

# Advanced DevOps Experiment - 3

Sanket More

D15A 30

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

## Implementation:-

### Creating Instance using Amazon Linux

#### 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](#)

[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

##### ▼ Summary

Number of instances [Info](#)

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...read more  
ami-0c2af51e265bd5e0e

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 on-demand

##### ▼ Instance type [Info](#) | [Get advice](#)

Instance type

t3.xlarge

Family: t3 4 vCPU 16 GiB Memory Current generation: true  
On-Demand Linux base pricing: 0.1728 USD per Hour  
On-Demand Windows base pricing: 0.2464 USD per Hour  
On-Demand SUSE base pricing: 0.2291 USD per Hour  
On-Demand RHEL base pricing: 0.2304 USD per Hour

☐ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

##### ▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

[Create new key pair](#)

## Create key pair



### Key pair name

Key pairs allow you to connect to your instance securely.

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

### Key pair type



RSA

RSA encrypted private and public key pair



ED25519

ED25519 encrypted private and public key pair

### Private key file format



.pem

For use with OpenSSH



.ppk

For use with PuTTY



When prompted, store the private key in a secure and accessible location on

Cancel

Create key pair

Instances (4) [Info](#)

Last updated  
less than a minute ago



Connect

Instance state ▼

Actions ▼

Launch instances ▼

Find Instance by attribute or tag (case-sensitive)

All states ▼

< 1 > ⚙

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
<input type="checkbox"/>	30SAN-env	i-06dae01fd646e93ba	Running	t3.micro	3/3 checks passed	<a href="#">View alarms</a> +	eu-north-1a	ec2-13-49-
<input type="checkbox"/>	Master	i-05e71d31a2d0b9a11	Stopped	t3.xlarge	-	<a href="#">View alarms</a> +	eu-north-1a	-
<input type="checkbox"/>	node1	i-0065bd5c032d2fda0	Stopped	t3.xlarge	-	<a href="#">View alarms</a> +	eu-north-1b	-
<input type="checkbox"/>	node2	i-095cb5930eba6bee4	Stopped	t3.xlarge	-	<a href="#">View alarms</a> +	eu-north-1b	-



# Installing Docker:-

eu-north-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0e5a240fd74d745f6&osUser=ec2-user&region=eu-north-1&sshPort...

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

```
[ec2-user@ip-172-31-47-115 ~]$ sudo su
[root@ip-172-31-47-115 ec2-user]# yum install docker -y
Last metadata expiration check: 0:09:05 ago on Thu Aug 29 08:35:03 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository
Installing: docker	x86_64	25.0.6-1.amzn2023.0.1	amazonlinux
Installing dependencies:			
containerd	x86_64	1.7.20-1.amzn2023.0.1	amazonlinux
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux
libcgroup	x86_64	3.0-1.amzn2023.0.1	amazonlinux

i-0e5a240fd74d745f6 (Master)

PublicIPs: 13.60.196.238 PrivateIPs: 172.31.47.115

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

86°F Mostly cloudy

eu-north-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=eu-north-1&connType=standard&instanceId=i-0410fb5b37129c459&osUser=ec2-user&sshPort...

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

```
[ec2-user@ip-172-31-33-3 ~]$ sudo su
[root@ip-172-31-33-3 ec2-user]# yum install docker -y
Last metadata expiration check: 0:09:03 ago on Thu Aug 29 08:37:13 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing: docker	x86_64	25.0.6-1.amzn2023.0.1	amazonlinux	44 M
Installing dependencies:				
containerd	x86_64	1.7.20-1.amzn2023.0.1	amazonlinux	35 M
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
libcgroup	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k

i-0410fb5b37129c459 (node-1)

PublicIPs: 13.60.191.4 PrivateIPs: 172.31.33.3

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

86°F Mostly cloudy

eu-north-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=eu-north-1&connType=standard&instanceId=i-09f9de7e33da75f69&osUser=ec2-user&sshPort...

