

## 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. Like 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.

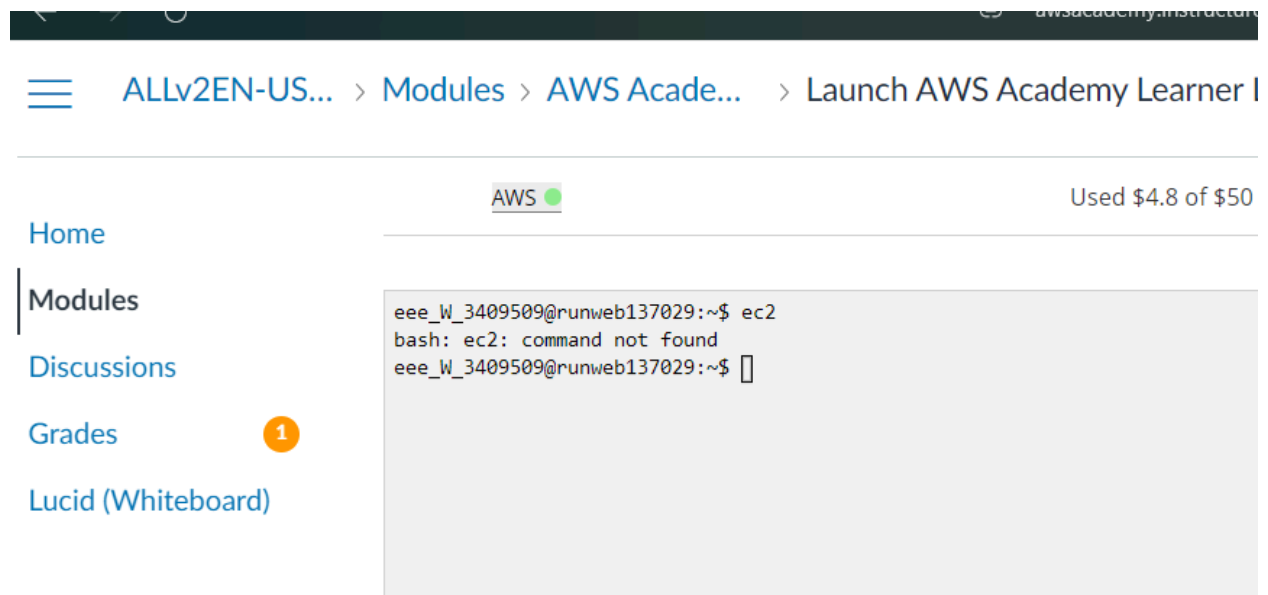
Kubernetes simplifies the deployment and operation of applications in a microservice architecture by providing an abstraction layer over a group of hosts. This allows development teams to deploy their applications while Kubernetes takes care of key tasks, including:

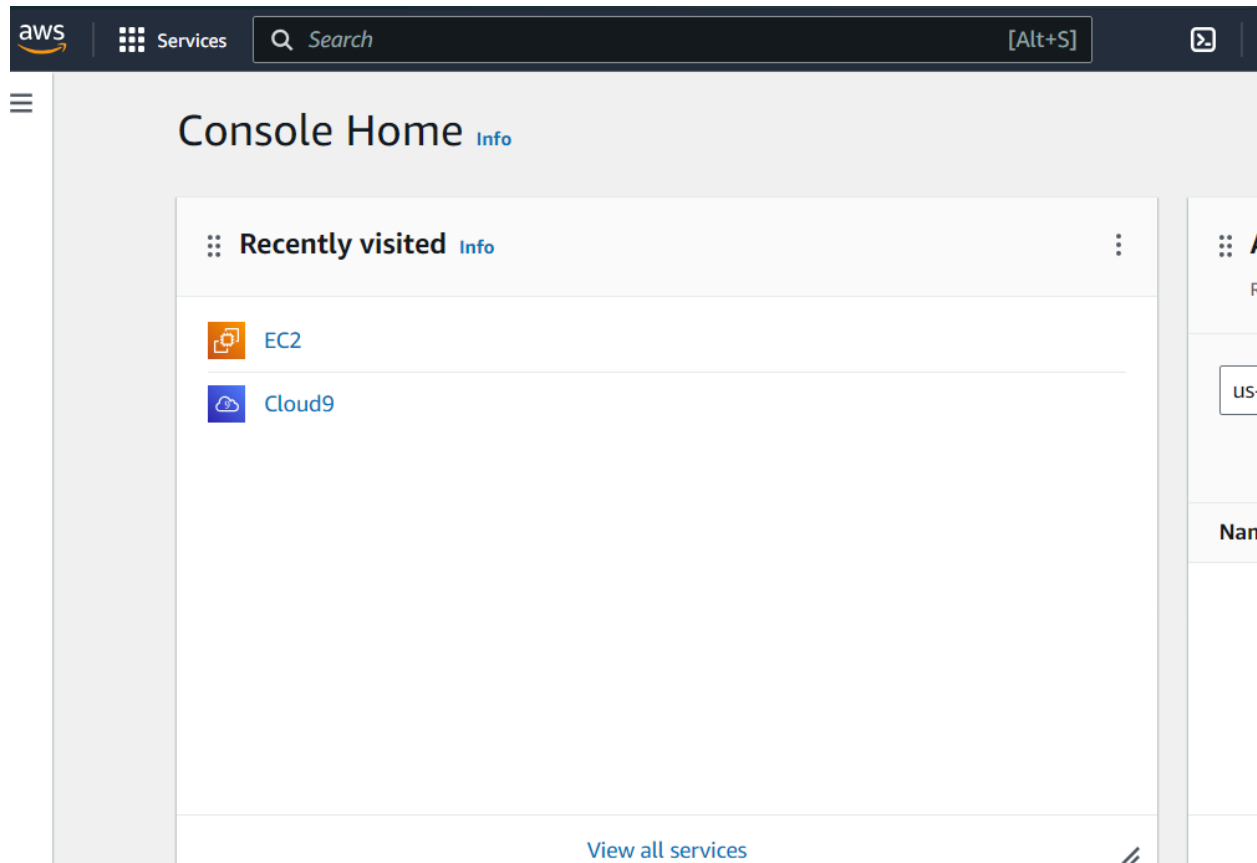
- Managing resource consumption by applications or teams
- Distributing application load evenly across the infrastructure
- Automatically load balancing requests across multiple instances of an application
- Monitoring resource usage to prevent applications from exceeding resource limits and automatically restarting them if needed
- Moving application instances between hosts when resources are low or if a host fails
- Automatically utilizing additional resources when new hosts are added to the cluster
- Facilitating canary deployments and rollbacks with ease.

