

## Experiment No: 4

**AIM:** To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

### Theory :

Kubernetes, originally developed by Google, is an open-source container orchestration platform. It automates the deployment, scaling, and management of containerized applications, ensuring high availability and fault tolerance. Kubernetes is now the industry standard for container orchestration and is governed by the Cloud Native Computing Foundation (CNCF), with contributions from major cloud and software providers like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

**Kubernetes Deployment:** Is a resource in Kubernetes that provides declarative updates for Pods and ReplicaSets. With a Deployment, you can define how many replicas of a pod should run, roll out new versions of an application, and roll back to previous versions if necessary. It ensures that the desired number of pod replicas are running at all times.

#### Necessary Requirements:

- **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as Kubernetes demands sufficient resources for effective functioning.

- **Minimum Requirements:**

- Instance Type: t2.medium
- CPUs: 2
- Memory: Adequate for container orchestration.

This ensured that the Kubernetes cluster had the necessary resources to function smoothly

Step 1: Log in to your AWS Academy/personal account and launch a new Ec2 Instance. Select Ubuntu as AMI and t2.medium as Instance Type, create a key of type RSA with .pem extension, and move the downloaded key to the new folder.

**Instance summary for i-063600dea9823b368 (Exp\_4\_61)** Info

Updated less than a minute ago

<b>Instance ID</b> i-063600dea9823b368 (Exp_4_61)	<b>Public IPv4 address</b> 54.197.12.249   <a href="#">open address</a>	<b>Private IPv4 addresses</b> 172.31.85.68
<b>IPv6 address</b> -	<b>Instance state</b> Running	<b>Public IPv4 DNS</b> ec2-54-197-12-249.compute-1.amazonaws.com   <a href="#">open address</a>
<b>Hostname type</b> IP name: ip-172-31-85-68.ec2.internal	<b>Private IP DNS name (IPv4 only)</b> ip-172-31-85-68.ec2.internal	<b>Elastic IP addresses</b> -
<b>Answer private resource DNS name</b> IPv4 (A)	<b>Instance type</b> t2.medium	<b>AWS Compute Optimizer finding</b> <a href="#">Opt-in to AWS Compute Optimizer for recommendations.</a> <a href="#">Learn more</a>
<b>Auto-assigned IP address</b> 54.197.12.249 [Public IP]	<b>VPC ID</b> vpc-07187e57bdb9cb1f9	<b>Auto Scaling Group name</b> -
<b>IAM Role</b> -	<b>Subnet ID</b> subnet-00fe085523b65af02	

Step 2 :

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"
```

```
root@ip-172-31-85-68:/home/ubuntu# 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
root@ip-172-31-85-68:/home/ubuntu# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
root@ip-172-31-85-68:/home/ubuntu# sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive uri=https download docker com linux ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive uri=https download docker com linux ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [83.1 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4576 B]
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

sudo apt-get update

sudo apt-get install -y docker-ce

```
root@ip-172-31-85-68:/home/ubuntu# sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
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 DE
PRECATION section in apt-key(8) for details.
root@ip-172-31-85-68:/home/ubuntu# 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 slirp4netas
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
  slirp4netas
0 upgraded, 10 newly installed, 0 to remove and 143 not upgraded.
Need to get 123 MB of archives.
```

i-063600dea9823b368 (Exp\_4\_61)  
PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

sudo mkdir -p /etc/docker

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

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

```
root@ip-172-31-85-68:/home/ubuntu# sudo mkdir -p /etc/docker
root@ip-172-31-85-68:/home/ubuntu# cat <<EOF | sudo tee /etc/docker/daemon.json
> {
"exec-opts": ["native.cgroupdriver=systemd"]
}
> EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
root@ip-172-31-85-68:/home/ubuntu#
```

sudo systemctl enable docker

sudo systemctl daemon-reload

sudo systemctl restart docker

```
root@ip-172-31-85-68:/home/ubuntu# 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
root@ip-172-31-85-68:/home/ubuntu# sudo systemctl daemon-reload
root@ip-172-31-85-68:/home/ubuntu# sudo systemctl restart docker
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)  
PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

Step 3 :

Run the below command to install Kubernets.

```
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor  
-o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

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

```
root@ip-172-31-85-68:/home/ubuntu# curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/k  
ubernetes-apt-keyring.gpg  
root@ip-172-31-85-68:/home/ubuntu# 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/ /  
root@ip-172-31-85-68:/home/ubuntu#
```

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

```
root@ip-172-31-85-68:/home/ubuntu# sudo apt-get update  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease  
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease  
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease  
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease  
Get:6 https://prod-cdn.packages.k8s.io/repositories/iscv/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]  
Get:7 https://prod-cdn.packages.k8s.io/repositories/iscv/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]  
Fetched 6051 B in 1s (11.5 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  
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  
PRECATION section in apt-key(8) for details.  
root@ip-172-31-85-68:/home/ubuntu# 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 143 not upgraded.  
Need to get 87.4 MB of archives.
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

```
root@ip-172-31-85-68:/home/ubuntu# sudo apt-mark hold kubelet kubeadm kubectl  
kubelet set on hold.  
kubeadm set on hold.  
kubectl set on hold.  
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68



sudo systemctl enable --now kubelet

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

```
root@ip-172-31-85-68:/home/ubuntu# sudo systemctl enable --now kubelet
root@ip-172-31-85-68:/home/ubuntu# sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W1001 17:30:48.692533 5553 checks.go:1080] [preflight] WARNING: Couldn't create the int
d to create new CRI runtime service: validate service connection: validate CRI v1 runtime
.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 connec
[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 run
inerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeServ
make a check non-fatal with '--ignore-preflight-errors=...'
To see the stack trace of this error execute with --v=5 or higher
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

sudo apt-get install -y containerd

