

EXPERIMENT NO. 3

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

1. Launch three EC2 instances and assign unique names to each. Create and assign a key pair, ensuring SSH access is enabled.

Instances (3)
[Info](#)

Last updated

less than a minute ago

↻

Connect

Instance state ▾

Actions ▾

Launch instances

▾















All states ▾

<

1

>

⚙️

<input type="checkbox"/>	Name  ▾	Instance ID	Instance state  ▾	Instance type ▾	Status check	Alarm status	Availability zone
<input type="checkbox"/>	worker-2	i-0aa9f89768e3a199c	✔️ Running  	t2.micro	 Initializing	View alarms 	us-east-1
<input type="checkbox"/>	worker-1	i-05c2b3ba79c1d85ab	✔️ Running  	t2.micro	 Initializing	View alarms 	us-east-1
<input type="checkbox"/>	master	i-0bc867eb06b7964b7	✔️ Running  	t2.micro	 Initializing	View alarms 	us-east-1

2. SSH into each instance, open the terminal, and proceed to install Docker and Kubernetes.

```
#  
~\##### Amazon Linux 2023  
~~\#####\  
~~ \###|  
~~ \#/ https://aws.amazon.com/linux/amazon-linux-2023  
~~ V~' '->  
~~~~/  
~~.-./.  
~/m/'
```

[ec2-user@ip-172-31-80-190 ~]\$

3. Install Docker on all three instances. Follow the same installation steps for each one.

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-172-31-80-190 ec2-user]# sudo service docker status
Redirecting to /bin/systemctl status docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
   Active: active (running) since Fri 2024-09-13 04:23:51 UTC; 25s ago
 TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
  Main PID: 29009 (dockerd)
    Tasks: 7
   Memory: 29.6M
      CPU: 330ms
   CGroup: /system.slice/docker.service
           └─29009 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Sep 13 04:23:50 ip-172-31-80-190.ec2.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
Sep 13 04:23:50 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:50.998251424Z" level=info msg="Starting up"
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:51.068745150Z" level=info msg="Loading containers: start."
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:51.526546118Z" level=info msg="Loading containers: done."
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:51.555765635Z" level=info msg="Docker daemon" commit=b08a51f
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:51.556129762Z" level=info msg="Daemon has completed initialization"
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal dockerd[29009]: time="2024-09-13T04:23:51.598513281Z" level=info msg="API listen on /run/docker.sock"
Sep 13 04:23:51 ip-172-31-80-190.ec2.internal systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-20/20 (END)
```

4. For Kubernetes installation, use kubeadm. Fetch the necessary installation commands and complete the setup on all instances.

```

Installing      : kubeadm-1.31.1-150500.1.1.x86_64      8/9
Installing      : kubectl-1.31.1-150500.1.1.x86_64      9/9
Running scriptlet: kubectl-1.31.1-150500.1.1.x86_64      9/9
Verifying       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 1/9
Verifying       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 2/9
Verifying       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 3/9
Verifying       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64 4/9
Verifying       : cri-tools-1.31.1-150500.1.1.x86_64      5/9
Verifying       : kubeadm-1.31.1-150500.1.1.x86_64      6/9
Verifying       : kubectl-1.31.1-150500.1.1.x86_64      7/9
Verifying       : kubelet-1.31.1-150500.1.1.x86_64      8/9
Verifying       : kubernetes-cni-1.5.1-150500.1.1.x86_64 9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  kubeadm-1.31.1-150500.1.1.x86_64
  kubelet-1.31.1-150500.1.1.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-1.31.1-150500.1.1.x86_64
  kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-83-1 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-172-31-83-1 ec2-user]#

```

5. Verify the repositories are correctly configured

```

[root@ip-172-31-83-1 ec2-user]# yum repolist
repo id                                repo name
amazonlinux                            Amazon Linux 2023 repository
kernel-livepatch                       Amazon Linux 2023 Kernel Livepatch repository
kubernetes                             Kubernetes
[root@ip-172-31-83-1 ec2-user]#

```

6. Run the kubeadm command to initialize Kubernetes.

```

[root@ip-172-31-83-1 ec2-user]# kubeadm init
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[WARNING FileExisting-socat]: socat not found in system path
[WARNING FileExisting-tc]: tc not found in system path
error execution phase preflight: [preflight] Some fatal errors occurred:
[ERROR NumCPU]: the number of available CPUs 1 is less than the required 2
[ERROR Mem]: the system RAM (949 MB) is less than the minimum 1700 MB
[preflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=NumCPU'
To see the stack trace of this error execute with --v=5 or higher
[root@ip-172-31-83-1 ec2-user]#

```