## Steps:

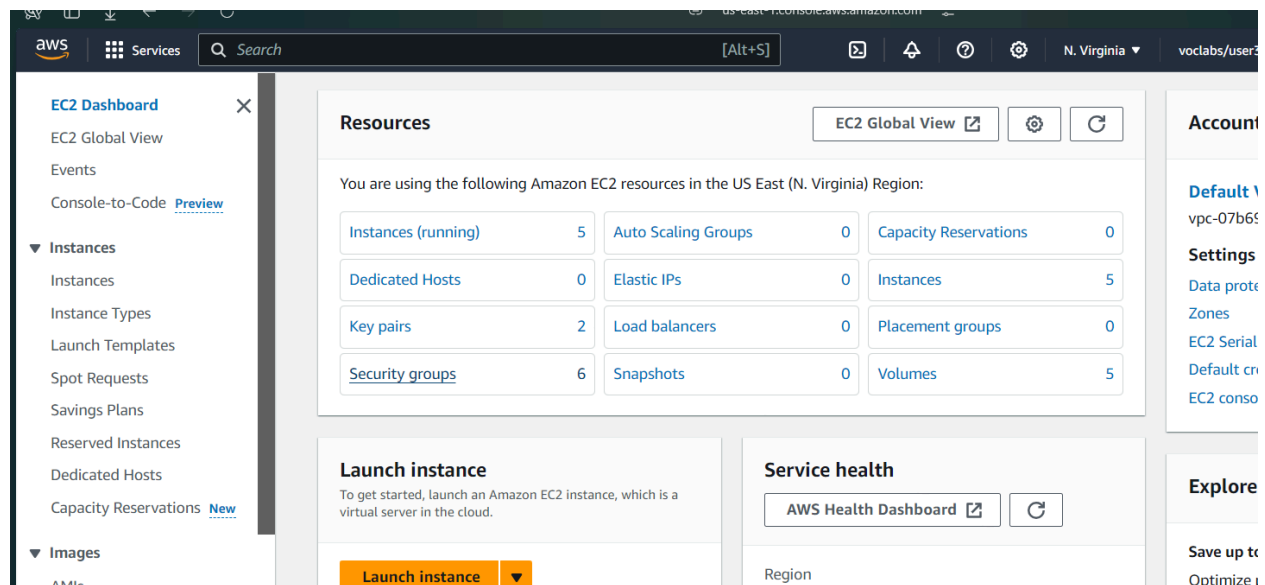
### Set Up the instances of each machine

1. open the aws academy.

