MiniKube and pod

Minikube:

- It is a tool used to setup single node cluster on K8's.
- It contains API Servers, ETDC database and container runtime
- It helps you to containerized applications.
- It is used for development, testing, and experimentation purposes on local.
- Here Master and worker runs on same machine It is a platform Independent.
- By default it will create one node only.
- Installing MiniKube is simple compared to other tools.
- NOTE: But we don't implement this in real-time

MINIKUBE SETUP:

REQUIREMENTS:

- ✓ 2 CPUs or more
- ✓ 2GB of free memory
- ✓ 20GB of free disk space
- ✓ Internet connection
- ✓ Container or virtual machine manager, such as: Docker.

🖥 1. Update the Server

Update all packages and upgrade to the latest versions:

sudo apt update -y

sudo apt upgrade -y

🐳 2. Install Docker

Install necessary packages and install Docker using the official install script:

sudo apt install curl wget apt-transport-https -y

sudo curl -fsSL https://get.docker.com -o get-docker.sh

sudo sh get-docker.sh

Check Docker installation:

docker --version

Add your user to the Docker group to avoid using sudo with Docker:

sudo usermod -aG docker $USER

newgrp docker

```
ubuntu@kiran:~$ sudo docker version
Client: Docker Engine - Community
 Version:           28.1.1
 API version:       1.49
 Go version:        go1.23.8
 Git commit:        4eba377
 Built:             Fri Apr 18 09:52:14 2025
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          28.1.1
  API version:      1.49 (minimum version 1.24)
  Go version:       go1.23.8
  Git commit:       01f442b
  Built:            Fri Apr 18 09:52:14 2025
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
  Version:          1.2.5
  GitCommit:        v1.2.5-0-g59923ef
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
ubuntu@kiran:~$
```

📦 3. Install Minikube

Download the latest Minikube binary and move it to your system path:

sudo curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
ubuntu@ip-172-31-34-215:~$ 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  119M  100  119M    0     0  11.6M      0  0:00:10  0:00:10 --:--:-- 16.1M
ubuntu@ip-172-31-34-215:~$
```

sudo mv minikube-linux-amd64 /usr/local/bin/minikube

sudo chmod +x /usr/local/bin/minikube

Verify Minikube installation:

minikube version

```
ubuntu@kiran:~$ minikube version
minikube version: v1.35.0
commit: dd5d320e41b5451cdf3c01891bc4e13d189586ed-dirty
ubuntu@kiran:~$
```

☸️ 4. Install kubectl (Kubernetes CLI)

Download and install the latest stable version of kubectl:

sudo curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

sudo curl -LO https://dl.k8s.io/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256

Verify the integrity of the binary:

echo "$(cat kubectl.sha256) kubectl" | sha256sum --check

Install kubectl:

sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

```
ubuntu@kiran:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
curl -LO "https://dl.k8s.io/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
echo "$(cat kubectl.sha256) kubectl" | sha256sum --check
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0    454      0 --:--:-- --:--:-- --:--:--   453
100 57.3M  100 57.3M    0     0  16.8M      0  0:00:03  0:00:03 --:--:-- 20.1M
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0    448      0 --:--:-- --:--:-- --:--:--   448
100    64  100    64    0     0    185      0 --:--:-- --:--:-- --:--:--   185
kubectl: OK
ubuntu@kiran:~$
```

Check kubectl version:

kubectl version --client

kubectl version --client --output=yaml

```
ubuntu@kiran:~$ kubectl version --client
Client Version: v1.33.0
Kustomize Version: v5.6.0
ubuntu@kiran:~$
```

🟢 5. Start Minikube with Docker Driver

Start Minikube using Docker as the container runtime:

minikube start --driver=docker --force

🔷 Note: If you skipped adding your user to the Docker group, you may need to run the above with sudo.

```
ubuntu@kiran:~$ minikube start --driver=docker
* minikube v1.35.0 on Ubuntu 24.04 (xen/amd64)
* Using the docker driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.46 ...
* Updating the running docker "minikube" container ...
* Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
* 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
ubuntu@kiran:~$ kubectl get pods
No resources found in default namespace.
ubuntu@kiran:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

ubuntu@kiran:~$
```

✅ Final Checks

Ensure everything is running:

kubectl get nodes

minikube status

```
ubuntu@kiran:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

ubuntu@kiran:~$
```

📌 Notes

- This setup is tested on Ubuntu 20.04+ (should also work on Debian-based distros).
- You must have a system that supports virtualization (for Minikube to work).
- Docker must be running before starting Minikube.

KUBECTL:

kubectl is the CLI which is used to interact with a Kubernetes cluster.

We can create, manage pods, services, deployments, and other resources

We can also monitoring, troubleshooting, scaling and updating the pods.

To perform these tasks it communicates with the Kubernetes API server.

It has many options and commands, to work on.

The configuration of kubectl is in the $HOME/.kube directory.

SYNTAX:

kubectl [command] [TYPE] [NAME] [flags]

POD:

It is a smallest unit of deployment in K8's.

It is a group of containers.

Pods are ephemeral (short living objects)

Mostly we can use single container inside a pod but if we required, we can create multiple containers inside a same pod.

when we create a pod, containers inside pods can share the same network namespace, and can share the same storage volumes .

While creating pod, we must specify the image, along with any necessary configuration and resource limits.

K8's cannot communicate with containers, they can communicate with only pods.

We can create this pod in two ways,

1. Imperative(command)
2. Declarative (Manifest file)

POD CREATION:

IMPERATIVE:

The imperative way uses kubectl command to create pod.

This method is useful for quickly creating and modifying the pods.

SYNTAX: kubectl run pod_name --image=image_name

COMMAND: kubectl run pod-1 --image=nginx

```
ubuntu@kiran:~$ kubectl run pod-1 --image=nginx
pod/pod-1 created
ubuntu@kiran:~$ kubectl get pod
NAME                                READY   STATUS              RESTARTS   AGE
hello-minikube-ffcbb5874-xp6c9      1/1     Running             0          13m
hellow-minikube-74ffd7cc95-pmq92    1/1     Running             0          13m
pod-1                               0/1     ContainerCreating   0          6s
ubuntu@kiran:~$
```

kubect1 : command line tool

run : action

pod-1 : name of pod

nginx : name of image

DECLARATIVE:

The Declarative way we need to create a Manifest file in YAML Extension.

This file contains the desired state of a Pod.

It takes care of creating, updating, or deleting the resources.

This manifest file need to follow the yaml indentation.

YAML file consist of KEY-VALUE Pair.

Here we use create or apply command to execute the Manifest file.

nginx-pod.yml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 80
```

SYNTAX: kubectl create/apply -f file_name

```
ubuntu@kiran:~$ kubectl apply -f nginx-pod.yml
pod/nginx-pod created
ubuntu@kiran:~$ kubectl get pod
NAME                                READY   STATUS    RESTARTS   AGE
hello-minikube-ffcbb5874-xp6c9      1/1     Running   0          15m
hellow-minikube-74ffd7cc95-pmq92    1/1     Running   0          15m
nginx-pod                           1/1     Running   0          8s
pod-1                               1/1     Running   0          2m31s
```

CREATE: if you are creating the object for first time we use create only.

APPLY: if we change any thing on files and changes need to apply the resources.

apiVersion: For communicating with master node

Kind: it is a type of resource

Metadata: data about pod

Spec: it is a specifications of a container

Name: Name of the Container

Image: Conatiner image

Ports: To Expose the Application

- : It is called as Array