

CaseStudy

Kubernetes Deployment

A Kubernetes Deployment is used to tell Kubernetes how to create or modify instances of the pods that hold a containerized application. Deployments can scale the number of replica pods, enable the rollout of updated code in a controlled manner, or roll back to an earlier deployment version if necessary.

Steps:

1. Create an EC2 Ubuntu Instance on AWS.

The screenshot shows the 'Launch an instance' page in the AWS Management Console. The breadcrumb navigation at the top reads 'EC2 > ... > Launch an instance'. The main heading is 'Launch an instance' with an 'Info' link. Below the heading is a descriptive paragraph: '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.' The page is divided into sections. The first section is 'Name and tags' with an 'Info' link. It contains a 'Name' label and a text input field with the value 'Master'. To the right of the input field is a link 'Add additional tags'. The second section is 'Application and OS Images (Amazon Machine Image)' with a dropdown arrow and an 'Info' link. It contains a paragraph: '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'. At the bottom of this section is a search bar with a magnifying glass icon and the placeholder text 'Search our full catalog including 1000s of application and OS images'.

EC2 > ... > Launch an instance

Launch an instance [Info](#)

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

Name and tags [Info](#)

Name

[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

▼ Application and OS Images (Amazon Machine Image) Info

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

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

Recents Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Li

Browse more AMIs

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

ami-0866a3c8686eaeaba (64-bit (x86)) / ami-0325498274077fac5 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

▼ Summary

Number of instances Info

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.6.2...read more

ami-06b21cceaef8cd686

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel Launch instance

Preview code

2. Edit the Security Group Inbound Rules to allow SSH

Inbound rules Info

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
SSH ▼	TCP	22	Custom ▼ Q 0.0.0.0/0 ✕	<div></div> Delete
HTTP ▼	TCP	80	Custom ▼ Q 0.0.0.0/0 ✕	<div></div> Delete
HTTPS ▼	TCP	443	Custom ▼ Q 0.0.0.0/0 ✕	<div></div> Delete

Add rule

3. SSH into the machine `ssh -i <keyname>.pem ubuntu@<public_ip_address>`

```
Lenovo@LAPTOP-8VIT9J4N MINGW64 ~/Downloads
$ ssh -i "adi.pem" ubuntu@ec2-54-221-25-104.compute-1.amazonaws.com
The authenticity of host 'ec2-54-221-25-104.compute-1.amazonaws.com (54.221.25.104)' can't be established.
ED25519 key fingerprint is SHA256:EQ7HQpmT3L+nP2hoSutoRCa3Ztche1bw+LS36PE+qRQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-221-25-104.compute-1.amazonaws.com' (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 Mon Oct 21 03:33:17 UTC 2024

System load:  0.06               Processes:           114
Usage of /:   29.3% of 6.71GB    Users logged in:    1
Memory usage: 27%               IPv4 address for enx0: 172.31.17.238
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
```

Step 4: Run the below commands to install and setup Docker. `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-172-31-17-238:~$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debotstrap docker-buildx
  docker-compose-v2 docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc
  ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 0 not upgraded.
Need to get 76.8 MB of archives.
After this operation, 289 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz a
md64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-uti
ls amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 ru
nc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 co
ntainerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dns-root-d
ata all 2023112702~willsync1 [4450 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dnsmasq-ba
se amd64 2.90-2build2 [375 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd6
4 docker.io amd64 24.0.7-0ubuntu4.1 [29.1 MB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 ubuntu
-fan all 0.12.16 [35.2 kB]
Fetched 76.8 MB in 2s (44.9 MB/s)
Preconfiguring packages ...
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 bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7.1-1ubuntu2_amd64.deb ...
Unpacking bridge-utils (1.7.1-1ubuntu2) ...

```

sudo apt-get update

```

ubuntu@ip-172-31-80-240:~$ 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: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the fil
e has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPREC
ATION section in apt-key(8) for details.
ubuntu@ip-172-31-80-240:~$ 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:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 25 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 noble/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 [63.0 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 0.17.1-1ubuntu.24.04-noble [30.3 MB]

```

sudo apt-get install -y docker-ce