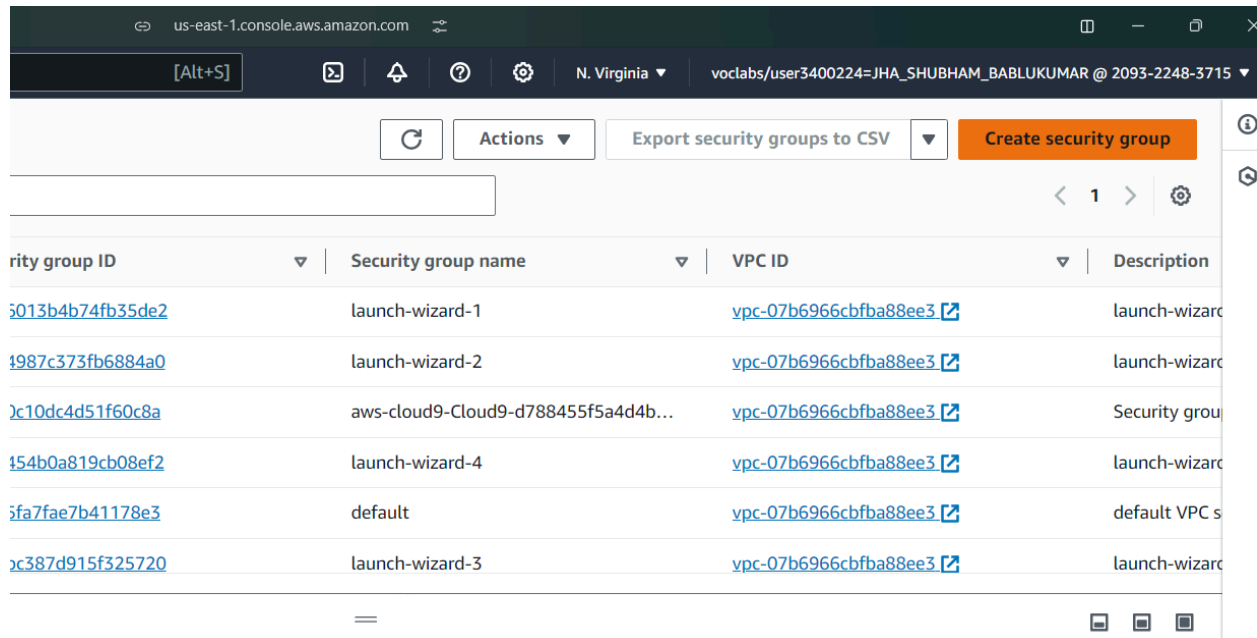




## 2. Click on security groups



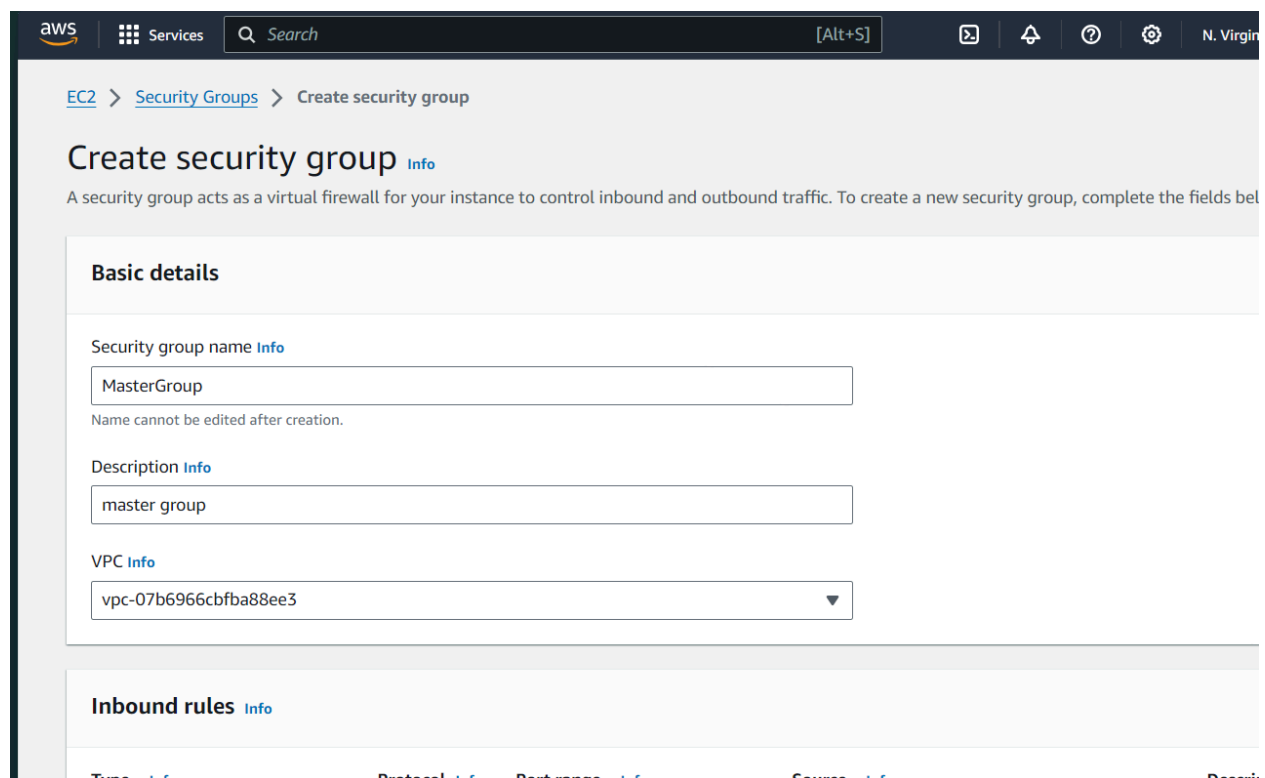
## 3. create two secure groups one for master and other for the two nodes.



The screenshot shows the AWS Management Console for the us-east-1 region. At the top, there are navigation tabs for 'Actions', 'Export security groups to CSV', and a prominent orange 'Create security group' button. Below these is a table listing existing security groups. The table has four columns: 'Security group ID', 'Security group name', 'VPC ID', and 'Description'. The listed security groups include 'launch-wizard-1' through 'launch-wizard-4', 'aws-cloud9-Cloud9-d788455f5a4d4b...', 'default', and 'launch-wizard-3'. Each entry includes a link to the security group's details page.

| Security group ID                | Security group name                 | VPC ID                                | Description    |
|----------------------------------|-------------------------------------|---------------------------------------|----------------|
| <a href="#">s013b4b74fb35de2</a> | launch-wizard-1                     | <a href="#">vpc-07b6966cbfba88ee3</a> | launch-wizard  |
| <a href="#">t987c373fb6884a0</a> | launch-wizard-2                     | <a href="#">vpc-07b6966cbfba88ee3</a> | launch-wizard  |
| <a href="#">c10dc4d51f60c8a</a>  | aws-cloud9-Cloud9-d788455f5a4d4b... | <a href="#">vpc-07b6966cbfba88ee3</a> | Security group |
| <a href="#">t54b0a819cb08ef2</a> | launch-wizard-4                     | <a href="#">vpc-07b6966cbfba88ee3</a> | launch-wizard  |
| <a href="#">tfa7fae7b41178e3</a> | default                             | <a href="#">vpc-07b6966cbfba88ee3</a> | default VPC s  |
| <a href="#">c387d915f325720</a>  | launch-wizard-3                     | <a href="#">vpc-07b6966cbfba88ee3</a> | launch-wizard  |

4. enter details and add inbound rules (I have given MasterGroup for the master group)



The screenshot shows the 'Create security group' page in the AWS Management Console. The breadcrumb navigation indicates the path: EC2 > Security Groups > Create security group. The page title is 'Create security group' with an 'Info' link. A descriptive sentence states: 'A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.' The form is divided into two main sections: 'Basic details' and 'Inbound rules'. The 'Basic details' section contains three fields: 'Security group name' (with a value of 'MasterGroup'), 'Description' (with a value of 'master group'), and 'VPC' (with a dropdown menu showing 'vpc-07b6966cbfba88ee3'). The 'Inbound rules' section is partially visible at the bottom, showing a table header with columns for 'Type', 'Protocol', 'Port range', 'Source', and 'Description'.

