

Case Study 02

Kubernetes Application Deployment

Name:Sarthak Harade

Class: D15B

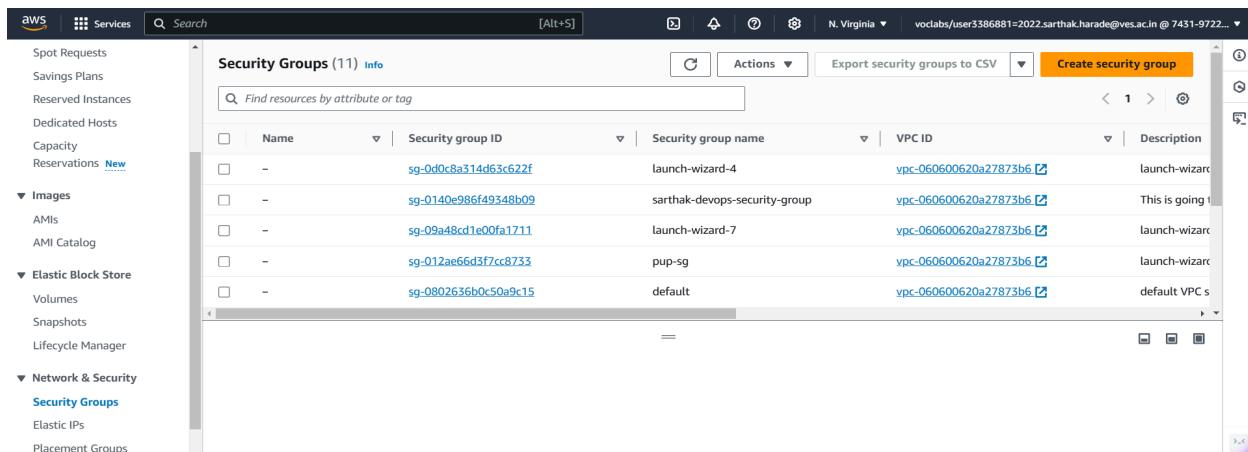
Roll no.: 22

-
- Concepts Used: Kubernetes, AWS Cloud9 IDE, and Kubectl.
 - Problem Statement: "Set up a Kubernetes cluster on AWS using the Cloud9 IDE. Deploy a sample application using kubectl and ensure it runs successfully."
 - Tasks:
 - Install and configure kubectl using AWS Cloud9 IDE.
 - Deploy a sample application (like a simple Nginx server) on the Kubernetes cluster.
 - Verify the application deployment by accessing it through a NodePort or LoadBalancer.

SOLUTION

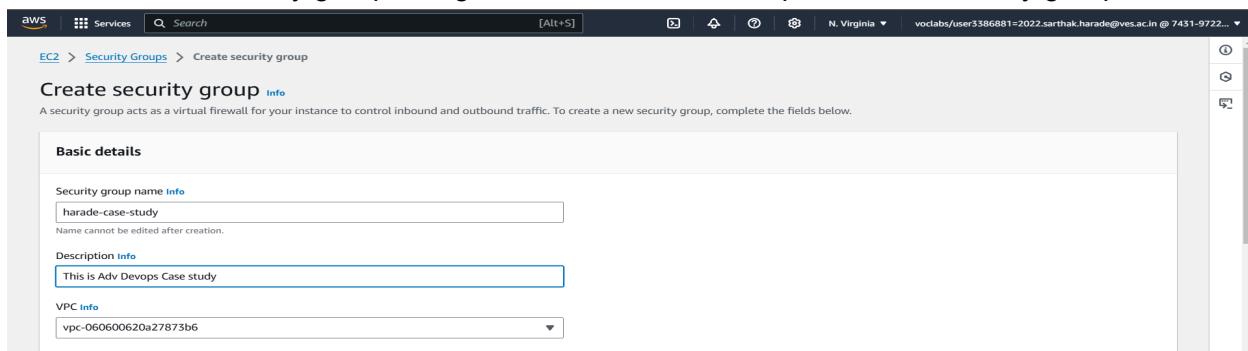
Ensure instances have the appropriate security groups allowing traffic on ports 6443 (Kubernetes API),10250 (kubelet), and other necessary ports like 80 (HTTP) and 443 (HTTPS) for LoadBalancer.

Create Security Groups for Instances Log in to your AWS account and navigate to Security Groups (in EC2).



The screenshot shows the AWS EC2 Security Groups page. The left sidebar lists services like Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, and Security Groups. The main pane displays a table titled 'Security Groups (11) Info' with columns: Name, Security group ID, Security group name, VPC ID, and Description. The table lists several security groups, including 'launch-wizard-4', 'sarthal-devops-security-group', 'launch-wizard-7', 'pup-sg', and 'default'. Each row has a 'Create security group' button at the top right.

Click on create security group and give the name and description to the security group.



The screenshot shows the 'Create security group' wizard. Step 1: Basic details. It asks for a security group name ('harade-case-study') and a description ('This is Adv Devops Case study'). A note says '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.' Below these fields is a 'VPC info' dropdown set to 'vpc-060600620a27873b6'.

Add the following inbound rules to the security group

The screenshot shows the AWS Management Console interface for managing security groups. The top section displays a list of inbound rules for a specific security group. The rules include:

- HTTP (TCP, port 80) from Anywhere to 0.0.0.0/0
- All traffic (All, All) from Anywhere to 0.0.0.0/0
- Custom TCP (TCP, port 6443) from Anywhere to 0.0.0.0/0
- Custom TCP (TCP, port 10251) from Anywhere to 0.0.0.0/0
- Custom TCP (TCP, port 10250) from Anywhere to 0.0.0.0/0
- All TCP (TCP, port range 0 - 65535) from Anywhere to 0.0.0.0/0
- Custom TCP (TCP, port 10252) from Anywhere to 0.0.0.0/0
- SSH (TCP, port 22) from Anywhere to 0.0.0.0/0

A warning message at the bottom states: "⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only." A "Delete" button is visible for each rule.

The screenshot shows the AWS Management Console interface for managing security groups. The top section displays a list of outbound rules for a specific security group. The rules include:

- All traffic (All, All) to Custom destination 0.0.0.0/0
- HTTP (TCP, port 80) to Anywhere 0.0.0.0/0
- HTTPS (TCP, port 443) to Anywhere 0.0.0.0/0

A warning message at the bottom states: "⚠ Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses." A "Delete" button is visible for each rule.

Create a security group.