```

ubuntu@ip-172-31-80-240:~$ 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:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 25 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 noble/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 (74.2 MB/s)
Selecting previously unselected package pigz.
(Reading database ... 67836 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 ...

```

```

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

```

```

ubuntu@ip-172-31-80-240:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-80-240:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}

```

```

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

```

```

ubuntu@ip-172-31-80-240:~$ 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-172-31-80-240:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-80-240:~$ sudo systemctl restart docker

```

Step 5: Run the below command to install Kubernetes. curl -fsSL

```

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

```

```
ubuntu@ip-172-31-80-240:~$ 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-172-31-80-240:~$
```

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

```
ubuntu@ip-172-31-80-240:~$ 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
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:6 https://prod-odn.packages.k8s.io/repositories/isv:/kubernetes/core:/stable:/v1.31/deb InRelease
Fetched 126 kB in 1s (240 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
kubelet is already the newest version (1.31.1-1.1).
kubeadm is already the newest version (1.31.1-1.1).
kubectl is already the newest version (1.31.1-1.1).
0 upgraded, 0 newly installed, 0 to remove and 25 not upgraded.
kubelet was already set on hold.
kubeadm was already set on hold.
kubectl was already set on hold.
```

sudo systemctl enable --now kubelet

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

```
ubuntu@ip-172-31-80-240:~$ sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.1
[preflight] Running pre-flight checks
W1020 10:02:22.795027 5411 checks.go:1080 [preflight] WARNING: Couldn't create the interface used for talking to the container runtime: failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService
[WARNING FileExisting-socat]: socat not found in system path
[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'
error execution phase preflight: [preflight] Some fatal errors occurred:
failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService[preflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=...'
To see the stack trace of this error execute with --v=5 or higher
ubuntu@ip-172-31-80-240:~$
```

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-80-240:~$ 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 libsllp0 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 25 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 (70.4 MB/s)
(Reading database ... 68159 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1ubuntu.24.04-noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68139 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 ...
```

sudo mkdir -p /etc/containerd

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

```

ubuntu@ip-172-31-80-240:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

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

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

```

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

```

ubuntu@ip-172-31-80-240:~$ sudo systemctl restart containerd
ubuntu@ip-172-31-80-240:~$ sudo systemctl enable containerd
ubuntu@ip-172-31-80-240:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Sun 2024-10-20 10:05:40 UTC; 21s ago
     Docs: https://containerd.io
   Main PID: 5885 (containerd)
    Tasks: 8
   Memory: 13.1M (peak: 13.9M)
      CPU: 129ms
   CGroup: /system.slice/containerd.service
           └─5885 /usr/bin/containerd

Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.905924648Z" level=error msg="failed to load cni during init, please check
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906093670Z" level=info msg="Start subscribing containerd event"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906132437Z" level=info msg="serving... address=/run/containerd/containerd
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906135956Z" level=info msg="Start recovering state"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906185395Z" level=info msg="serving... address=/run/containerd/containerd
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906205371Z" level=info msg="Start event monitor"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906215409Z" level=info msg="Start snapshots syncer"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906223526Z" level=info msg="Start cni network conf syncer for default"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906229655Z" level=info msg="Start streaming server"
Oct 20 10:05:40 ip-172-31-80-240 containerd[5885]: time="2024-10-20T10:05:40.906282756Z" level=info msg="containerd successfully booted in 0.028768s"
lines 1-21/21 (END)

```

sudo apt-get install -y socat

```

ubuntu@ip-172-31-80-240:~$ 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 25 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/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (11.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68203 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 processes... [=
Scanning processes... [==
Scanning processes... [===
Scanning processes... [====
Scanning processes... [=====
Scanning processes... [=====
Scanning processes... [=====

```

Step 6: Initialize the Kubecluster**sudo kubeadm init --pod-network-cidr=10.244.0.0/16**

```

ubuntu@ip-172-31-80-240:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.1
[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'
W1020 10:07:42.067638 6134 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-80-240.kubernetes.kubernetes.default.kubernetes.default.svc.kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.80.240]
[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-172-31-80-240.localhost] and IPs [172.31.80.240 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-80-240.localhost] and IPs [172.31.80.240 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file