**Create security group** [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

**Basic details**

Security group name [Info](#)

MasterGroup

Name cannot be edited after creation.

Description [Info](#)

master group

VPC [Info](#)

vpc-07b6966cbfba88ee3

**Inbound rules** [Info](#)

| Type | Protocol | Port range | Source | Description |
|------|----------|------------|--------|-------------|
|------|----------|------------|--------|-------------|

You have to look for the particular configuration which I did (in the image below)

**Inbound rules** [Info](#)

| Type <a href="#">Info</a> | Protocol <a href="#">Info</a> | Port range <a href="#">Info</a> | Source <a href="#">Info</a> | Description - optional <a href="#">Info</a> |        |
|---------------------------|-------------------------------|---------------------------------|-----------------------------|---|--------|
| HTTP                      | TCP                           | 80                              | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 6443                            | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| All traffic               | All                           | All                             | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 10251                           | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 10252                           | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 10250                           | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| All TCP                   | TCP                           | 0 - 65535                       | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| SSH                       | TCP                           | 22                              | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |

click on create security group below.

now do the same for a node group.

**Inbound rules** [Info](#)

| Type <a href="#">Info</a> | Protocol <a href="#">Info</a> | Port range <a href="#">Info</a> | Source <a href="#">Info</a> | Description - optional <a href="#">Info</a> |        |
|---------------------------|-------------------------------|---------------------------------|-----------------------------|---|--------|
| All traffic               | All                           | All                             | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| SSH                       | TCP                           | 22                              | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 10250                           | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| All TCP                   | TCP                           | 0 - 65535                       | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| Custom TCP                | TCP                           | 30000 - 32767                   | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |
| HTTP                      | TCP                           | 80                              | Anywher...                  | 0.0.0.0/0                                   | Delete |
|                           |                               |                                 |                             | 0.0.0.0/0                                   |        |

Add rule

5. now go to ec2 and launch an instance

The screenshot displays the Amazon EC2 Dashboard. On the left is a navigation sidebar with the following sections:

- EC2 Dashboard** (with a close icon)
- EC2 Global View
- Events
- Console-to-Code [Preview](#)
- Instances** (expanded)
  - Instances
  - Instance Types
  - Launch Templates
  - Spot Requests
  - Savings Plans
  - Reserved Instances
  - Dedicated Hosts
  - Capacity Reservations [New](#)
- Images**
  - AMIs
  - AMI Catalog
- Elastic Block Store**
  - Volumes
  - Snapshots
  - Lifecycle Manager

The main content area is titled **Resources** and states: "You are using the following Amazon EC2 resources in the US East (N. Virginia) Region". It contains a table of resources:

| Resource            | Count |
|---------------------|-------|
| Instances (running) | 5     |
| Dedicated Hosts     | 0     |
| Key pairs           | 2     |
| Security groups     | 8     |

Below the table, there are links for other resources: [Auto Scaling Groups](#), [Elastic IPs](#), [Load balancers](#), and [Snapshots](#).

The **Launch instance** section is highlighted, with the text: "To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud." It features a prominent orange **Launch instance** button, a dropdown arrow, and a **Migrate a server** button with an external link icon.

A note states: "Note: Your instances will launch in the US East (N. Virginia) Region".

The **Instance alarms** section includes a **View in CloudWatch** button with an external link icon.

On the far right, a partial view of the "Server" section is visible, showing "Region: US E", "Status: ✔ T", and "Zor".

add name and set ubuntu:

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

Recents | **Quick Start**

Amazon Linux  
aws


macOS  
Mac

Ubuntu  
ubuntu®

Windows  
Microsoft

Red Hat  
Red Hat

SUSE Linux  
SUSE

  
[Browse more AMIs](#)  
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

### ▼ Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu 18.04 LTS (64-bit) ami-0e86e20dae922

Virtual server type: t2.micro

Firewall (security group): New security group

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

**Free tier:** Includes 750 hours of t2.micro instances in the Region, 30 million public IPv4 addresses per month, 30 million I/Os, and 100 GB of internet data transfer.

create a key if you want

## Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

aws1331

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type



☒ RSA  
RSA encrypted private and public key pair

☐ ED25519  
ED25519 encrypted private and public key pair

Private key file format

☒ .pem  
For use with OpenSSH

☐ .ppk  
For use with PuTTY

 When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#) 

Cancel

Create key pair

If you want you can reuse the key pair generated earlier.



Select the security group for master.

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

Q |

|                          |   |                      |
|--------------------------|---|----------------------|
| <input type="checkbox"/> | launch-wizard-1<br>VPC: vpc-07b6966cbfba88ee3   | sg-06013b4b74fb35de2 |
| <input type="checkbox"/> | MasterGroup<br>VPC: vpc-07b6966cbfba88ee3   | sg-00c39d8526dda67f7 |
| <input type="checkbox"/> | launch-wizard-2<br>VPC: vpc-07b6966cbfba88ee3   | sg-04987c373fb6884a0 |
| <input type="checkbox"/> | aws-cloud9-Cloud9-d788455f5a4d4b4083454091233a80eb-<br>InstanceSecurityGroup-OjiPSymDkJTu<br>VPC: vpc-07b6966cbfba88ee3 | sg-00c10dc4d51f60c8a |
| <input type="checkbox"/> | launch-wizard-4<br>VPC: vpc-07b6966cbfba88ee3   | sg-0454b0a819cb08ef2 |
| <input type="checkbox"/> | default<br>VPC: vpc-07b6966cbfba88ee3   | sg-05fa7fae7b41178e3 |
| <input type="checkbox"/> | NodeGroup   | sg-0c4f15084d343fbdf |

[Compare security group rules](#)

erfaces.

