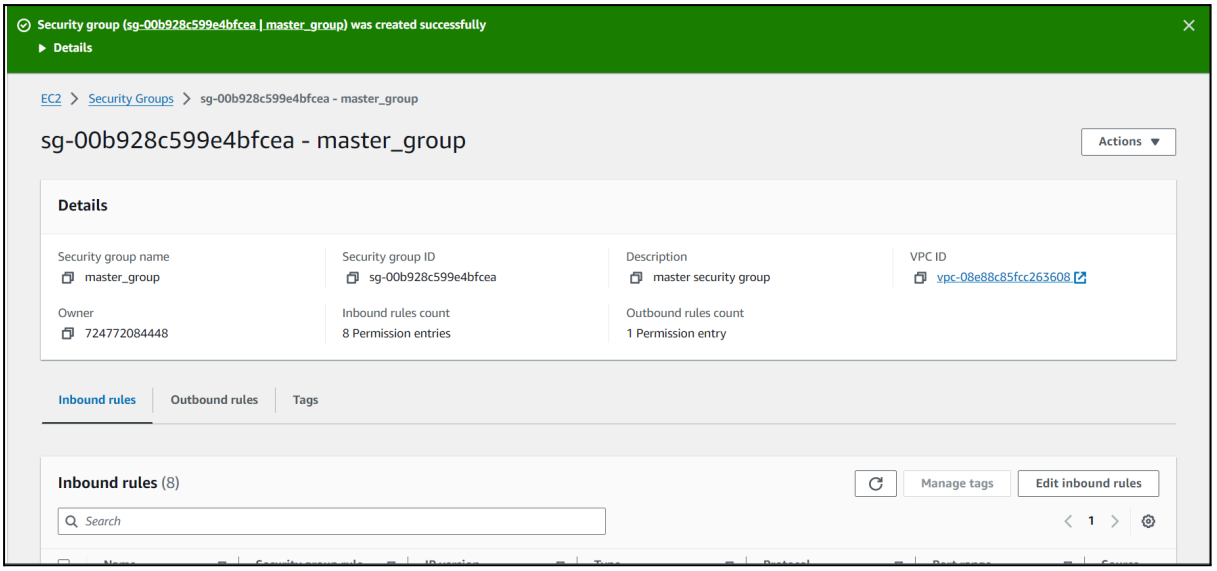
Experiment 3

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

Theory: Container-based microservices architectures have revolutionized how development and operations teams test and deploy modern software. Containers allow companies to scale and deploy applications more efficiently, but they also introduce new challenges, adding complexity by creating a whole new infrastructure ecosystem.

Today, both large and small software companies are deploying thousands of container instances daily. Managing this level of complexity at scale requires advanced tools. Enter Kubernetes. Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. Kubernetes has quickly become the de facto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), supported by major players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

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



Security group (sg-02442b6c1aa42e308 | node_security_group) was created successfully

Details

EC2 > Security Groups > sg-02442b6c1aa42e308 - node_security_group

sg-02442b6c1aa42e308 - node_security_group

Actions

Details

Security group name	Security group ID	Description	VPC ID
node_security_group	sg-02442b6c1aa42e308	security group for nodes	vpc-08e88c85fcc263608
Owner	**Inbound rules count**	**Outbound rules count**	
724772084448	6 Permission entries	1 Permission entry	

Inbound rules | Outbound rules | Tags

Inbound rules (6)

Manage tags | Edit inbound rules

Search

1

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

Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Li

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible

ami-04cdc91e49cb06165 (64-bit (x86)) / ami-02b7539372433cf6b (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).