```

Copy the mkdir and chown commands from the top and execute them. 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-172-31-80-240:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

```

Add a common networking plugin called flannel as mentioned in the code.

kubectl apply -f**<https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml>**

```

ubuntu@ip-172-31-80-240:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created

```

kubectl apply -f <https://k8s.io/examples/application/deployment.yaml>

```

ubuntu@ip-172-31-80-240:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created

```

kubectl get pods

```

ubuntu@ip-172-31-80-240:~$ kubectl get pods

```

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-87mfp	0/1	Pending	0	26s
nginx-deployment-d556bf558-fcfx2	0/1	Pending	0	26s


```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath = "{.item[0].metadata.name}")
ubuntu@ip-172-31-80-240:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending

```

kubectl get nodes

```

ubuntu@ip-172-31-80-240:~$ kubectl get nodes

```

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-80-240	Ready	control-plane	3m46s	v1.31.1

```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80

```

```

ubuntu@ip-172-31-80-240:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-80-240:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending

```

command kubectl taint nodes--all node-role.kubernetes.io/control-plane-node/ip-172-3120-171 untainted

```

ubuntu@ip-172-31-80-240:~$ command kubectl taint nodes --all node-role.kubernetes.io/control-plane:NoSchedule-
node/ip-172-31-80-240 untainted

```

kubectl get nodes

```

ubuntu@ip-172-31-80-240:~$ kubectl get nodes

```

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-80-240	Ready	control-plane	9m52s	v1.31.1

kubectl get pods

```

ubuntu@ip-172-31-80-240:~$ kubectl get pods

```

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-87mfp	1/1	Running	0	8m
nginx-deployment-d556bf558-fcfx2	1/1	Running	0	8m

```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8082:80

```

```

ubuntu@ip-172-31-80-240:~$ kubectl port-forward nginx-deployment-d556bf558-87mfp 8082:80
Forwarding from 127.0.0.1:8082 -> 80
Forwarding from [::1]:8082 -> 80
Handling connection for 8082

```

Step 8: Verify your deployment

Open up a new terminal and ssh to your EC2 instance.

Then, use this curl command to check if the Nginx server is running. curl

--head <http://127.0.0.1:8082>

```
ubuntu@ip-172-31-80-240:~$ curl --head http://127.0.0.1:8082
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sun, 20 Oct 2024 10:20:51 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

kubectl get services

```
ubuntu@ip-172-31-80-240:~$ kubectl get services
NAME            TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
kubernetes      ClusterIP     10.96.0.1     <none>         443/TCP    29m
```

kubectl create deployment nginx --image=nginx

```
ubuntu@ip-172-31-80-240:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
```

kubectl get deployments

```
ubuntu@ip-172-31-80-240:~$ kubectl get deployments
NAME            READY    UP-TO-DATE    AVAILABLE    AGE
nginx            1/1      1              1            11s
nginx-deployment 2/2      2              2            29m
```

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

```
ubuntu@ip-172-31-80-240:~$ kubectl expose deployment nginx --type=NodePort --port=80
service/nginx exposed
```

Nginx server is running successfully on the EC2 instance, and it's accessible locally via **localhost** on port **31801**.

curl <http://127.0.0.1/31801>

```
ubuntu@ip-172-31-80-240:~$ curl http://localhost:31301
<!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>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```



Conclusion:

In this experiment, we successfully set up Kubernetes and Docker on an AWS EC2 Ubuntu instance, configured the necessary settings, and initialized a Kubernetes cluster. We

deployed an Nginx server using a Kubernetes Deployment and implemented the Flannel networking plugin for pod communication. By checking the pod status and forwarding ports, we were able to access the Nginx server locally. The successful `200 OK` response from the `curl` command confirmed that the deployment was functioning correctly. This setup highlighted key Kubernetes operations, such as cluster management, application deployment, and verification, demonstrating the effectiveness of Kubernetes in orchestrating containerized applications efficiently.