[Advance](#)

or Magnetic

X

the first 0 instance store

then launch:

The screenshot shows the AWS Management Console interface for launching an instance. The 'Configure storage' section is active, showing a root volume of 8 GiB with gp3 storage type. A notification indicates that free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. The 'Advanced details' section is also visible, showing the instance type as t2.micro and the security group as MasterGroup. The 'Launch instance' button is highlighted in orange.

do the same for node instance just select the number of instance as 2.  
and select custom security group as node group.

The screenshot shows the AWS Management Console interface for launching an instance. The 'Network settings' section is active, showing the network as vpc-07b6966cbfa88ee3 and the subnet as No preference. The 'Summary' section is also visible, showing the instance type as t2.micro and the security group as NodeGroup. The 'Launch instance' button is highlighted in orange.

dont give name now. and launch instance.

now go to instances and give name to the blanck ones:

| Instances (1/8) <a href="#">Info</a>               |                     |                      |               |                                |                             |                   |                          |        |  |
|--|---------------------|----------------------|---------------|--------------------------------|-----------------------------|-------------------|--------------------------|--------|--|
| Find Instance by attribute or tag (case-sensitive) |                     |                      |               | All states                     |                             |                   |                          |        |  |
| Name   | Instance ID         | Instance state       | Instance type | Status check                   | Alarm status                | Availability Zone | Public IPv4 DNS          | Public |  |
| <input type="checkbox"/> Master                    | i-0ab175e9c60cc3a23 | <span>Running</span> | t2.micro      | <span>2/2 checks passed</span> | <a href="#">View alarms</a> | us-east-1b        | ec2-184-73-147-45.co...  | 184.7  |  |
| <input type="checkbox"/> node-1                    | i-08ad30b7114767ca2 | <span>Running</span> | t2.micro      | <span>Initializing</span>      | <a href="#">View alarms</a> | us-east-1b        | ec2-107-21-179-161.co... | 107.2  |  |
| <input checked="" type="checkbox"/> node-2         | i-03c70d364fb762af5 | <span>Running</span> | t2.micro      | <span>Initializing</span>      | <a href="#">View alarms</a> | us-east-1b        | ec2-44-210-122-179.co... | 44.21  |  |

6. select master and connect:

| Instances (1/8) <a href="#">Info</a>               |                     |                      |               |                                |                             |                   |                         |        |  |
|--|---------------------|----------------------|---------------|--------------------------------|-----------------------------|-------------------|-------------------------|--------|--|
| Find Instance by attribute or tag (case-sensitive) |                     |                      |               | All states                     |                             |                   |                         |        |  |
| Name   | Instance ID         | Instance state       | Instance type | Status check                   | Alarm status                | Availability Zone | Public IPv4 DNS         | Public |  |
| <input checked="" type="checkbox"/> Master         | i-0ab175e9c60cc3a23 | <span>Running</span> | t2.micro      | <span>2/2 checks passed</span> | <a href="#">View alarms</a> | us-east-1b        | ec2-184-73-147-45.co... |        |  |

click on ssh client:

copy the command below the SSH client session

[EC2](#) > [Instances](#) > [i-0ab175e9c60cc3a23](#) > [Connect to instance](#)

## Connect to instance [Info](#)

Connect to your instance i-0ab175e9c60cc3a23 (Master) using any of these options

[EC2 Instance Connect](#)
[Session Manager](#)
[SSH client](#)
[EC2 serial console](#)

Instance ID

[i-0ab175e9c60cc3a23](#) (Master)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is aws1331.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
 `chmod 400 "aws1331.pem"`
4. Connect to your instance using its Public DNS:  
 `ec2-184-73-147-45.compute-1.amazonaws.com`

Command copied

`ssh -i "aws1331.pem" ubuntu@ec2-184-73-147-45.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.

Cancel

## 7. Enter the copied command to a terminal window.

```

Microsoft Windows [Version 10.0.22631.4112]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>ssh -i "C:\Users\Lenovo\Downloads\aws1331.pem" ubuntu@ec2-34-203-217-53.compute-1.amazonaws.com
The authenticity of host 'ec2-34-203-217-53.compute-1.amazonaws.com (34.203.217.53)' can't be established.
ED25519 key fingerprint is SHA256:3onu4BDyF+uS+Fwt16U1L99+0SyVYZbTNPiWoY074Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? |

```

It would prompt whether we want to continue connecting. type yes.

```

System information as of Fri Sep 27 15:25:58 UTC 2024

System load:  0.0          Processes:      104
Usage of /:   22.8% of 6.71GB Users logged in: 0
Memory usage: 19%         IPv4 address for enX0: 172.31.87.211
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

Updates can be applied immediately.

Disable 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.

ubuntu@ip-172-31-87-211:~$ |

```

The step is similar for node 1 and node 2 instances too. Just use different terminal windows.

node 1:

EC2 > Instances > i-08ad30b7114767ca2 > Connect to instance

### Connect to instance Info

Connect to your instance i-08ad30b7114767ca2 (node-1) using any of these options

EC2 Instance Connect

Session Manager

**SSH client**

EC2 serial console

Instance ID  
i-08ad30b7114767ca2 (node-1)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is aws1331.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
`chmod 400 "aws1331.pem"`
4. Connect to your instance using its Public DNS:  
`ssh -i "aws1331.pem" ubuntu@ec2-3-88-249-77.compute-1.amazonaws.com`

Command copied

`ssh -i "aws1331.pem" ubuntu@ec2-3-88-249-77.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.

```

System load:  0.0          Processes:      104
Usage of /:   22.8% of 6.71GB Users logged in: 0
Memory usage: 19%         IPv4 address for enX0: 172.31.89.24
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

Updates can be applied immediately.

Disable 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.

ubuntu@ip-172-31-89-24:~$ |

```

node 2:

Connect to your instance i-03c70d364fb762af5 (node-2) using any of these options

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

Instance ID  
i-03c70d364fb762af5 (node-2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is aws1331.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
`chmod 400 "aws1331.pem"`
4. Connect to your instance using its Public DNS:  
ec2-35-173-124-11.compute-1.amazonaws.com

Command copied

`ssh -i "aws1331.pem" ubuntu@ec2-35-173-124-11.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.

```
System load:  0.08      Processes:      104
Usage of /:   22.8% of 6.71GB  Users logged in:  0
Memory usage: 19%      IPv4 address for enX0: 172.31.88.60
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>".
See "man sudo_root" for details.

ubuntu@ip-172-31-88-60:~$ |
```

8. From now on run the commands on all the 3 terminals unless instructed otherwise.

and the images (screen shots) will only be of master unless stated otherwise.

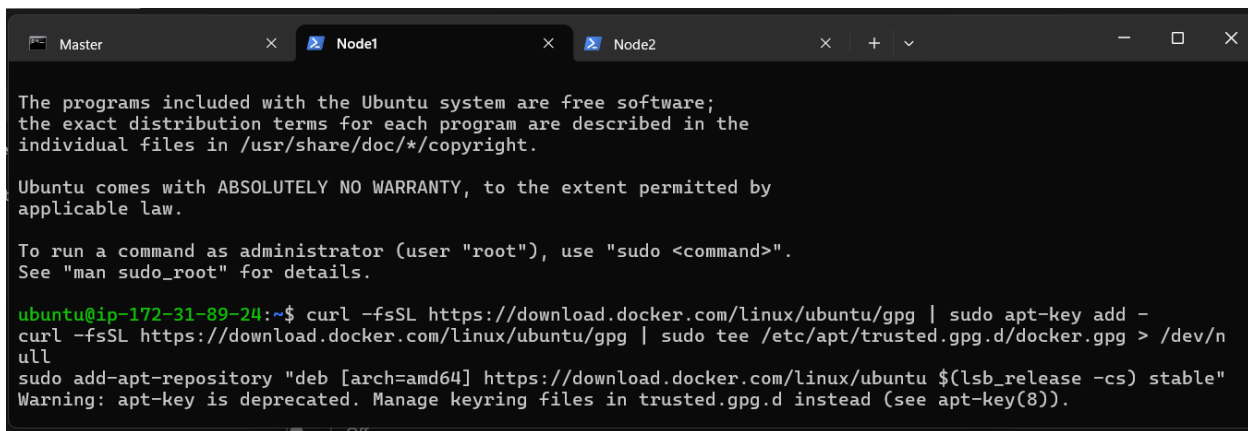
**curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**

**curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee  
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null**

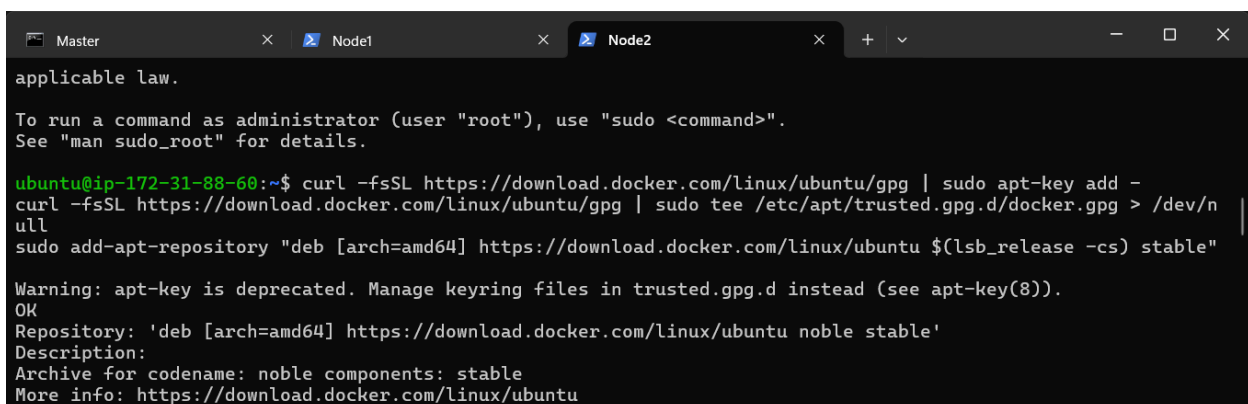
**sudo add-apt-repository "deb [arch=amd64] <https://download.docker.com/linux/ubuntu> \$(lsb\_release -cs) stable"**

```
ubuntu@ip-172-31-87-211:~$ 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
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 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
```

now for node1 and node2:



The image shows a terminal window titled 'Node1' with a dark background. It displays the same sequence of commands and output as the first terminal block, including the curl command to fetch the Docker GPG key, the tee command to save it, and the add-apt-repository command to add the Docker repository for the 'noble' codename. The output shows the repository being added and the sources.list.d file being updated.