Summary

Number of instances Info

1

Software Image (AMI)

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

ami-04cdc91e49cb06165

Virtual server type (instance type)

t3.medium

Firewall (security group)

master_group

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

Cancel | Launch instance

Review commands

Instance type Info | Get advice

Instance type

t3.medium
Family: t3 2 vCPU 4 GiB Memory Current generation: true
On-Demand RHEL base pricing: 0.072 USD per Hour
On-Demand Linux base pricing: 0.0432 USD per Hour
On-Demand Windows base pricing: 0.0616 USD per Hour
On-Demand SUSE base pricing: 0.0995 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 before you launch the instance.

Key pair name - *required*

master_ec2_key

Create new key pair

Network settings Info

Edit

Network Info

Summary

Number of instances Info

1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-04cdc91e49cb06165

Virtual server type (instance type)
t3.medium

Firewall (security group)
master_group

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

Cancel

Launch instance

Review commands

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_group sg-00b928c599e4bfcea
VPC: vpc-08e88c85fcc263608

Compare security group rules

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

Configure storage Info

Advanced

1x 8 GiB gp3 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Summary

Number of instances Info

1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-04cdc91e49cb06165

Virtual server type (instance type)
t3.medium

Firewall (security group)
master_group

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

Cancel

Launch instance

Review commands

EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance (i-0d309ff71c5feceb7)

