```

sudo apt-get install -y socat



**Step 6:** Initialize the Kubecluster .Now Perform this Command only for Master.

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
buntu@ip-172-31-90-179: $ 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'
0923 15:52:44.615475 6699 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the containe
runtime is inconsistent with that used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI san
blox image.
certs] Using certificateDir folder "/etc/kubernetes/pki"
certs] Generating "an' certificate and key
certs] Generating "apiserver" certificate and key
certs] apiserver serving cert is signed for DNS names [ip-172-31-90-179 kubernetes kubernetes.default kubernetes.default
.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.90.179]
certs] Generating "apiserver-kubelet-client" certificate and key
certs] Generating "front-proxy-ca" certificate and key
certs] Generating "front-proxy-ca" certificate and key
```

```
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 172.31.90.179:6443 --token aw1zqe.njkmrtdi6k1zpz5k \
 --discovery-token-ca-cert-hash sha256:2c1c7752cb31b471bfb5281cc2426e96c19b9f06d0df9749f5aca61691986791
```

# Copy the kudeadm join any number of worker nodes command to use it later for joining Node 1 and Node 2 with master

- 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-172-31-90-179:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-90-179:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
cp: overwrite '/home/ubuntu/.kube/config'? yes
ubuntu@ip-172-31-90-179:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

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

```
ubuntu@ip-172-31-90-179:~$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
ip-172-31-90-179 NotReady control-plane 3m45s v1.31.1
```

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

sudo kubeadm join 172.31.95.244:6443 --token kzfth2.ug3970lp3qeeieb4\ --discovery-token-ca-cert-hash
 sha256:dec27d33f1bfd1dca7a50caa2c05d4cad1d0a18aa88ad75c7ea83f15c529f4ca

#### Node 1:

## Node 2:

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

```
ubuntu@ip-172-31-90-179:~$ kubectl get nodes
                  STATUS
NAME
                             ROLES
                                                  VERSION
                                            AGE
ip-172-31-90-171
                  NotReady
                             <none>
                                            31s
                                                  v1.31.1
ip-172-31-90-179
                  NotReady
                             control-plane
                                            17m
                                                  v1.31.1
ip-172-31-90-52
                  NotReady
                            <none>
                                            22s
                                                  v1.31.1
```

**Step 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-172-31-90-179: $ 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/blockaffinities.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/felixconfigurations.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
```

sudo systemctl status kubelet

```
..√$ 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 Mon 2024-09-23 15:53:03 UTC; 19min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 7369 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 32.5M (peak: 33.0M)
        CPU: 16.010s
     CGroup: /system.slice/kubelet.service

L-7369 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/ku>
Sep 23 16:12:08 ip-172-31-90-179 kubelet[7369]: I0923 16:12:08.364641
                                                                                7369 pod_container_deletor.go:80] "Container r
                                                                                7369 scope.go:117] "RemoveContainer" container 7369 scope.go:117] "RemoveContainer" container
Sep 23 16:12:08 ip-172-31-90-179 kubelet[7369]: I0923 16:12:08.365702
Sep 23 16:12:08 ip-172-31-90-179 kubelet[7369]: I0923 16:12:08.370899
                                                                                7369 scope.go:117] "RemoveContainer" container
Sep 23 16:12:08 ip-172-31-90-179 kubelet[7369]: I0923 16:12:08.377678
Sep 23 16:12:09 ip-172-31-90-179 kubelet[7369]: I0923 16:12:09.368160
                                                                                7369 pod_container_deletor.go:80] "Container
Sep 23 16:12:09 ip-172-31-90-179 kubelet[7369]: I0923 16:12:09.368187
                                                                                7369 scope.go:117] "RemoveContainer" container
Sep 23 16:12:09 ip-172-31-90-179 kubelet[7369]: E0923 16:12:09.456550
                                                                                7369 pod_workers.go:1301] "Error syncing pod, 7369 scope.go:117] "RemoveContainer" containe
Sep 23 16:12:10 ip-172-31-90-179 kubelet[7369]: I0923 16:12:10.385576
Sep 23 16:12:10 ip-172-31-90-179 kubelet[7369]: E0923 16:12:10.385687
                                                                                7369 pod_workers.go:1301] "Error syncing pod,
Sep 23 16:12:12 ip-172-31-90-179 kubelet[7369]: I0923 16:12:12.385087
                                                                                7369 scope.go:117] "RemoveContainer" container
```

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

```
kubectl get nodes -o wide
                                                                                                               KERNEL-VE
NAME
                  STATUS
                           ROLES
                                           AGE
                                                   VERSION
                                                             INTERNAL-IP
                                                                             EXTERNAL-IP
                                                                                           OS-TMAGE
RSION
      CONTAINER-RUNTIME
                                                                                           Ubuntu 24.04 LTS
                                                                                                               6.8.0-101
ip-172-31-90-171
                  Ready
                           <none>
                                           2m15s
                                                   v1.31.1
                                                             172.31.90.171
                                                                             <none>
      containerd://1.7.12
ip-172-31-90-179 Ready
                                                   v1.31.1
                                                             172.31.90.179
                                                                                           Ubuntu 24.04 LTS
                                           19m
                                                                                                               6.8.0-101
                           control-plane
                                                                             <none>
-aws containerd://1.7.12
ip-172-31-90-52
                  Ready
                                                   v1.31.1 172.31.90.52
                                                                                           Ubuntu 24.04 LTS
                                                                                                               6.8.0-101
                           <none>
      containerd://1.7.12
buntu@ip-172-31-90-179:~$ kubectl label node ip-172-31-90-171 kubernetes.io/role=Node1
node/ip-172-31-90-171 labeled
```

The Roles are not yet assigned to the Nodes

- Rename to Node 1: kubectl label node ip-172-31-28-117 kubernetes.io/role=Node1
- Rename to Node 2: kubectl label node ip-172-31-18-135 kubernetes.io/role=Node2

```
ubuntu@ip-172-31-90-179:~$ kubectl label node ip-172-31-90-171 kubernetes.io/role=Node1
node/ip-172-31-90-171 labeled
ubuntu@ip-172-31-90-179:~$ kubectl label node ip-172-31-90-52 kubernetes.io/role=Node2
node/ip-172-31-90-52 labeled
```

• Run kubectl get nodes to check if roles are assigned now to the nodes

ubuntu@ip-172-31-9	0-179:~\$	kubectl get node	<u>!</u> S	
NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-90-171	Ready	Node1	62m	v1.31.1
ip-172-31-90-179	Ready	control-plane	80m	v1.31.1
ip-172-31-90-52	Ready	Node2	62m	v1.31.1