The image shows a terminal window titled 'Node2' with a dark background. It displays the same sequence of commands and output as the first terminal block, including the curl command to fetch the Docker GPG key, the tee command to save it, and the add-apt-repository command to add the Docker repository for the 'noble' codename. The output shows the repository being added and the sources.list.d file being updated.

**sudo apt-get update**

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

```
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-172-31-87-211:~$ |
```

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-87-211:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
ubuntu@ip-172-31-87-211:~$ |
```

**sudo systemctl enable docker**

**sudo systemctl daemon-reload**

**sudo systemctl restart docker**

```
Master x Node1 x Node2 x + v - □ x
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-87-211:~$ 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-172-31-87-211:~$ |
```

9. 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
```

```
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-172-31-87-211:~$ 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
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/
ubuntu@ip-172-31-87-211:~$
```

Run the commands:

```
sudo apt-get update
```

```
sudo apt-get install -y kubelet kubeadm kubectl
```

```
sudo apt-mark hold kubelet kubeadm kubectl
```

```
ubuntu@ip-172-31-87-211:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
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]
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
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]
```

```
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-172-31-87-211:~$ 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
  pigz slirp4netns
Use 'sudo apt autoremove' to remove them.

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-172-31-87-211:~$ |
```

**sudo mkdir -p /etc/containerd**

**sudo containerd config default | sudo tee /etc/containerd/config.toml**

```
Master x Node1 x Node2 x + v
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-172-31-87-211:~$ 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 = ""
```

```
[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-172-31-87-211:~$ |
```

**sudo systemctl restart containerd**

**sudo systemctl enable containerd**

**sudo systemctl status containerd**

```
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133071347Z" level=info msg=serve
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133103323Z" level=info msg=serve
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133174859Z" level=info msg="Sta
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133199320Z" level=info msg="Sta
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133236780Z" level=info msg="Sta
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133244763Z" level=info msg="Sta
Sep 27 16:24:28 ip-172-31-87-211 containerd[4900]: time="2024-09-27T16:24:28.133252776Z" level=info msg="Sta
ubuntu@ip-172-31-87-211:~$ |
```

exit with ctrl+c.

**sudo apt-get install -y socat**

```
ubuntu@ip-172-31-87-211:~$ 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
  pigz slurp4netns
Use 'sudo apt autoremove' to remove them.
```

Run the following command in master only:

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

if it gives error use:

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=NumCPU,Mem
```

```
To see the stack trace of this error execute with --v=5 or higher
ubuntu@ip-172-31-87-211:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=NumCPU,Mem
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (957 MB) is less than the minimum 1700 MB
[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'
W0927 16:47:12.068193      6025 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.
```

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 172.31.87.211:6443 --token lhbvbx.kzlxu87vmv8hvx4o \
--discovery-token-ca-cert-hash sha256:1cef7709c45a42691a2ff0e44e3acf7f0e214fec7f4f822bb6818f3cfd24ea43
ubuntu@ip-172-31-87-211:~$
```

## Token and ca

**Note:** copy the text after kubeadm that you see at the later part like below:

```
kubeadm join 172.31.87.211:6443 --token lhbvbx.kzlxu87vmv8hvx4o \
```

```
--discovery-token-ca-cert-hash
```

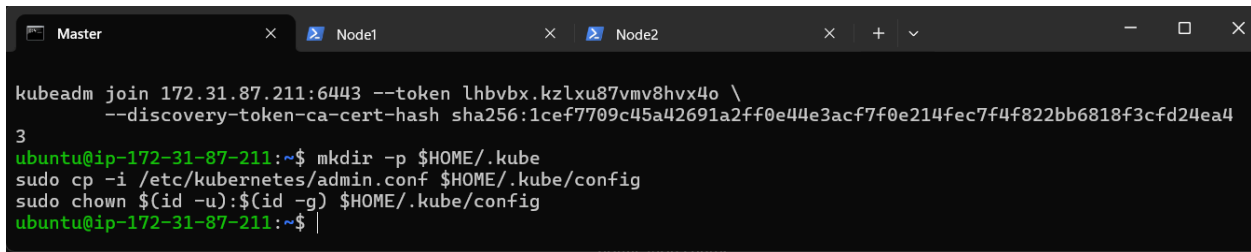
```
sha256:1cef7709c45a42691a2ff0e44e3acf7f0e214fec7f4f822bb6818f3cfd24ea43
```

Run this command on master

```
mkdir -p $HOME/.kube
```

```
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```

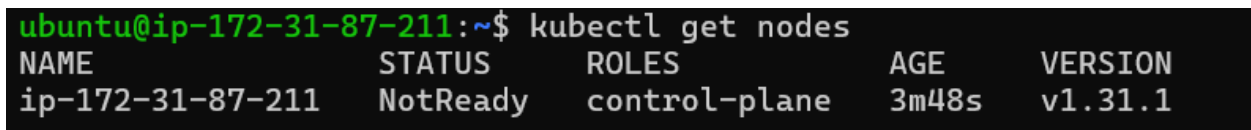
```
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```



```
kubeadm join 172.31.87.211:6443 --token lhbvbx.kzlxu87vmv8hvx4o \
--discovery-token-ca-cert-hash sha256:1cef7709c45a42691a2ff0e44e3acf7f0e214fec7f4f822bb6818f3cfd24ea4
3
ubuntu@ip-172-31-87-211:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-87-211:~$
```

Now Run the command **kubectl get nodes** to see the nodes before executing Join

command on nodes.



```
ubuntu@ip-172-31-87-211:~$ kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
ip-172-31-87-211    NotReady control-plane 3m48s v1.31.1
```