Launch log

Instances (2) Info

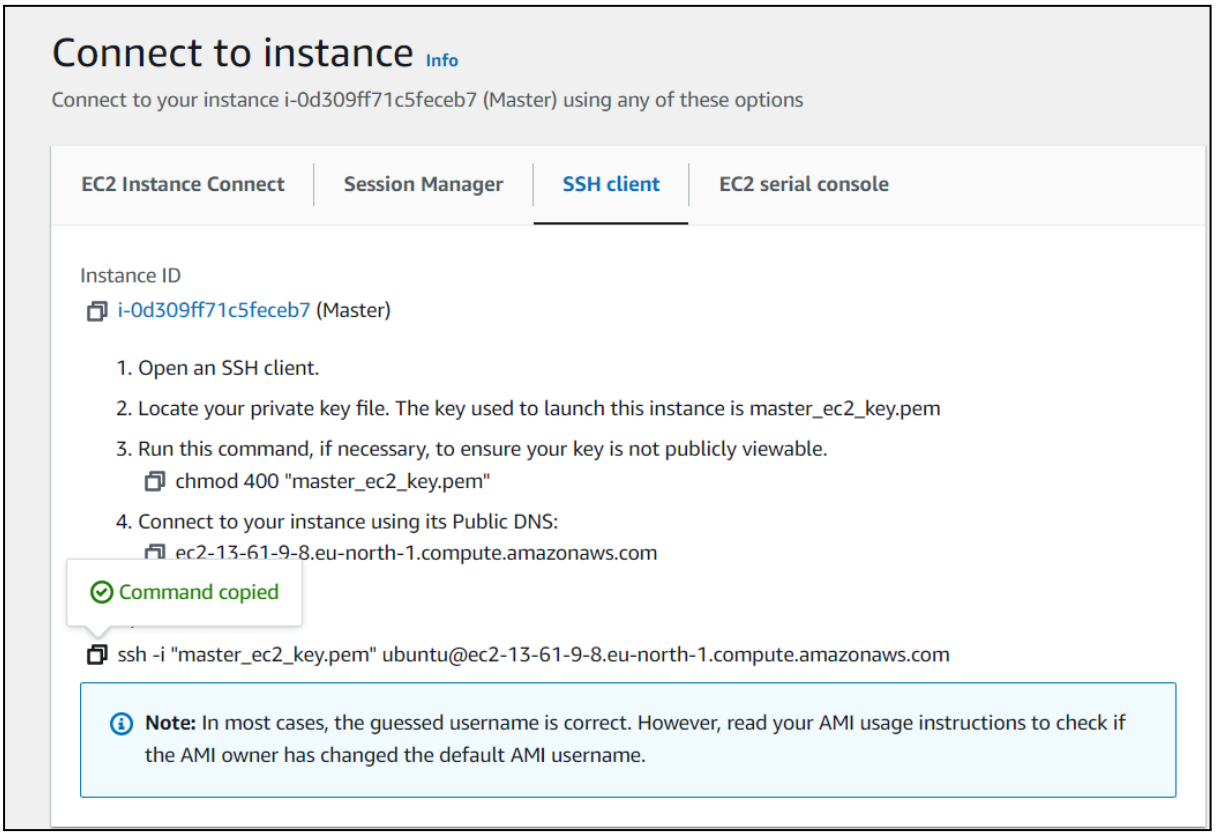
Last updated less than a minute ago

Connect | Instance state | Actions | Launch instances

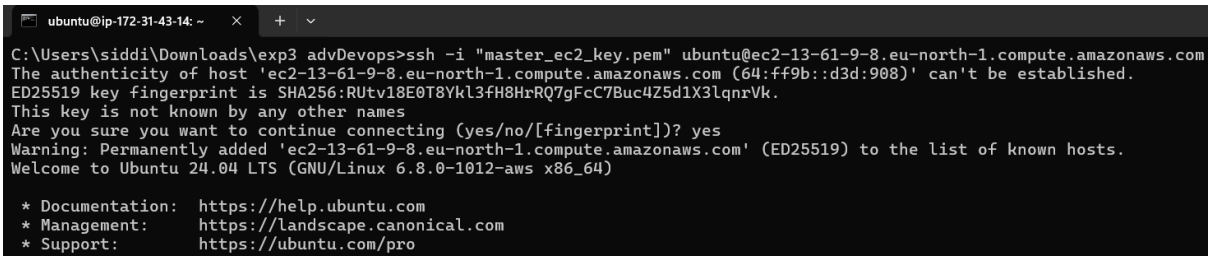
Find Instance by attribute or tag (case-sensitive) | All states

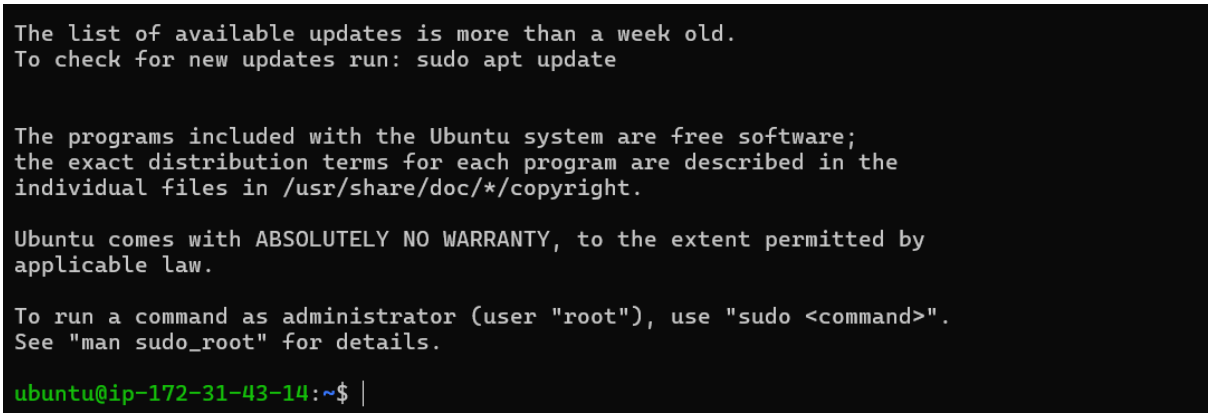
1

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public I
Master	i-0d309ff71c5feceb7	Running	t3.medium	Initializing	View alarms +	eu-north-1b	ec2-13-
Node1	i-0475ac9631a14fc29	Running	t3.medium	Initializing	View alarms +	eu-north-1b	ec2-13-



