

Experiment -3.1

Student Name: Ayush Pandey

Branch: CSE-DevOps

Semester: 5

Subject Name: Docker and Kubernetes

UID: 22BDO10038

Section/Group: 22BCD-1(A)

Date of Performance: 21-10-24

Subject Code: 22CSH-343

1. Aim/Overview of the practical:

Installing Kubernetes as a Single Node.

2. Apparatus: PC, Docker Engine, Kubernetes, Minikube, Ubuntu Linux

3. Steps for experiment/practical:

- To install the latest minikube stable release on x86-64 Linux using binary download:
 1. curl -LO
<https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64>
 2. sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm
minikube-linux-amd64

```
ayush@Linux:~$ curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kube
rnetes-release/release/stable.txt`/bin/linux/amd64/kubectl
% Total % Received % Xferd Average Speed Time Time Current
Dload Upload Total Spent Left Speed
100 53.7M 100 53.7M 0 0 407k 0 0:02:15 0:02:15 ---:--: 1065k
ayush@Linux:~$ chmod +x kubectl
ayush@Linux:~$ sudo mv kubectl /usr/local/bin/
ayush@Linux:~$ kubectl version -o yaml
clientVersion:
  buildDate: "2024-08-13T07:37:34Z"
  compiler: gc
  gitCommit: 9edcffcde5595e8a5b1a35f88c421764e575afce
  gitTreeState: clean
  gitVersion: v1.31.0
  goVersion: go1.22.5
  major: "1"
  minor: "31"
  platform: linux/amd64
kustomizeVersion: v5.4.2

The connection to the server localhost:8080 was refused - did you specify the right host or port?
```

```
ayush@Linux:~$ docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu2~22.04.1
ayush@Linux:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 99.0M  100 99.0M    0     0  676k      0  0:02:29  0:02:29 --:--:--  546k
ayush@Linux:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
[sudo] password for ayush:
ayush@Linux:~$ minikube version
minikube version: v1.34.0
commit: 210b148df93a80eb872ecbeb7e35281b3c582c61
```

- From a terminal with administrator access (but not logged in as root), run:
 - minikube start

```
ayush@Linux:~$ minikube start --driver=docker
minikube v1.34.0 on Ubuntu 22.04 (vbox/amd64)
Using the docker driver based on user configuration
Using Docker driver with root privileges
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.45 ...
Downloading Kubernetes v1.31.0 preload ...
> preloaded-images-k8s-v18-v1...: 326.69 MiB / 326.69 MiB 100.00% 1.95 Mi
> gcr.io/k8s-minikube/kicbase...: 487.89 MiB / 487.90 MiB 100.00% 2.38 Mi
Creating docker container (CPUs=2, Memory=2200MB) ...
Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
  ■ Generating certificates and keys ...
  ■ Booting up control plane ...
  ■ Configuring RBAC rules ...
Configuring bridge CNI (Container Networking Interface) ...
Verifying Kubernetes components...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: storage-provisioner, default-storageclass
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

- minikube can download the appropriate version of kubectl and you should be able to use it like this:
 - minikube kubectl -- get po -A

```
ayush@Linux:~$ kubectl get nodes
NAME        STATUS    ROLES    AGE   VERSION
minikube    Ready     control-plane 68s   v1.31.0
ayush@Linux:~$ kubectl get nodes
NAME        STATUS    ROLES    AGE   VERSION
minikube    Ready     control-plane 97s   v1.31.0
ayush@Linux:~$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.58.2:8443
CoreDNS is running at https://192.168.58.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

- Initially, some services such as the storage-provisioner, may not yet be in a Running state. This is a normal condition during cluster bring-up, and will resolve itself momentarily. For additional insight into your cluster state, minikube bundles the Kubernetes Dashboard, allowing you to get easily acclimated to your new environment:
 - minikube dashboard

```
ayush@Linux:~$ minikube dashboard
🐼 Verifying dashboard health ...
🚀 Launching proxy ...
🐼 Verifying proxy health ...
🌐 Opening http://127.0.0.1:36271/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
Gtk-Message: 12:27:45.323: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
```

- Create a sample deployment and expose it on port 80:
 - kubectl create deployment nginx-web --image=nginx
 - kubectl expose deployment nginx-web --type=NodePort --port=80

```
ayush@Linux:~$ kubectl create deployment nginx-web --image=nginx
deployment.apps/nginx-web created
ayush@Linux:~$ kubectl expose deployment nginx-web --type NodePort --port=80
service/nginx-web exposed
ayush@Linux:~$ kubectl get deployment,pod,svc
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/nginx-web	1/1	1	1	73s

NAME	READY	STATUS	RESTARTS	AGE
pod/nginx-web-5cb57cfb4b-4rsr6	1/1	Running	0	73s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	3m50s
service/nginx-web	NodePort	10.100.118.209	<none>	80:32651/TCP	22s

- The easiest way to access this service is to let minikube launch a web browser for you:
 - minikube service nginx-web

The screenshot shows the Kubernetes Dashboard interface. The top navigation bar includes the 'Workloads' menu. The left sidebar lists various Kubernetes resources: Workloads (N), Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses (N), Ingress Classes, Services (N), Config and Storage, Config Maps (N), Persistent Volume Claims (N), Secrets (N), and Storage Classes. The main content area displays 'Workload Status' with three green circles representing the status of Deployments, Pods, and Replica Sets, each with a 'Running: 1' label. Below this, the 'Deployments' table shows a single deployment named 'nginx-web' using the 'nginx' image, with 1 pod running, created 8 minutes ago. The 'Pods' table is also visible, showing columns for Name, Images, Labels, Node, Status, Restarts, CPU Usage (cores), Memory Usage (bytes), and Created.

● Management of clusters and pods

```
ayush@Linux:~$ minikube stop
Stopping node "minikube" ...
Powering off "minikube" via SSH ...
1 node stopped.
ayush@Linux:~$ minikube start
minikube v1.34.0 on Ubuntu 22.04 (vbox/amd64)
Using the docker driver based on existing profile
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.45 ...
Restarting existing docker container for "minikube" ...
Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
Verifying Kubernetes components...
■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
■ Using image registry.k8s.io/ingress-nginx/controller:v1.11.2
■ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
■ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
■ Using image docker.io/kubernetes/dashboard:v2.7.0
■ Using image docker.io/kubernetes/metrics-scraper:v1.0.8
Verifying ingress addon...
Some dashboard features require the metrics-server addon. To enable all features please run:

    minikube addons enable metrics-server

Enabled addons: storage-provisioner, dashboard, default-storageclass, ingress
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
ayush@Linux:~$
```

Learning outcomes (What I have learnt):

1. I have learnt the concept of containerization and virtualization.
2. I have learnt about orchestration and orchestration tools.
3. I have learnt about Kubernetes and its architecture.
4. I have learnt the purpose of using microservice architecture over monolithic.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			