Now paste the token and ca that I asked to copy earlier, on both the nodes.

use sudo before them.

it would be something like:

```
sudo kubeadm join <your-master-node-ip>:6443 --token <your-token>
--discovery-token-ca-cert-hash sha256:<your-ca-cert-hash>
```

(it has placeholders)

Node1:

```
ubuntu@ip-172-31-89-24:~$ sudo kubeadm join 172.31.87.211:6443 --token lhbvbx.kzlxu87vmv8hvx4o \
--discovery-token-ca-cert-hash sha256:1cef7709c45a42691a2ff0e44e3acf7f0e214fec7f4f822bb6818f3cfd24ea4
3
[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
```

Node2:

```
ubuntu@ip-172-31-88-60:~$ sudo kubeadm join 172.31.87.211:6443 --token lhbvbx.kzlxu87vmv8hvx4o \
--discovery-token-ca-cert-hash sha256:1cef7709c45a42691a2ff0e44e3acf7f0e214fec7f4f822bb6818f3cfd24ea4
3
[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 1.00236733s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap
```

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

```
ubuntu@ip-172-31-87-211:~$ kubectl get nodes
NAME                STATUS    ROLES    AGE     VERSION
ip-172-31-87-211    NotReady control-plane 7m35s    v1.31.1
ip-172-31-88-60     NotReady <none>      12s      v1.31.1
ip-172-31-89-24     NotReady <none>      32s      v1.31.1
ubuntu@ip-172-31-87-211:~$
```

**Step 10:** Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

paste this command on master terminal. (and the following commands will be based on master unless states otherwise.

**kubectl apply -f <https://docs.projectcalico.org/manifests/calico.yaml>**

**sudo systemctl status kubelet**

again use ctrl+c to exit.

```

Master  x  Node1  x  Node2  x  +  v  -  □  x
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 Fri 2024-09-27 16:47:34 UTC; 17min ago
Docs: https://kubernetes.io/docs/
Main PID: 6553 (kubelet)
Tasks: 9 (limit: 1130)
Memory: 72.2M (peak: 74.3M)
CPU: 12.057s
CGroup: /system.slice/kubelet.service
└─6553 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubecon>
Sep 27 17:04:26 ip-172-31-87-211 kubelet[6553]: E0927 17:04:26.639597 6553 pod_workers.go:1301] "Error sy>
Sep 27 17:04:29 ip-172-31-87-211 kubelet[6553]: I0927 17:04:29.984518 6553 pod_container_deletor.go:80] ">
Sep 27 17:04:29 ip-172-31-87-211 kubelet[6553]: I0927 17:04:29.987738 6553 scope.go:117] "RemoveContainer>
lines 1-16
ubuntu@ip-172-31-87-211:~$ |

```

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

```

lines 1-16
ubuntu@ip-172-31-87-211:~$ kubecttl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE
KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-87-211 Ready control-plane 18m v1.31.1 172.31.87.211 <none> Ubuntu 24.04 LTS
6.8.0-1012-aws containerd://1.7.12
ip-172-31-88-60 Ready <none> 11m v1.31.1 172.31.88.60 <none> Ubuntu 24.04 LTS
6.8.0-1012-aws containerd://1.7.12
ip-172-31-89-24 Ready <none> 11m v1.31.1 172.31.89.24 <none> Ubuntu 24.04 LTS
6.8.0-1012-aws containerd://1.7.12
ubuntu@ip-172-31-87-211:~$ |

```

Now to Rename run this command

Syntax: **kubecttl label node <node-ip> [kubernetes.io/role=worker](https://kubernetes.io/role=worker)**

examples:

Rename to Node 1: **kubecttl label node ip-<node1ip> kubernetes.io/role=Node1**

Rename to Node 2: **kubecttl label node ip-<node2ip> kubernetes.io/role=Node2**

```

ubuntu@ip-172-31-87-211:~$ kubecttl label node ip-172-31-88-60 kubernetes.io/role=Node2
node/ip-172-31-88-60 labeled
ubuntu@ip-172-31-87-211:~$ kubecttl label node ip-172-31-89-24 kubernetes.io/role=Node1
node/ip-172-31-89-24 labeled
ubuntu@ip-172-31-87-211:~$ |

```

**Step 11: Run command **kubecttl get nodes -o wide** . And Hence we can see we have Successfully connected Node 1 and Node 2 to the Master.**

```
node/ip-172-31-89-24 labeled
ubuntu@ip-172-31-87-211:~$ kubectl get nodes -o wide
NAME                                STATUS    ROLES    AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE
KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-87-211  Ready     control-plane  24m   v1.31.1   172.31.87.211  <none>        Ubuntu 24.04 LTS
6.8.0-1012-aws   containerd://1.7.12
ip-172-31-88-60  Ready     Node2      17m   v1.31.1   172.31.88.60  <none>        Ubuntu 24.04 LTS
6.8.0-1012-aws   containerd://1.7.12
ip-172-31-89-24  Ready     Node1      17m   v1.31.1   172.31.89.24  <none>        Ubuntu 24.04 LTS
6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-87-211:~$
```

## Conclusion:

In this Advanced DevOps Lab experiment, we began by setting up three EC2 Ubuntu instances on AWS, designating one as the Master node and the others as Worker nodes.

We then installed Docker and Kubernetes on all instances, ensuring Docker was properly configured.

The Kubernetes cluster was initialized on the Master node, and the Flannel networking plugin was applied to facilitate communication between nodes.

Finally, we joined the Worker nodes to the cluster using the provided token and hash, resulting in a fully operational Kubernetes cluster ready for managing and scaling containerized applications.