Step 3: 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





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

```
ubuntu@ip-172-31-43-14:~$ 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-----

mQINBFit2ioBEADhWpZ8/wvZ6hUTiXOwQHXMAlaFHcPH9hAtr4F1y2+OYdbtMuth
lqqwp028AqyY+PRfVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh
38UuLa+z077PxyxQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOXbnq
L4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlANyEVlZzeqo7XKl7
UrwV5inawTSzWNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N
cmhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht
ku4bIQN1X5P07fNWzlgaRL5Z4POXDDZTlIQ/El58j9kp4bnWRCJW0lya+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtl/94VFYVJOleAv8W92KdgDkhTcTD
```

```
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 (6027 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/a
/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-43-14:~$
```

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

```
ubuntu@ip-172-31-43-14:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-north-1.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 section 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 139 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
```

```
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.
ubuntu@ip-172-31-43-14:~$
```

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

```
ubuntu@ip-172-31-43-14:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-43-14:~$
```

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

```
ubuntu@ip-172-31-43-14:~$ 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
ubuntu@ip-172-31-43-14:~$
```

Step 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-172-31-43-14:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31
/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-ke
yring.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
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-43-14:~$
```

error:

```
ubuntu@ip-172-31-43-14:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
sudo mkdir -p /etc/apt/keyrings i
E: Malformed entry 1 in list file /etc/apt/sources.list.d/kubernetes.list (U
RI)
E: The list of sources could not be read.
E: Malformed entry 1 in list file /etc/apt/sources.list.d/kubernetes.list (U
RI)
E: The list of sources could not be read.
E: Malformed entry 1 in list file /etc/apt/sources.list.d/kubernetes.list (U
RI)
E: The list of sources could not be read.
```

Resolving the error:

```
ubuntu@ip-172-31-43-14:~$ sudo mkdir -p /etc/apt/keyrings
ubuntu@ip-172-31-43-14:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31
/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-ke
yring.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N)
 y
ubuntu@ip-172-31-43-14:~$ 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.k8
s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-43-14:~$ sudo apt-get update
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InReleas
e [126 kB]
Hit:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRele
ase
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd
64 Packages [530 kB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/v1.31/deb  InRelease [1186 B]
Get:8 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Tra
nslation-en [128 kB]
Get:9 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/v1.31/deb  Packages [4865 B]
Fetched 791 kB in 1s (1080 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is st
ored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATI
ON section in apt-key(8) for details.
```

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

```
ubuntu@ip-172-31-43-14:~$ 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://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 connt
rack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/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:/s
table:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/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:/s
table:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
Fetched 87.4 MB in 1s (66.9 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) ...
Selecting previously unselected package kubectl.
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) ...
```

```
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.
ubuntu@ip-172-31-43-14:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-43-14:~$
```

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