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

```
[ec2-user@ip-172-31-40-242 ~]$ sudo su
[root@ip-172-31-40-242 ec2-user]# yum install docker -y
Last metadata expiration check: 0:09:26 ago on Thu Aug 29 08:39:24 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing: docker	x86_64	25.0.6-1.amzn2023.0.1	amazonlinux	44 M
Installing dependencies:				
containerd	x86_64	1.7.20-1.amzn2023.0.1	amazonlinux	35 M
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
libcgroup	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k

i-09f9de7e33da75f69 (node-2)

PublicIPs: 13.53.132.185 PrivateIPs: 172.31.40.242

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

86°F Mostly cloudy

# Installing Kubernetes:-

The screenshot shows the Kubernetes documentation page for installing kubeadm. The page is titled "Installing kubeadm" and is part of the "Production environment" section. The left sidebar contains a navigation menu with links to "Documentation", "Getting started", "Learning environment", "Production environment", "Container", "Runtimes", "Installing", "Kubernetes with deployment tools", "Bootstrapping clusters with kubeadm", and "Installing kubeadm". The main content area explains that the repository definition ensures that the packages related to Kubernetes are not upgraded upon running 'yum update' as there's a special procedure that must be followed for upgrading Kubernetes. It also notes that this repository has packages only for Kubernetes 1.31; for other Kubernetes minor versions, you need to change the Kubernetes minor version in the URL to match your desired minor version (you should also check that you are reading the documentation for the version of Kubernetes that you plan to install).

```
# This overwrites any existing configuration in /etc/yum.repos.d/kubern
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

3. Install kubelet, kubeadm and kubectl:

```
sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
```

Before you begin

- Verify the MAC address and product\_uuid are unique for every node
- Check network adapters
- Check required ports
- Installing a container runtime
- Installing kubeadm, kubelet and kubectl
- Configuring a cgroup driver
- Troubleshooting
- What's next

```
[root@ip-172-31-47-115 ec2-user]# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-47-115 ec2-user]# sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Kubernetes
Dependencies resolved.
16 kB/s | 6.5 kB 00:00
```

```
Complete!
[root@ip-172-31-47-115 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-47-115 ec2-user]#
```

i-0e5a240fd74d745f6 (Master)

PublicIPs: 13.60.196.238 PrivateIPs: 172.31.47.115

```

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  kubeadm-1.31.0-150500.1.1.x86_64
  kubelet-1.31.0-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
  cri-tools-1.31.1-150500.1.1.x86_64
  kubectl-1.31.0-150500.1.1.x86_64
  kubernetes-cni-1.5.0-150500.2.1.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  socat-1.7.4.2-1.amzn2023.0.2.x86_64

```

```

Complete!
[root@ip-172-31-33-3 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-33-3 ec2-user]#

```

i-0410fb5b37129c459 (node-1)

PublicIPs: 13.60.191.4 PrivateIPs: 172.31.33.3

```

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  kubeadm-1.31.0-150500.1.1.x86_64
  kubelet-1.31.0-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
  cri-tools-1.31.1-150500.1.1.x86_64
  kubectl-1.31.0-150500.1.1.x86_64
  kubernetes-cni-1.5.0-150500.2.1.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  socat-1.7.4.2-1.amzn2023.0.2.x86_64

```

```

Complete!
[root@ip-172-31-40-242 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-40-242 ec2-user]#

```

i-09f9de7e33da75f69 (node-2)

PublicIPs: 13.53.132.185 PrivateIPs: 172.31.40.242

## Initializing Kubeadm:-

```

[root@ip-172-31-37-74 ec2-user]# kubeadm init
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
      [WARNING FileExisting-tc]: tc 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'
W0911 06:06:02.983804 2545 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of t
y 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

```

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

```

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

```

Alternatively, if you are the root user, you can run:

```

export KUBECONFIG=/etc/kubernetes/admin.conf

```

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:  
<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