The screenshot shows the AWS Management Console interface for EC2 Security Groups. A success message indicates that the security group "sg-0a6d2077dd2d29a93 | harade-case-study" was created successfully. The details page for this security group shows the following information:

- Details** tab selected.
- Security group name:** harade-case-study
- Security group ID:** sg-0a6d2077dd2d29a93
- Description:** This is Adv Devops Case study
- VPC ID:** vpc-060600620a27873b6
- Inbound rules count:** 8 Permission entries
- Outbound rules count:** 3 Permission entries

Navigation links include EC2 > Security Groups > sg-0a6d2077dd2d29a93 - harade-case-study. Action buttons include "Actions" and "Edit".

Similarly, create one more security group (for Node).

The screenshot shows the 'Create security group' page in the AWS EC2 console. The 'Basic details' section contains the following information:

- Security group name**: harade-case-study-node
- Description**: This is for node
- VPC**: vpc-060600620a27873b6

Add the inbound rules for Node security group.

The screenshot shows the 'Inbound rules' configuration page in the AWS EC2 console. It displays the following rules:

Type	Protocol	Port range	Source	Description - optional
All traffic	All	All	Anyw... 0.0.0.0/0	0.0.0/0 X
SSH	TCP	22	Anyw... 0.0.0.0/0	0.0.0/0 X
Custom TCP	TCP	10250	Anyw... 0.0.0.0/0	0.0.0/0 X
All TCP	TCP	0 - 65535	Anyw... 0.0.0.0/0	0.0.0/0 X
HTTP	TCP	80	Anyw... 0.0.0.0/0	0.0.0/0 X
Custom TCP	TCP	30000 - 32676	Anyw... 0.0.0.0/0	0.0.0/0 X

An 'Add rule' button is located at the bottom left.

The screenshot shows the 'Security Groups' page in the AWS EC2 console. A success message is displayed: 'Security group (sg-0900c9044ab80cff5 | harade-case-study-node) was created successfully'. The details for the security group 'sg-0900c9044ab80cff5 - harade-case-study-node' are shown:

Details			
Security group name: harade-case-study-node	Security group ID: sg-0900c9044ab80cff5	Description: This is for node	VPC ID: vpc-060600620a27873b6
Owner: 743197227673	Inbound rules count: 6 Permission entries	Outbound rules count: 1 Permission entry	

The 'Inbound rules' tab is selected at the bottom.

Launch EC2 Instances

Navigate to EC2.

The screenshot shows the AWS EC2 Dashboard. On the left, there's a sidebar with 'Instances' selected, showing sub-options like 'Instances', 'Instance Types', 'Launch Templates', etc. The main area is titled 'Instances Info' with a search bar and filters for 'Name', 'Instance ID', 'Instance state', 'Instance type', 'Status check', 'Alarm status', 'Availability Zone', and 'Public IPv4'. A message says 'No instances' and 'You do not have any instances in this region.' At the bottom is a large orange 'Launch instances' button.

MASTER

Launch a new EC2 instance for master with the following settings.

This screenshot shows the 'Launch an instance' wizard. Step 1: 'Name and tags' shows a 'Name' field with 'harade-master'. Step 2: 'Summary' shows 'Number of instances' set to 1, 'Software Image (AMI)' as 'Amazon Linux 2023.6.2...', 'Virtual server type (instance type)' as 't2.micro', and 'Firewall (security group)' as 'New security group'.

This screenshot shows the 'Amazon Machine Image (AMI)' selection screen. It lists several AMI options: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and a 'Browse more AMIs' section. A specific AMI for 'Ubuntu Server 24.04 LTS (HVM), SSD Volume Type' is selected. The summary on the right shows the instance configuration: 'Number of instances' set to 1, 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...', 'Virtual server type (instance type)' as 't2.micro', 'Firewall (security group)' as 'New security group', and 'Storage (volumes)' as '1 volume(s) - 8 GiB'. A tooltip for the free tier is visible.

Generate the .pem file (or key-value pair) once while configuring the master ec2

The screenshot shows the 'Create key pair' dialog box overlaid on the AWS CloudFormation console. In the 'Key pair name' field, 'harade-case-study' is entered. Under 'Key pair type', 'RSA' is selected. A note at the bottom of the dialog box states: '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.' A yellow warning icon is present next to the note. At the bottom right of the dialog box is a 'Create key pair' button.

Keep the instance type as t2.medium

The screenshot shows the 'Launch instance' dialog box. Under 'Instance type', 't2.medium' is selected. In the 'Key pair (login)' section, 'harade-case-study' is chosen from the dropdown. On the right side of the dialog box, there is a 'Summary' section with fields for 'Number of instances' (set to 1), 'Software Image (AMI)', 'Virtual server type (instance type)' (set to t2.medium), and 'Storage (volumes)'. A yellow 'Free tier' callout box is visible, stating: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance'. At the bottom right is a 'Launch instance' button.

Select the existing security group under the network setting tab in which you have to select the group of master which was made in the beginning of the experiment.

The screenshot shows the 'Network settings' dialog box. Under 'Firewall (security groups)', the 'Select existing security group' option is selected. A dropdown menu shows 'harade-case-study' selected. In the 'Common security groups' section, 'harade-case-study sg-0a6d2077dd2d29a93' is listed. A yellow 'Free tier' callout box is visible, stating: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance'. At the bottom right is a 'Launch instance' button.

Successfully launched the ec2 instance.

The screenshot shows the AWS EC2 Instances page. At the top, there is a green success message: "Success Successfully initiated launch of instance (i-0a1e468afe5c3aecc)". Below this, the main table displays one instance: "harade-master" (Instance ID: i-0a1e468afe5c3aecc), which is "Running" (Status check: Initializing) and has an "t2.medium" instance type. The instance is located in the "us-east-1c" Availability Zone and has a public IP of "ec2-54-91-".

NODE

Similarly, launch 2 new ec2 instances for nodes.

The screenshot shows the "Launch an instance" wizard. In the "Name and tags" step, the name "harade-node1" is entered. In the "Number of instances" field, the value "2" is selected. In the "Software Image (AMI)" step, the "Amazon Linux 2023 AMI 2023.6.2..." option is chosen. In the "Virtual server type (instance type)" step, the "t2.micro" instance type is selected. A tooltip for the "Free tier" indicates it includes 750 hours of t2.micro usage. Finally, in the summary step, the "Launch instance" button is visible.

Keep the instance type of node as t2.medium and select the same key-value pair which was created during the master configuration.

Instance type

t2.medium

Family: t2 - 2 vCPU - 4 GiB Memory - Current generation: true

On-Demand Ubuntu Pro base pricing: 0.6360 USD per Hour

On-Demand Linux base pricing: 0.0454 USD per Hour

On-Demand RHEL base pricing: 0.0752 USD per Hour

On-Demand Windows base pricing: 0.0644 USD per Hour

On-Demand SUSE base pricing: 0.1464 USD per Hour

All generations

Compare instance types

Key pair (login)

Key pair name - required

harade-case-study

Create new key pair

Summary

Number of instances

2

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI)

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

ami-0866a3c8686eaeeba

Virtual server type (instance type)

t2.medium

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or

Select the existing security group which was made for Node in the beginning of the experiment.