```
ubuntu@ip-172-31-43-14:~$ 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 requir
ed:
  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 139 not upgraded.
```

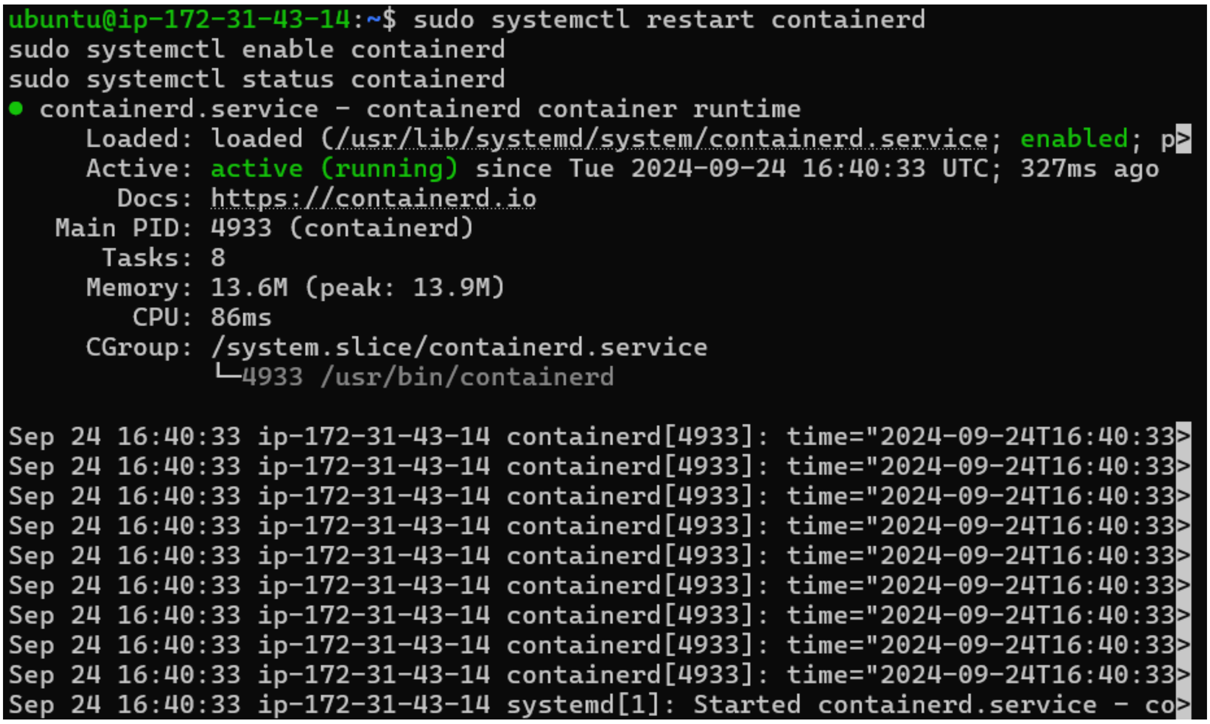
```
Running kernel seems to be up-to-date.

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-172-31-43-14:~$
```

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

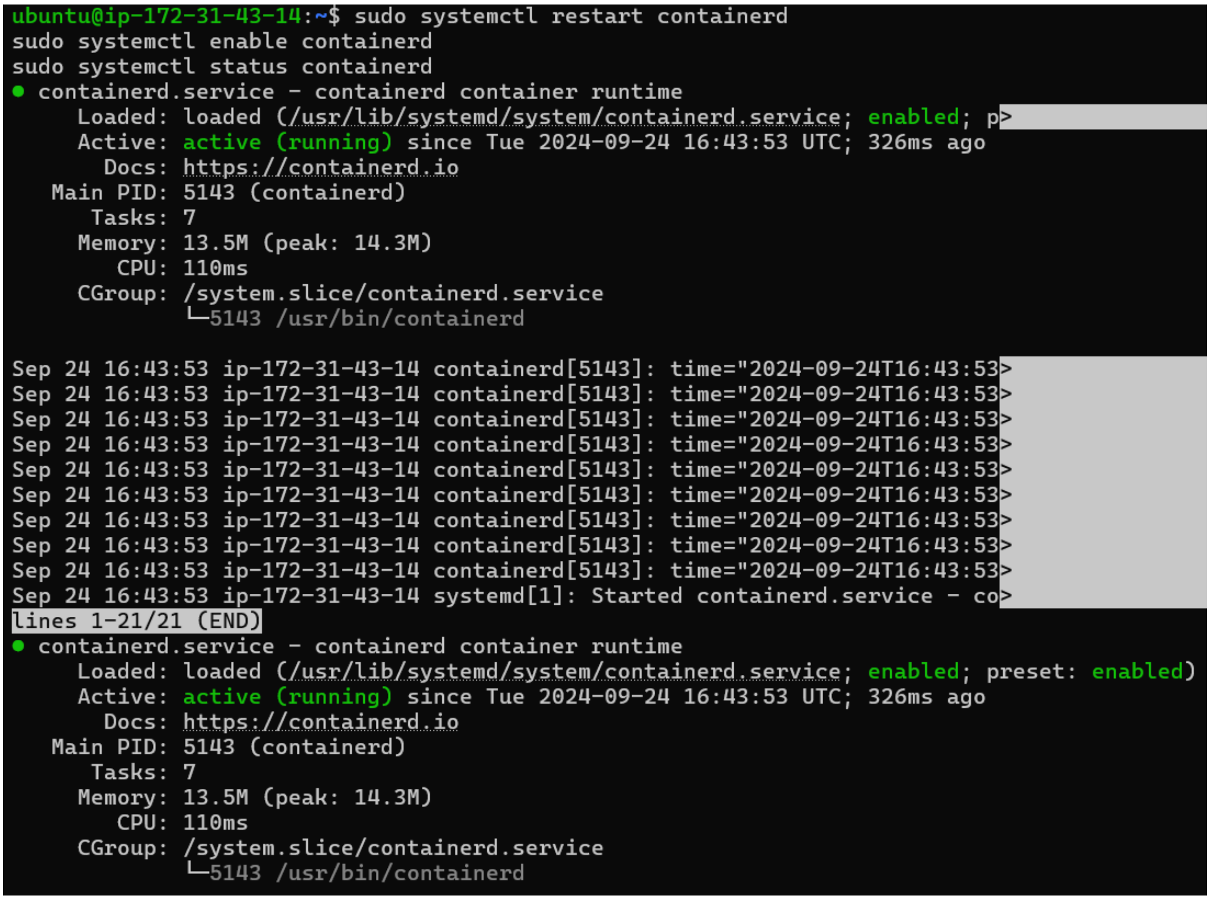
```
ubuntu@ip-172-31-43-14:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