Then you can join any number of worker nodes by running the following on each as root:

```

kubeadm join 172.31.25.144:6443 --token kw6rp8.milvmgzveqzzeifs \
--discovery-token-ca-cert-hash sha256:e2285a0bb9324e4ebd564331961c506c7c8172e4639c81d5d89c86a47f0ad842
[root@ip-172-31-25-144 ec2-user]#

```

```

[root@master ~]# mkdir -p $HOME/.kube
[root@master ~]# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
[root@master ~]# sudo chown $(id -u):$(id -g) $HOME/.kube/config
[root@master ~]# export KUBECONFIG=/etc/kubernetes/admin.conf

```



## Creating the kubernetes nodes by copying the link in the workers:-

```
[root@worker ~]# systemctl enable docker
[root@worker ~]# systemctl enable kubelet
Created symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /usr/lib/systemd/system/kubelet.service.
[root@worker ~]# systemctl restart docker kubelet
[root@worker ~]# kubeadm join 172.31.46.167:6443 --token d47j51.2ejotl12hbxm5tys \
--discovery-token-ca-cert-hash sha256:a5511cfbd8b2b8bd69f15e7ad4247c34e1e2218b448a3a36509c0bb15b6f8d43_
```

## Installing the Calico file:-

The screenshot shows the Project Calico website. The left sidebar has a menu with 'About', 'Install Calico', and 'Kubernetes' (expanded to show 'Quickstart', 'Managed public cloud', 'Self-managed public cloud', and 'Self-managed on-premises' with 'Install Calico for on-premises' selected). The main content area is titled 'Based on your datastore and number of nodes, select a link below to install Calico.' It includes a note about Typha and a list of installation options. The first option, 'Install Calico with Kubernetes API datastore, 50 nodes or less', is selected, showing a list of steps. Step 1 is 'Download the Calico networking manifest for the Kubernetes API datastore.' Below this is a terminal snippet showing the curl command to download the manifest. Step 2 is partially visible, mentioning pod CIDR. The right sidebar has links for 'Big picture', 'Value', 'Concepts', 'Calico operator', 'Calico manifests', 'Before you begin...', 'How to', 'Install Calico', 'Install Calico with Kubernetes API datastore, 50 nodes or less', 'Install Calico with Kubernetes API datastore, more than 50 nodes', 'Install Calico with etcd datastore', and 'Next steps'.

```
[root@master ~]# curl https://raw.githubusercontent.com/projectcalico/calico/v3.24.1/manifests/calico.yaml -O
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 229k 100 229k 0 0 175k 0 0:00:01 0:00:01 --:--:-- 176k
[root@master ~]#
[root@master ~]# kubectl apply -f 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
```

Every 2.0s: kubectl get pods -A

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	calico-kube-controllers-58dbc876ff-nnprd	0/1	ContainerCreating	0	29s
kube-system	calico-node-bs95w	0/1	Init:2/3	0	29s
kube-system	calico-node-hplqt	0/1	Init:2/3	0	29s
kube-system	coredns-565d847f94-prsps	0/1	ContainerCreating	0	6m7s
kube-system	coredns-565d847f94-qrg48	0/1	ContainerCreating	0	6m6s
kube-system	etcd-master	1/1	Running	0	6m14s
kube-system	kube-apiserver-master	1/1	Running	0	6m12s
kube-system	kube-controller-manager-master	1/1	Running	0	6m11s
kube-system	kube-proxy-v2kxq	1/1	Running	0	2m31s
kube-system	kube-proxy-w8nn7	1/1	Running	0	6m7s
kube-system	kube-scheduler-master	1/1	Running	0	6m13s

## Nodes Created:

Every 2.0s: kubectl get nodes

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-85-89.ec2.internal	Ready	control-plane	3m39s	v1.26.0
ip-172-31-89-46.ec2.internal	Ready	<none>	119s	v1.26.0
ip-172-31-94-70.ec2.internal	Ready	<none>	112s	v1.26.0