Network settings

Network

vpc-060600620a27873b6

Subnet

No preference (Default subnet in any availability zone)

Auto-assign public IP

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)

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

Select security groups

harade-case-study-node sg-0900c9044ab80cff5

VPC: vpc-060600620a27873b6

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

Summary

Number of instances

2

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI)

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

ami-0866a3c8686eaeeba

Virtual server type (instance type)

t2.medium

Firewall (security group)

harade-case-study-node

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or

Successfully, created the 3 ec2 instances (1 for master & 2 for nodes).

Instances (3) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
harade-master	i-0a1e468afe5c3aec	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1c	ec2-54-91-
harade-node2	i-0f3455789dffef9e	Running	t2.medium	Initializing	View alarms +	us-east-1c	ec2-98-81-
harade-node1	i-06c35879fb77a6653	Running	t2.medium	Initializing	View alarms +	us-east-1c	ec2-52-73-

Connect to the Instances

After launching the instances, go to the EC2 dashboard, select each instance, and click Connect to get the SSH command.

EC2 > Instances > i-0a1e468afe5c3aec > Connect to instance

Connect to instance Info

Connect to your instance i-0a1e468afe5c3aec (harade-master) using any of these options

EC2 Instance Connect | **Session Manager** | **SSH client** (selected) | **EC2 serial console**

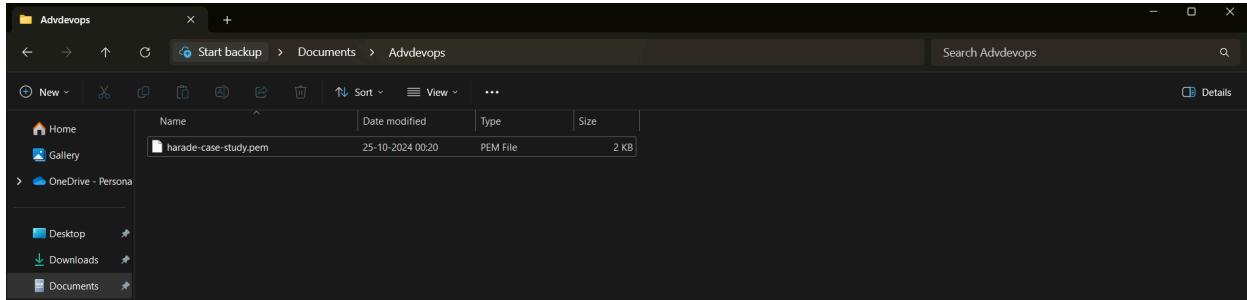
Instance ID
i-0a1e468afe5c3aec (harade-master)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is harade-case-study.pem
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "harade-case-study.pem"
- Connect to your instance using its Public DNS:
ec2-54-91-0-42.compute-1.amazonaws.com

Example:
ssh -i "harade-case-study.pem" ubuntu@ec2-54-91-0-42.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.

Locate the folder in which the .pem file is placed.



Open your terminal,

navigate to the folder where your .pem key is stored, and connect to each instance using:
ssh -i "your-key.pem" ubuntu@your-ec2-public-ip

```
ubuntu@ip-172-31-23-247:~ $ ssh -i "harade-case-study.pem" ubuntu@ip-172-31-20-191:~ 
The authenticity of host 'ip-172-31-20-191 (172.31.20.191)' can't be established.
ED25519 key fingerprint is SHA256:rzxdMaZBZAwyARxhnFlbhpzozn/6CKMjvoP97ylQ/PY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ip-172-31-20-191' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

System information as of Thu Oct 24 19:04:17 UTC 2024

System load: 0.0          Processes:           113
Usage of /: 22.8% of 6.71GB   Users logged in:     0
Memory usage: 5%            IPv4 address for enX0: 172.31.23.247
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.

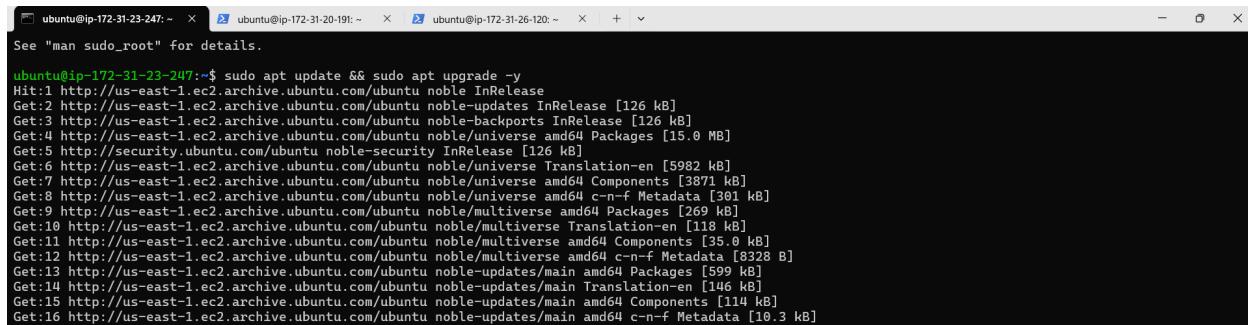
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-23-247:~ $
```

Update the Instances:

On all nodes (master and workers), update the system:

```
sudo apt update && sudo apt upgrade -y
```



```
ubuntu@ip-172-31-23-247:~$ sudo apt update && sudo apt upgrade -y
See "man sudo_root" for details.

ubuntu@ip-172-31-23-247:~$ 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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
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 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8228 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [599 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [146 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [114 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [10.3 kB]
```

Install Docker on All Instances (Master and Nodes)

On each instance (Master, Node 1, Node 2), run the following commands to install Docker:

```
# Add Docker's official GPG key
```