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

```
  [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_ke
yprovider.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
ubuntu@ip-172-31-43-14:~$ |
```

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



```
ubuntu@ip-172-31-43-14:~$ 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; p>
     Active: active (running) since Tue 2024-09-24 16:40:33 UTC; 327ms ago
       Docs: https://containerd.io
   Main PID: 4933 (containerd)
      Tasks: 8
     Memory: 13.6M (peak: 13.9M)
        CPU: 86ms
     CGroup: /system.slice/containerd.service
             └─4933 /usr/bin/containerd

Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 containerd[4933]: time="2024-09-24T16:40:33>
Sep 24 16:40:33 ip-172-31-43-14 systemd[1]: Started containerd.service - co>
```



```
ubuntu@ip-172-31-43-14:~$ 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; p>
     Active: active (running) since Tue 2024-09-24 16:43:53 UTC; 326ms ago
       Docs: https://containerd.io
   Main PID: 5143 (containerd)
      Tasks: 7
     Memory: 13.5M (peak: 14.3M)
        CPU: 110ms
     CGroup: /system.slice/containerd.service
             └─5143 /usr/bin/containerd

Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 containerd[5143]: time="2024-09-24T16:43:53>
Sep 24 16:43:53 ip-172-31-43-14 systemd[1]: Started containerd.service - co>
lines 1-21/21 (END)
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-09-24 16:43:53 UTC; 326ms ago
       Docs: https://containerd.io
   Main PID: 5143 (containerd)
      Tasks: 7
     Memory: 13.5M (peak: 14.3M)
        CPU: 110ms
     CGroup: /system.slice/containerd.service
             └─5143 /usr/bin/containerd
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-43-14:~$ 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 requir
ed:
  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 139 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat
 amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (13.1 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.
ubuntu@ip-172-31-43-14:~$
```

Run on master only

Step 6: Initialize the Kubecluster .