```
root@ip-172-31-85-68:/home/ubuntu# 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 requi
docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compo
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 143 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 amd
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd
Fetched 47.2 MB in 1s (53.2 MB/s)
(Reading database ... 68064 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 ... 68044 files and directories currently installed.)
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

```
sudo mkdir -p /etc/containerd
```

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

```
root@ip-172-31-85-68:/home/ubuntu# sudo mkdir -p /etc/containerd
root@ip-172-31-85-68:/home/ubuntu# 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"
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

```
sudo systemctl restart containerd
```

```
sudo systemctl enable containerd
```

```
sudo systemctl status containerd
```

```
root@ip-172-31-85-68:/home/ubuntu# sudo systemctl enable containerd
root@ip-172-31-85-68:/home/ubuntu# sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Tue 2024-10-01 17:32:39 UTC; 24s ago
     Docs: https://containerd.io
   Main PID: 5964 (containerd)
    Tasks: 8
   Memory: 13.6M (peak: 14.0M)
      CPU: 115ms
   CGroup: /system.slice/containerd.service
           └─5964 /usr/bin/containerd

Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.371130654Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.371164383Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.371202243Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.371227104Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.372111713Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.372136114Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.372144030Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.372151172Z" level=
Oct 01 17:32:39 ip-172-31-85-68 containerd[5964]: time="2024-10-01T17:32:39.372205611Z" level=
Oct 01 17:32:39 ip-172-31-85-68 systemd[1]: Started containerd.service - containerd container
lines 1-21/21 (END)
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

sudo apt-get install -y socat

```
root@ip-172-31-85-68:/home/ubuntu# 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
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docke
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 143 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
Fetched 374 kB in 0s (14.5 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

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

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68



Step 4 : Initialize the Kubecluster

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

```
root@ip-172-31-85-68:/home/ubuntu# sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet con
[preflight] You can also perform this action beforehand using 'kubeadm config images pul
W1001 17:34:36.917516 6240 checks.go:846] detected that the sandbox image "registry.k
with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the
[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-172-31-85-68 kubernetes kuber
vc.cluster.local] and IPs [10.96.0.1 172.31.85.68]
[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-172-31-85-68 localhost] and
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-85-68 localhost] and I
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

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

```
root@ip-172-31-85-68:/home/ubuntu# mkdir -p $HOME/.kube
root@ip-172-31-85-68:/home/ubuntu# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
root@ip-172-31-85-68:/home/ubuntu# sudo chown $(id -u):$(id -g) $HOME/.kube/config
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

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

kubectly apply -f

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

```
root@ip-172-31-85-68:/home/ubuntu# 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
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68



Step 5 :

Now that the cluster is up and running, we can deploy our nginx server on this cluster. Apply this deployment file using this command to create a deployment  
kubectl apply -f <https://k8s.io/examples/application/deployment.yaml>

```
root@ip-172-31-85-68:/home/ubuntu# kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
root@ip-172-31-85-68:/home/ubuntu#
```

kubectl get pods

```
root@ip-172-31-85-68:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-4pf7j    0/1     Pending   0           25s
nginx-deployment-d556bf558-dqs9g    0/1     Pending   0           25s
root@ip-172-31-85-68:/home/ubuntu#
```

POD\_NAME=\$(kubectl get pods -l app=nginx -o

jsonpath="{.items[0].metadata.name}")

kubectl port-forward \$POD\_NAME 8080:80

```
root@ip-172-31-85-68:/home/ubuntu# POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
root@ip-172-31-85-68:/home/ubuntu# kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
root@ip-172-31-85-68:/home/ubuntu#
```

We have faced an error as pod status is pending so make it running run below commands then again run above 2 commands.

kubectl taint nodes --all node-role.kubernetes.io/control-plane:ip-172-31-20-171  
untainted

kubectl get nodes

OR

kubectl taint nodes ip-172-31-85-68

node-role.kubernetes.io/control-plane:NoSchedule-

```
root@ip-172-31-85-68:/home/ubuntu# kubectl taint nodes --all node-role.kubernetes.io/control-plane:ip-172-31-85-68 untainted
error: at least one taint update is required
root@ip-172-31-85-68:/home/ubuntu#
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

Kubectrl get pods

```
root@ip-172-31-85-68:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-4pf7j    1/1     Running   0           10m
nginx-deployment-d556bf558-dqs9g    1/1     Running   0           10m
root@ip-172-31-85-68:/home/ubuntu#
```

i-063600dea9823b368 (Exp\_4\_61)

PublicIPs: 54.197.12.249 PrivateIPs: 172.31.85.68

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

kubectrl port-forward \$POD\_NAME 8080:80

```
root@ip-172-31-85-68:/home/ubuntu# POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
root@ip-172-31-85-68:/home/ubuntu# kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
^Croot@ip-172-31-85-68:/home/ubuntu# kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
[]
```

Step 6 : 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://127.0.0.1:8080>

```
root@ip-172-31-85-68:/home/ubuntu# curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Tue, 01 Oct 2024 17:53:16 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

root@ip-172-31-85-68:/home/ubuntu#
```

if the response is 200 OK and you can see the Nginx server name, your deployment was successful. We have successfully deployed our Nginx server on our EC2 instance.

**Conclusion:**

In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using Kubectl commands. We encountered two key challenges: first, the Kubernetes pod was initially in a pending state, which we resolved by removing the control-plane taint with the command `kubectl taint nodes --all`. Second, we faced an issue with the missing containerd runtime, which we fixed by installing and starting containerd. Utilising a t2.medium EC2 instance with 2 CPUs ensured that we met the necessary resource requirements for the Kubernetes setup and deployment.