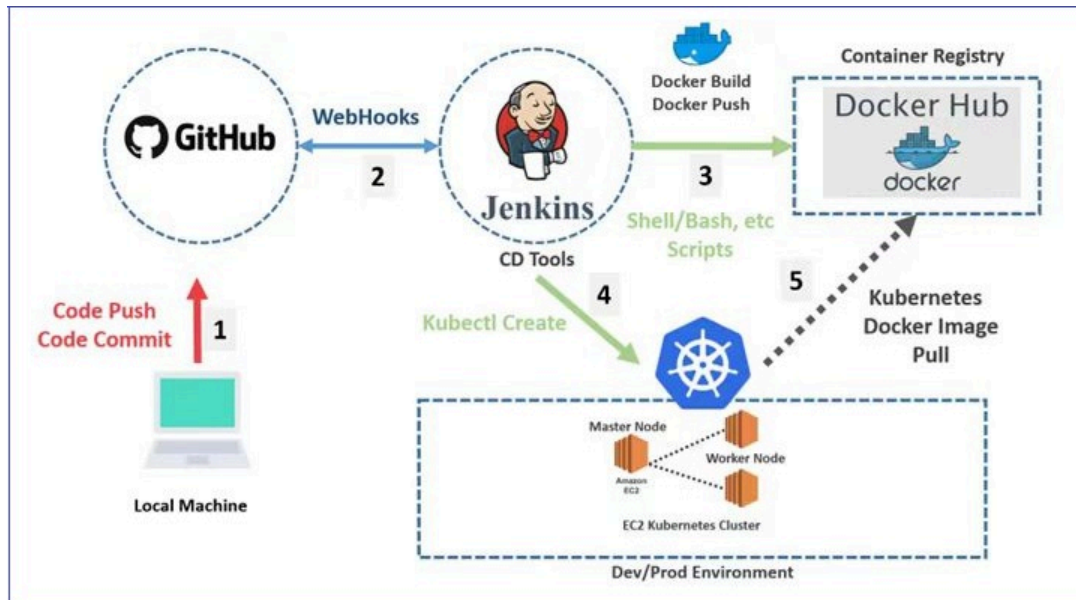


[CS304] Tutorial 13 - Introduction of Kubernetes

Kubernetes (K8s) enables us to manage a cluster of containers. Today's lesson covers very basic aspects of k8s.



In the framework above, k8s is a critical tool in step 4 and step 5. If you already have an image on docker-hub, you can deploy it using k8s.

Installation

We need to at least install two software [Install Tools](#). One is the command line utility `kubectl`, the other is the K8s service provider.

Install the former as in [Install and Set Up kubectl on Linux](#).

The latter has three choices, today we choose `minikube`. Install `minikube` as in [minikube start](#). Only do the first step. The following steps will be covered in the following sections. You are free to choose the other choices.