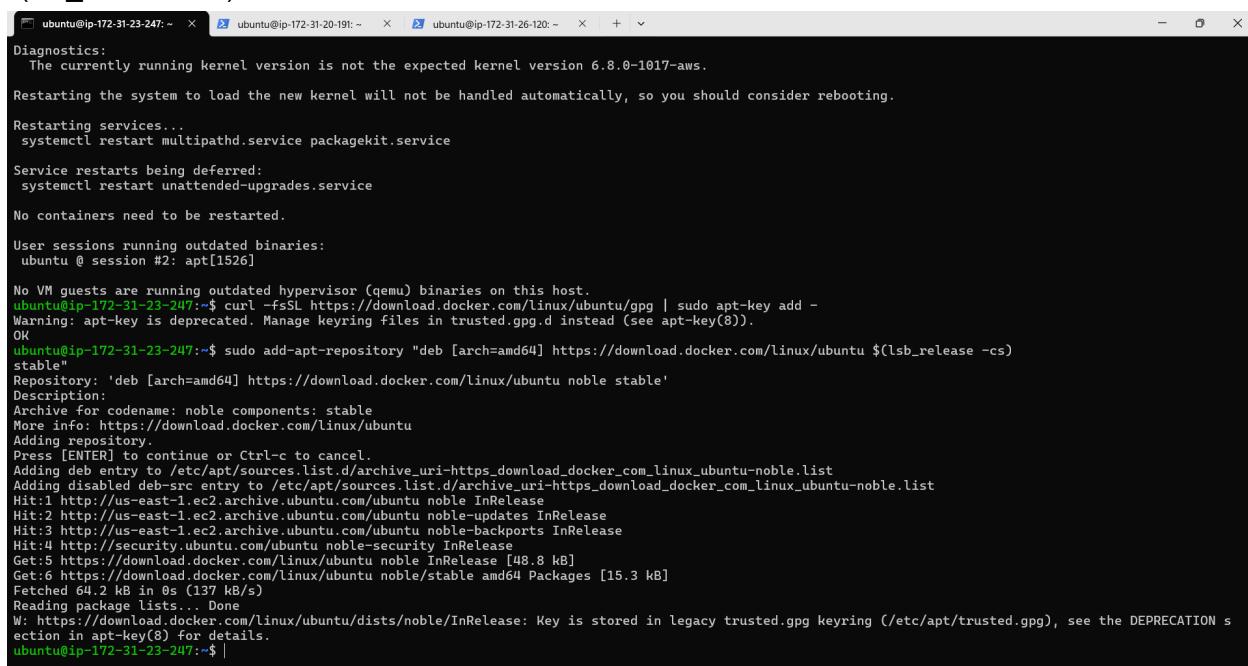
```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```



```
ubuntu@ip-172-31-23-247:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-23-247:~$ |
```

```
# Add Docker's APT repository
```

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
```



```
Diagnostics:
The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
systemctl restart multipathd.service packagekit.service

Service restarts being deferred:
systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #2: apt[1526]

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

ubuntu@ip-172-31-23-247:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-23-247:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs)
stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Fetched 64.2 kB in 0s (137 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-23-247:~$ |
```

```
# Update the package index sudo apt-get update
```

```

ubuntu@ip-172-31-26-120:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository...
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Fetched 64.2 kB in 0s (148 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-26-120:~$ 

[ 1  ubuntu@ip-172-31-23-247:~ ×  2  ubuntu@ip-172-31-20-191:~ ×  3  ubuntu@ip-172-31-26-120:~ ×  +  ×
systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #2: apt[1526]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-23-247:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-23-247:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository...
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Fetched 64.2 kB in 0s (137 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-23-247:~$ 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 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.
ubuntu@ip-172-31-23-247:~$ 

```

Install Docker CE (Community Edition)

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

```

[ 1  ubuntu@ip-172-31-23-247:~ ×  2  ubuntu@ip-172-31-20-191:~ ×  3  ubuntu@ip-172-31-26-120:~ ×  +  ×
ubuntu@ip-172-31-26-120:~$ sudo apt-get install -y docker-ce
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:
  autofs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 0 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu/noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
Get:6 https://download.docker.com/linux/ubuntu/noble/stable amd64 docker-buildx-plugin amd64 0.17.1-1~ubuntu.24.04~noble [30.3 MB]
Get:7 https://download.docker.com/linux/ubuntu/noble/stable amd64 docker-ce-cli amd64 5.27.3.1-1~ubuntu.24.04~noble [15.0 MB]
Get:8 https://download.docker.com/linux/ubuntu/noble/stable amd64 docker-ce amd64 5:27.3.1-1~ubuntu.24.04~noble [25.6 MB]
Get:9 https://download.docker.com/linux/ubuntu/noble/stable amd64 docker-ce-rootless-extras amd64 5:27.3.1-1~ubuntu.24.04~noble [9588 kB]
Get:10 https://download.docker.com/linux/ubuntu/noble/stable amd64 docker-compose-plugin amd64 2.29.7-1~ubuntu.24.04~noble [12.7 MB]
Fetched 123 MB in 2s (74.1 MB/s)
Selecting previously unselected package pigz.
(Reading database ... 98430 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../1-containerd.io_1.7.22-1_amd64.deb ...

```

Configure Docker to use systemd as the cgroup driver

```
sudo mkdir -p /etc/docker
```

```
cat <<EOF | sudo tee /etc/docker/daemon.json
```

```
{
```

```

"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF

```

```

ubuntu@ip-172-31-23-247:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-23-247:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-23-247:~$ |

```

```

ubuntu@ip-172-31-23-247:~ x  ubuntu@ip-172-31-20-19:~ x  ubuntu@ip-172-31-26-120:~ x + v
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.8.0-1016-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.
Restarting services...
Service restarts being deferred:
  systemctl restart unattended-upgrades.service
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-172-31-26-120:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-26-120:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-26-120:~$ |

```

Enable and start Docker sudo systemctl enable docker

```

ubuntu@ip-172-31-23-247:~ x  ubuntu@ip-172-31-20-19:~ x  ubuntu@ip-172-31-26-120:~ x + v
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
 6.8.0-1016-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.
Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.
Restarting services...
Service restarts being deferred:
  systemctl restart unattended-upgrades.service
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-172-31-23-247:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-23-247:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-23-247:~$ sudo systemctl enable 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-172-31-23-247:~$ |

```

```
sudo systemctl daemon-reload
```

```
ubuntu@ip-172-31-23-247:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-23-247:~$ |
```

```
sudo systemctl restart docker
```

```
ubuntu@ip-172-31-23-247:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-23-247:~$ sudo systemctl restart docker
ubuntu@ip-172-31-23-247:~$ |
```



Install Kubernetes on All Instances

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
```

```
ubuntu@ip-172-31-23-247:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N) y
ubuntu@ip-172-31-23-247:~$ |
```

```
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1016-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
  systemctl restart unattended-upgrades.service

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-172-31-26-128:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-26-128:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-26-128:~$ sudo systemctl enable 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-172-31-26-128:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-26-128:~$ sudo systemctl restart docker
ubuntu@ip-172-31-26-128:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
ubuntu@ip-172-31-26-128:~$ |
```

```
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-172-31-23-247:~$ 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-172-31-23-247:~$ |
```