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

```
ubuntu@ip-172-31-43-14:~$ 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 confi
g images pull'
W0924 16:50:39.484946    5597 checks.go:846] detected that the sandbox image
 "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with t
hat used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as
 the CRI sandbox image.
```

```
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 eac
h as root:

kubeadm join 172.31.43.14:6443 --token b84q3o.sj81phrsio2i35r9 \
        --discovery-token-ca-cert-hash sha256:1910a628058fc97f2da7c6121dfc1e
d022fc1d68301f07bddd2afa0f7311c454
ubuntu@ip-172-31-43-14:~$
```

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-172-31-43-14:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-43-14:~$
```

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

```
ubuntu@ip-172-31-43-14:~$ kubectl get nodes
NAME               STATUS     ROLES           AGE     VERSION
ip-172-31-43-14    NotReady   control-plane   2m34s   v1.31.1
ubuntu@ip-172-31-43-14:~$
```

Run on node:

```
ubuntu@ip-172-31-32-65:~$ sudo kubeadm join 172.31.43.14:6443 --token b84q3o
.sj81phrsio2i35r9 \
        --discovery-token-ca-cert-hash sha256:1910a628058fc97f2da7c6121dfc1e
d022fc1d68301f07bddd2afa0f7311c454
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-syst
em get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/conf
ig.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/li
b/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/heal
thz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.634348ms
[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 recei
ved.
* The Kubelet was informed of the new secure connection details.

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

ubuntu@ip-172-31-32-65:~$
```

On master

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

```
ubuntu@ip-172-31-43-14:~$ kubectl get nodes
NAME               STATUS     ROLES           AGE     VERSION
ip-172-31-32-65    NotReady   <none>          2m24s   v1.31.1
ip-172-31-43-14    NotReady   control-plane   48m     v1.31.1
ubuntu@ip-172-31-43-14:~$
```

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-172-31-43-14:~$ kubectl apply -f https://docs.projectcalico.org/ma
nifests/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.projectc
alico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org
 created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcal
ico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.project
calico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projec
tcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projec
tcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.proj
ectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectc
alico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalic
o.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.o
rg created
```

sudo systemctl status kubelet

```
ubuntu@ip-172-31-43-14:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; pres
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Tue 2024-09-24 16:51:06 UTC; 50min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 6271 (kubelet)
      Tasks: 11 (limit: 4586)
     Memory: 34.8M (peak: 35.5M)
        CPU: 55.314s
     CGroup: /system.slice/kubelet.service
             └─6271 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes

Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]:  > pod="kube-system/calico-k
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]: E0924 17:41:09.662382    627
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]:         rpc error: code = Un
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]:         : unknown
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]:  > podSandboxID="55cd262ca32
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]: E0924 17:41:09.662435    627
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]: E0924 17:41:09.662494    627
Sep 24 17:41:09 ip-172-31-43-14 kubelet[6271]: I0924 17:41:09.756328    627
Sep 24 17:41:11 ip-172-31-43-14 kubelet[6271]: I0924 17:41:11.685597    627
Sep 24 17:41:11 ip-172-31-43-14 kubelet[6271]: E0924 17:41:11.685797    627
```

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

```
ubuntu@ip-172-31-43-14:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-32-65   Ready    <none>          4m50s   v1.31.1   172.31.32.65   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-43-14   Ready    control-plane   50m     v1.31.1   172.31.43.14   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-43-14:~$
```

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-172-31-28-117 kubernetes.io/role=Node1

```
ubuntu@ip-172-31-43-14:~$ kubectl label node ip-172-31-32-65 kubernetes.io/r
ole=Node1
node/ip-172-31-32-65 labeled
ubuntu@ip-172-31-43-14:~$
```

```
ubuntu@ip-172-31-43-14:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-32-65   Ready    Node1           7m31s   v1.31.1   172.31.32.65   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-43-14   Ready    control-plane   53m     v1.31.1   172.31.43.14   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-43-14:~$
```

run kubectl get nodes

```
ubuntu@ip-172-31-43-14:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE     VERSION
ip-172-31-32-65   Ready    Node1           8m17s   v1.31.1
ip-172-31-43-14   Ready    control-plane   54m     v1.31.1
ubuntu@ip-172-31-43-14:~$
```