```

[1] 100%  ubuntu@ip-172-31-23-247: / ~ [2] 100%  ubuntu@ip-172-31-20-191: / ~ [3] 100%  ubuntu@ip-172-31-26-120: / ~ + - 
Scanning linux images...
Pending kernel upgrade!
Running kernel version:
 6.8.0-1016-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
Service restarts being deferred:
  systemctl restart unattended-upgrades.service

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-172-31-20-191:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-20-191:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-20-191:~$ sudo systemctl enable 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-172-31-20-191:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-20-191:~$ sudo systemctl restart docker
ubuntu@ip-172-31-20-191:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
ubuntu@ip-172-31-20-191:~$ 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-172-31-20-191:~$ |

```

sudo apt-get update

```

ubuntu@ip-172-31-23-247:/$ sudo nano /etc/apt/sources.list.d/kubernetes.list
ubuntu@ip-172-31-23-247:/$ sudo apt-get update
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 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [19.9 kB]
Get:8 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [6097 B]
Fetched 153 kB in 1s (302 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-23-247:/$ |

```

sudo apt-get install -y kubelet kubeadm kubectl

```

[1] 100%  ubuntu@ip-172-31-23-247: / ~ [2] 100%  ubuntu@ip-172-31-20-191: / ~ [3] 100%  ubuntu@ip-172-31-26-120: / ~ + - 
Reading package lists... Done
ubuntu@ip-172-31-23-247:~$ 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 0 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 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.2-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubectl 1.31.2-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubelet 1.31.2-1.1 [15.2 MB]
Fetched 87.4 MB in 1s (77.4 MB/s)
Selecting previously unselected package conntrack.
(Reading database... 989696 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3al.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_1.31.2-1.1_amd64.deb ...
Unpacking kubeadm (1.31.2-1.1) ...
Selecting previously unselected package kubectl.
Preparing to unpack .../3-kubectl_1.31.2-1.1_amd64.deb ...
Unpacking kubectl (1.31.2-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.5.1-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.5.1-1.1) ...
Selecting previously unselected package kubelet.

```

```
ubuntu@ip-172-31-23-247:~ X ubuntu@ip-172-31-20-191:~ X ubuntu@ip-172-31-26-120:~ X + v

*** System restart required ***
Last login: Thu Oct 24 19:06:56 2024 from 206.84.225.145
ubuntu@ip-172-31-20-191:~$ 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 kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 0 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/istv:/kubernetes/:core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/istv:/kubernetes/:core:/stable:/v1.31/deb kubeadm 1.31.2-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/istv:/kubernetes/:core:/stable:/v1.31/deb kubectl 1.31.2-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/istv:/kubernetes/:core:/stable:/v1.31/deb kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/istv:/kubernetes/:core:/stable:/v1.31/deb kubelet 1.31.2-1.1 [15.2 MB]
Fetched 87.4 MB in 1s (65.6 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 98696 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3a1.4.8-1ubuntu1_amd64.deb ...
Unpacking conntrack (1:1.4.8-1ubuntu1) ...
```

sudo apt-mark hold kubelet kubeadm kubectl

```
Last login: Thu Oct 24 19:04:18 2024 from 206.84.225.145
ubuntu@ip-172-31-23-247:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-23-247:~$ |
```

sudo systemctl enable --now kubelet

```
ubuntu@ip-172-31-23-247:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-23-247:~$ |
```

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-23-247:~ X ubuntu@ip-172-31-20-191:~ X ubuntu@ip-172-31-26-120:~ X + v

ubuntu@ip-172-31-23-247:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 0 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 MB]
Fetched 47.2 MB in 1s (66.0 MB/s)
(Reading database ... 98753 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 98733 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1016-aws
Diagnostics:
```

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

```
ubuntu@ip-172-31-23-247:~$ sudo mkdir -p /etc/containerd  
ubuntu@ip-172-31-23-247:~$ client_loop: send disconnect: Connection reset  
C:\Users\HP\Documents\Advdevops>ssh -i "harade-case-study.pem" ubuntu@ec2-54-91-0-42.compute-1.amazonaws.com  
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)  
  
* Documentation: https://help.ubuntu.com  
* Management: https://landscape.canonical.com  
* Support: https://ubuntu.com/pro  
  
System information as of Thu Oct 24 20:38:26 UTC 2024  
  
System load: 0.0 Processes: 123  
Usage of /: 42.6% of 6.71GB Users logged in: 1  
Memory usage: 10% IPv4 address for enX0: 172.31.23.247  
Swap usage: 0%  
  
* Ubuntu Pro delivers the most comprehensive open source security and  
compliance features.  
  
https://ubuntu.com/aws/pro  
  
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  
  
*** System restart required ***  
Last login: Thu Oct 24 20:25:09 2024 from 106.209.201.24  
ubuntu@ip-172-31-23-247:~$ 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 = ""
```

```
sudo systemctl restart containerd
```

```
C:\Windows\System32\cmd.exe > ubuntu@ip-172-31-20-191:~> ubuntu@ip-172-31-26-120:~>  
  
[config_path = ""  
max_concurrent_downloads = 3  
max_concurrent_uploaded_layers = 3  
  
[[plugins."io.containerd.transfer.v1.local".unpack_config]]  
differ = ""  
platform = "linux/amd64"  
snapshotter = "overlayfs"  
  
[proxy_plugins]  
  
[stream_processors]  
  
[stream_processors."io.containerd.ocicrypt.decoder.v1.tar"]  
accepts = ["application/vnd.oci.image.layer.v1.tar+encrypted"]  
args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]  
env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]  
path = "ctd-decoder"  
returns = "application/vnd.oci.image.layer.v1.tar"  
  
[stream_processors."io.containerd.ocicrypt.decoder.v1.tar.gzip"]  
accepts = ["application/vnd.oci.image.layer.v1.tar+gzip+encrypted"]  
args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]  
env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]  
path = "ctd-decoder"  
returns = "application/vnd.oci.image.layer.v1.tar+gzip"  
  
[timeouts]  
"io.containerd.timeout.bolt.open" = "0s"  
"io.containerd.timeout.metrics.shimstats" = "2s"  
"io.containerd.timeout.shim.cleanup" = "5s"  
"io.containerd.timeout.shim.load" = "5s"  
"io.containerd.timeout.shim.shutdown" = "3s"  
"io.containerd.timeout.task.state" = "2s"  
  
[ttrpc]  
address = ""  
gid = 0  
uid = 0  
ubuntu@ip-172-31-20-191:~$ sudo systemctl restart containerd  
ubuntu@ip-172-31-20-191:~$ |
```

sudo systemctl enable containerd

```
ubuntu@ip-172-31-23-247:~$ client_loop: send disconnect: Connection reset
C:\Users\HP\Documents\Advdevops>ssh -i "harade-case-study.pem" ubuntu@ec2-54-91-0-42.compute-1.amazonaws.com
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

System information as of Thu Oct 24 20:45:07 UTC 2024

System load: 0.0          Processes:           126
Usage of /: 42.6% of 6.71GB Users logged in:      1
Memory usage: 10%          IPv4 address for enX0: 172.31.23.247
Swap usage:  0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

https://ubuntu.com/aws/pro

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

*** System restart required ***
Last login: Thu Oct 24 20:38:27 2024 from 106.209.201.24
ubuntu@ip-172-31-23-247:~$ sudo systemctl restart containerd
ubuntu@ip-172-31-23-247:~$ sudo systemctl enable containerd
ubuntu@ip-172-31-23-247:~$ |
```

sudo systemctl status containerd

```
* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

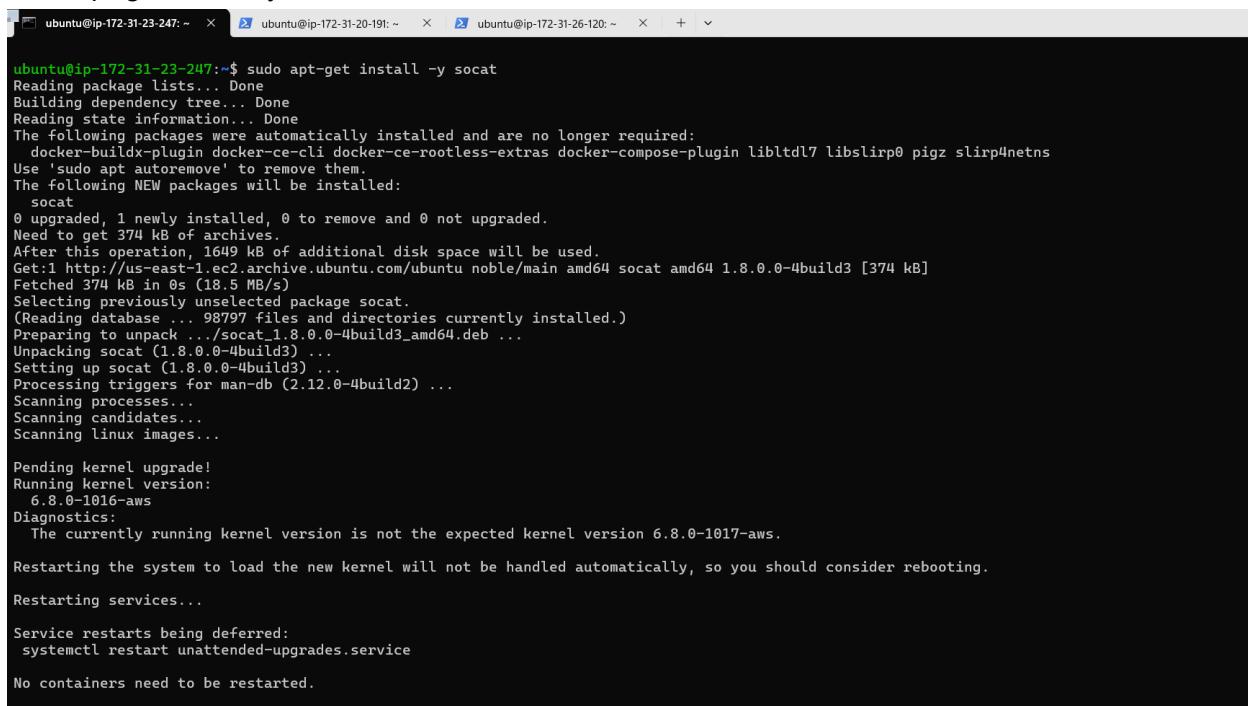
0 updates can be applied immediately.

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

*** System restart required ***
Last login: Thu Oct 24 20:38:27 2024 from 106.209.201.24
ubuntu@ip-172-31-23-247:~$ sudo systemctl restart containerd
ubuntu@ip-172-31-23-247:~$ sudo systemctl enable containerd
ubuntu@ip-172-31-23-247:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-10-24 20:45:22 UTC; 45s ago
     Docs: https://containerd.io
 Main PID: 16743 (containerd)
   Tasks: 8
      Memory: 13.5M (peak: 13.7M)
        CPU: 250ms
       CGroup: /system.slice/containerd.service
               └─16743 /usr/bin/containerd

Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486120880Z" level=info msg="Start subscribing containerd event"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486164168Z" level=info msg="Start recovering state"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486173186Z" level=info msg="serving... address=/run/containerd/containerd.sock"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486207375Z" level=info msg="serving... address=/run/containerd/containerd.sock"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486216114Z" level=info msg="Start event monitor"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486237915Z" level=info msg="Start snapshots syncer"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486247857Z" level=info msg="Start cni network conf syncer for default"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486254665Z" level=info msg="Start streaming server"
Oct 24 20:45:22 ip-172-31-23-247 containerd[16743]: time="2024-10-24T20:45:22.486311341Z" level=info msg="containerd successfully booted in 0.028191s"
Oct 24 20:45:22 ip-172-31-23-247 systemd[1]: Started containerd.service - containerd container runtime.
Lines 1-21/21 (END)
```

```
sudo apt-get install -y socat
```



```
ubuntu@ip-172-31-23-247:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (18.5 MB/s)
Selecting previously unselected package socat.
(Reading database ... 98797 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 candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1016-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1017-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

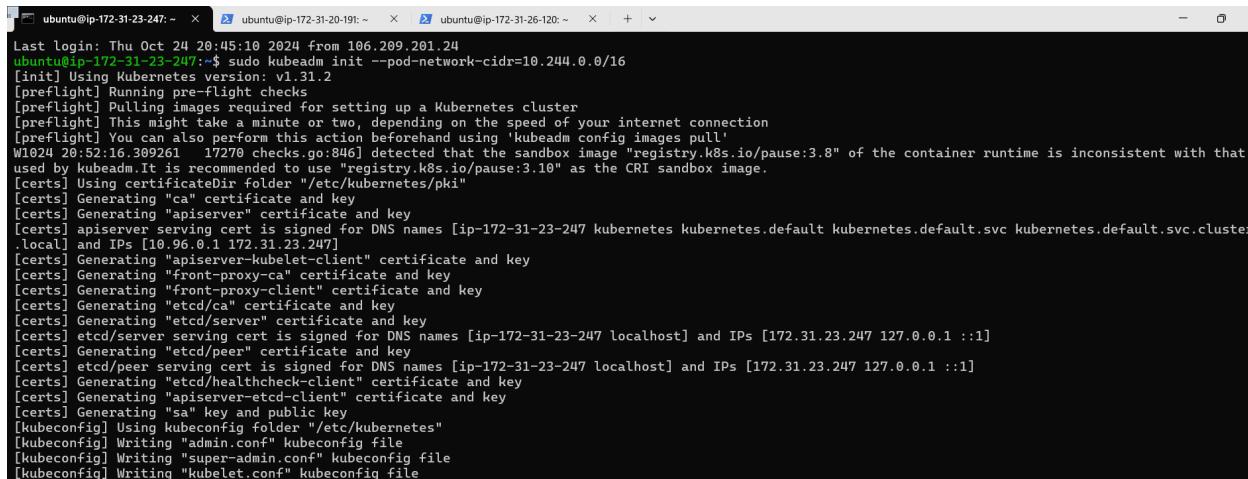
Service restarts being deferred:
  systemctl restart unattended-upgrades.service

No containers need to be restarted.
```

Initialize the Kubernetes Master Node

Initialize the Kubecluster .Now Perform this Command only for Master.

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



```
Last login: Thu Oct 24 20:45:10 2024 from 106.209.201.24
ubuntu@ip-172-31-23-247:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[kubelet] Using Kubernetes version: v1.31.2
[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'
W1024 20:52:16.309261 17270 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificate-dir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-23-247 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.23.247]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-23-247 localhost] and IPs [172.31.23.247 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-23-247 localhost] and IPs [172.31.23.247 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
```

Once the initialization is complete, you will see a command that looks like this:

```
sudo kubeadm join 172.31.23.247:6443 --token lgl6o7.qmk75sn9uifh3131 \
    --discovery-token-ca-cert-hash sha256:372ecad85c51a779a6e7e587a28d7ac671709e434256bd83f5dbddd760f22a71
```

Next, configure kubectl to interact with the cluster: 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-172-31-23-247:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-23-247:~$ |
```

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

```
ubuntu@ip-172-31-23-247:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE      VERSION
ip-172-31-23-247  NotReady  control-plane  3m2s    v1.31.2
ubuntu@ip-172-31-23-247:~$ |
```

Join Worker Nodes to the Cluster

On Node 1 and Node 2, run the kubeadm join command you saved from the master node initialization. It should look like this:

sudo kubeadm join :6443 --token --discovery-token-ca-cert-hash (In my case, command is as follows: sudo kubeadm join 172.31.26.24:6443 --token 9accqk.weumqfijmpmsoilz
--discovery-token-ca-cert-hash
sha256:6311d1e2d998752029085c10573c9facf87e6ffafeef00f3ddc882f28d7b7c45)

```
ubuntu@ip-172-31-20-191:~$ sudo kubeadm join 172.31.23.247:6443 --token lgl6o7.qmk75sn9uifh3131 \
--discovery-token-ca-cert-hash sha256:372ecad85c51a779a6e7e587a28d7ac671709e434256bd83f5dbddd760f22a71
[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 1.001799036s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
ubuntu@ip-172-31-20-191:~$ |
```

```
ubuntu@ip-172-31-23-247:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE      VERSION
ip-172-31-20-191  NotReady  <none>     27s      v1.31.2
ip-172-31-23-247  NotReady  control-plane  5m14s    v1.31.2
ubuntu@ip-172-31-23-247:~$ |
```

```
ubuntu@ip-172-31-26-120:~$ sudo kubeadm join 172.31.23.247:6443 --token lgl6o7.qmk75sn9uifh3131 \
--discovery-token-ca-cert-hash sha256:372ecad85c51a779a6e7e587a28d7ac671709e434256bd83f5dbddd760f22a71
[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 1.001705578s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
ubuntu@ip-172-31-26-120:~$ |
```

Verify the Cluster

On the master node, verify that the nodes have joined the cluster: kubectl get nodes

```
ip 172 31 23 247 NotReady control-plane 5m14s v1.31.2
ubuntu@ip-172-31-23-247:~$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
ip-172-31-20-191 NotReady <none> 72s v1.31.2
ip-172-31-23-247 NotReady control-plane 5m59s v1.31.2
ip-172-31-26-120 NotReady <none> 7s v1.31.2
ubuntu@ip-172-31-23-247:~$ |
```

Install a Network Plugin (Calico)

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-172-31-23-247:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

sudo systemctl status kubelet

```
ubuntu@ip-172-31-23-247:~$ sudo systemctl status kubelet
● kubelet.service - 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 Thu 2024-10-24 20:52:42 UTC; 7 min ago
       Docs: https://kubernetes.io/docs/
     Main PID: 17937 (kubelet)
        Tasks: 10 (limit: 4676)
      Memory: 32.7M (peak: 33.2M)
        CPU: 7.880s
       CGroup: /system.slice/kubelet.service
                 └─17937 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/...
```



```
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: E1024 21:00:12.943609 17937 kuberuntime_container.go:851] "Kill container failed" err=<
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]:         rpc error: code = Unknown desc = failed to kill container "2ab7eb8c9ff9dcda281d39a62d0693d2b9a4d164" >
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]:         : unknown
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: > pod="kube-system/etcd-ip-172-31-23-247" podUID="fabe0b309d193997278b5c4e8d268e2f" containerName="etcd" >
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: E1024 21:00:12.951620 17937 log.go:32] "StopPodsSandbox from runtime service failed" err=<
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]:         rpc error: code = Unknown desc = failed to stop container "2ab7eb8c9ff9dcda281d39a62d0693d2b9a4d164" >
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]:         : unknown
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: > podSandboxID="838e9b8e4dc2f7fd92dd2659096bfbc354dff85b406a6fa71d9705bbc422f53d"
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: E1024 21:00:12.951667 17937 kuberuntime_manager.go:1479] "Failed to stop sandbox" podSandboxID={"Type":>
Oct 24 21:00:12 ip-172-31-23-247 kubelet[17937]: E1024 21:00:12.951707 17937 kubelet.go:1865] "KillPod failed" err="["failed to \\"KillContainer\\" for \\"etcd\\\" podSandboxID=838e9b8e4dc2f7fd92dd2659096bfbc354dff85b406a6fa71d9705bbc422f53d" at index 0"]" >
lines 1-23/23 (END)
```

Now Run command (we can see Status is ready).

kubectl get nodes -o wide

```
ubuntu@ip-172-31-23-247:~$ kubectl get nodes -o wide
NAME STATUS ROLES AGE VERSION INTERNAL-IP EXTERNAL-IP OS-IMAGE KERNEL-VERSION CONTAINER-RUNTIME
ip-172-31-20-191 Ready <none> 3m28s v1.31.2 172.31.20.191 <none> Ubuntu 24.04.1 LTS 6.8.0-1016-aws containerd://1.7.12
ip-172-31-23-247 Ready control-plane 8m15s v1.31.2 172.31.23.247 <none> Ubuntu 24.04.1 LTS 6.8.0-1016-aws containerd://1.7.12
ip-172-31-26-120 Ready <none> 2m23s v1.31.2 172.31.26.120 <none> Ubuntu 24.04.1 LTS 6.8.0-1016-aws containerd://1.7.12
ubuntu@ip-172-31-23-247:~$ |
```

Rename/Label the Nodes

```
kubectl label node ip-172-31-22-204 kubernetes.io/role=Node1
```

```
kubectl label node ip-172-31-23-77 kubernetes.io/role=Node2
```

```
ubuntu@ip-172-31-23-247:~$ kubectl label node ip-172-31-20-191 kubernetes.io/role=Node1
node/ip-172-31-20-191 labeled
ubuntu@ip-172-31-23-247:~$ kubectl label node ip-172-31-26-120 kubernetes.io/role=Node2
node/ip-172-31-26-120 labeled
ubuntu@ip-172-31-23-247:~$ |
```

Check Again

Run (again to confirm all nodes are in the Ready state and properly labeled).

```
kubectl get nodes -o wide
```

```
ubuntu@ip-172-31-23-247:~$ kubectl get nodes -o wide
NAME      STATUS   ROLES      AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE       KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-20-191   Ready    Node1      8m32s   v1.31.2   172.31.20.191  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws  containerd://1.7.12
ip-172-31-23-247   Ready    control-plane   13m     v1.31.2   172.31.23.247  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws  containerd://1.7.12
ip-172-31-26-120   Ready    Node2      7m27s   v1.31.2   172.31.26.120  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws  containerd://1.7.12
ubuntu@ip-172-31-23-247:~$ |
```

Deploy the Sample Nginx Application

1. Deploy Nginx:

- Create a deployment:

```
kubectl create deployment nginx --image=nginx
```

```
ubuntu@ip-172-31-23-247:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
ubuntu@ip-172-31-23-247:~$ |
```

Expose the Nginx Deployment:

Expose the deployment via NodePort (or LoadBalancer if using cloud load balancing):

```
kubectl expose deployment nginx --port=80 --type=NodePort
```

```
ubuntu@ip-172-31-23-247:~$ kubectl expose deployment nginx --port=80 --type=NodePort
service/nginx exposed
ubuntu@ip-172-31-23-247:~$ |
```

Get the NodePort:

To get the NodePort: kubectl get services

- Note the NodePort value (it will be something like 30000-32767).

```
service/nginx exposed
ubuntu@ip-172-31-23-247:~$ kubectl get services
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)      AGE
kubernetes  ClusterIP  10.96.0.1    <none>        443/TCP     16m
nginx      NodePort   10.96.126.244  <none>        80:31682/TCP 64s
ubuntu@ip-172-31-23-247:~$ |
```

Access the Application:

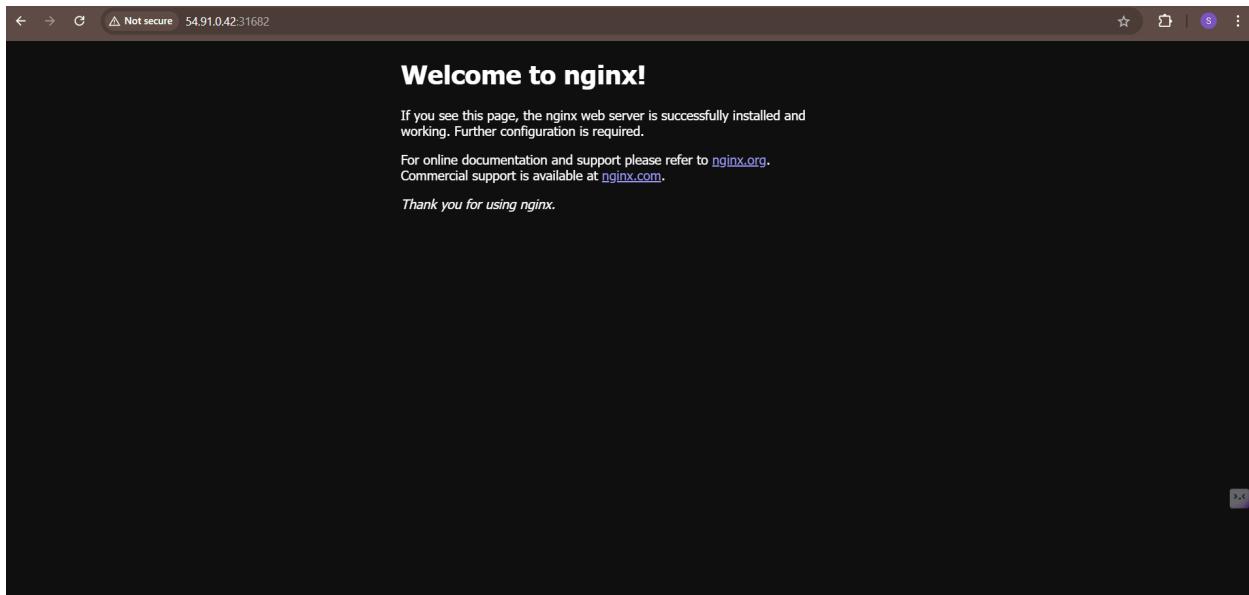
- Open the browser and visit <http://public-ip> of master instance:port to verify that the Nginx server is running.

```

PS C:\Users\HP> curl http://54.91.0.42:31682
{
  StatusCode : 200
  StatusDescription : OK
  Content : <!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style...
RawContent : HTTP/1.1 200 OK
Connection: keep-alive
Accept-Ranges: bytes
Content-Length: 615
Content-Type: text/html
Date: Thu, 24 Oct 2024 21:15:05 GMT
ETag: "66fd630f-267"
Last-Modified: Wed, 02 Oct 2024 ...
Forms : {}
Headers : {[Connection, keep-alive], [Accept-Ranges, bytes], [Content-Length, 615], [Content-Type, text/html]...}
Images : {}
InputFields : {}
Links : {@{innerHTML=nginx.org; innerText=nginx.org; outerHTML=<A href="http://nginx.org/">nginx.org</A>; outerText=nginx.org; tagName=A; href=http://nginx.org/}, @{innerHTML=nginx.com; innerText=nginx.com; outerHTML=<A href="http://nginx.com/">nginx.com</A>; outerText=nginx.com; tagName=A; href=http://nginx.com/}}
ParsedHtml : mshtml.HTMLDocumentClass
RawContentLength : 615
}

PS C:\Users\HP> |

```



Conclusion:-

The case study successfully demonstrated the setup of a Kubernetes cluster on AWS and the deployment of a sample Nginx application. The steps outlined provide a clear pathway for future deployments and emphasize the importance of understanding each component's role within a Kubernetes architecture.