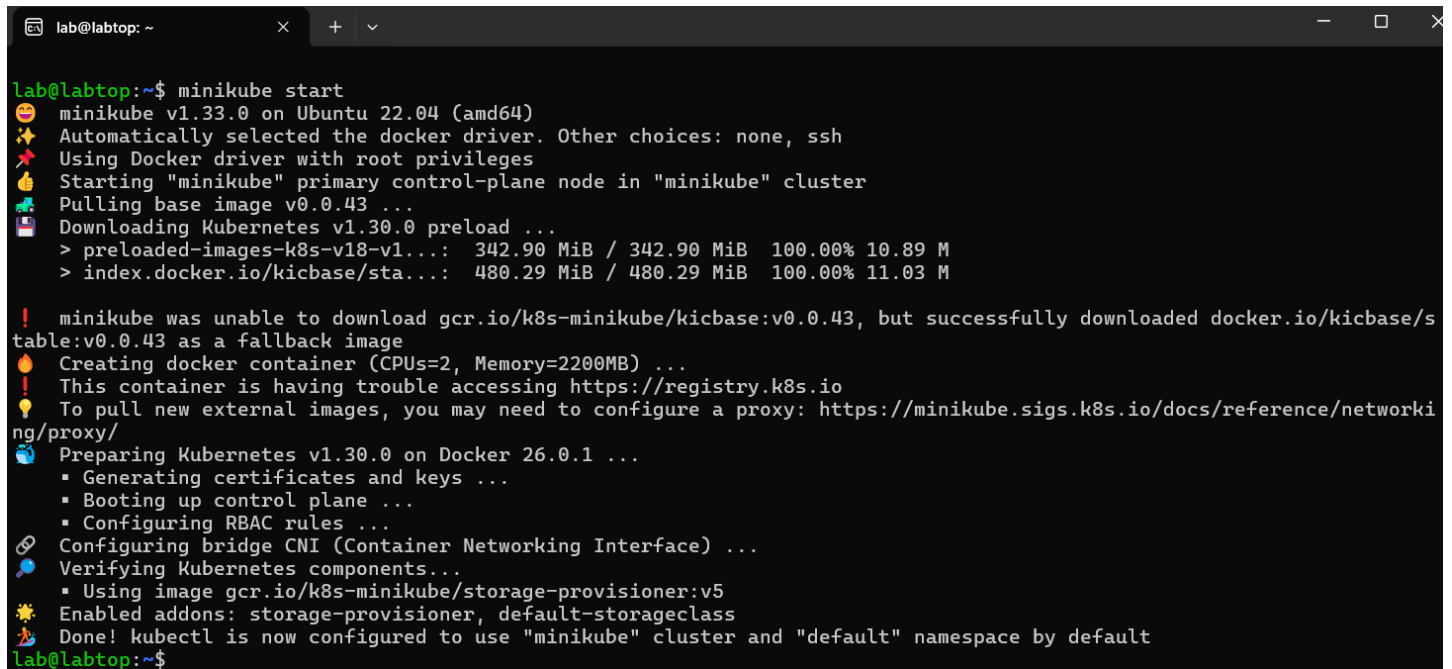
K8s hello world

K8s works like this: you create a cluster with Kubernetes (minikube this time) to run containers, and use `kubectl` to control them. A cluster consists one or more `node`, where each `node` is a physical or virtual machine to run containers.

Now let's begin with [Hello Minikube](#).

Create a minikube cluster

```
minikube start
```



```
lab@labtop: ~$ minikube start
minikube v1.33.0 on Ubuntu 22.04 (amd64)
✨ Automatically selected the docker driver. Other choices: none, ssh
🔧 Using Docker driver with root privileges
👉 Starting "minikube" primary control-plane node in "minikube" cluster
📦 Pulling base image v0.0.43 ...
📦 Downloading Kubernetes v1.30.0 preload ...
> preloaded-images-k8s-v18-v1...: 342.90 MiB / 342.90 MiB 100.00% 10.89 M
> index.docker.io/kicbase/sta...: 480.29 MiB / 480.29 MiB 100.00% 11.03 M

! minikube was unable to download gcr.io/k8s-minikube/kicbase:v0.0.43, but successfully downloaded docker.io/kicbase/s
table:v0.0.43 as a fallback image
🔥 Creating docker container (CPUs=2, Memory=2200MB) ...
! This container is having trouble accessing https://registry.k8s.io
💡 To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networki
ng/proxy/
🔧 Preparing Kubernetes v1.30.0 on Docker 26.0.1 ...
  ▪ 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
lab@labtop:~$
```

Open the Dashboard

Open a new terminal, and run:

```
minikube dashboard
```

Then leave this terminal alone. If you failed to start the dashboard, don't panic, just continue to the next section.

Create a Deployment

Switch back to the first terminal and continue with following steps.

1. Create a Deployment with one Pod. A Pod is the smallest unit K8s manages that contains a set of containers. Note that we have specified the image we want to run in the command. Here `sismics/docs:v1.11` is Teedy's official image. You can also use your own image instead of Teedy's official image.

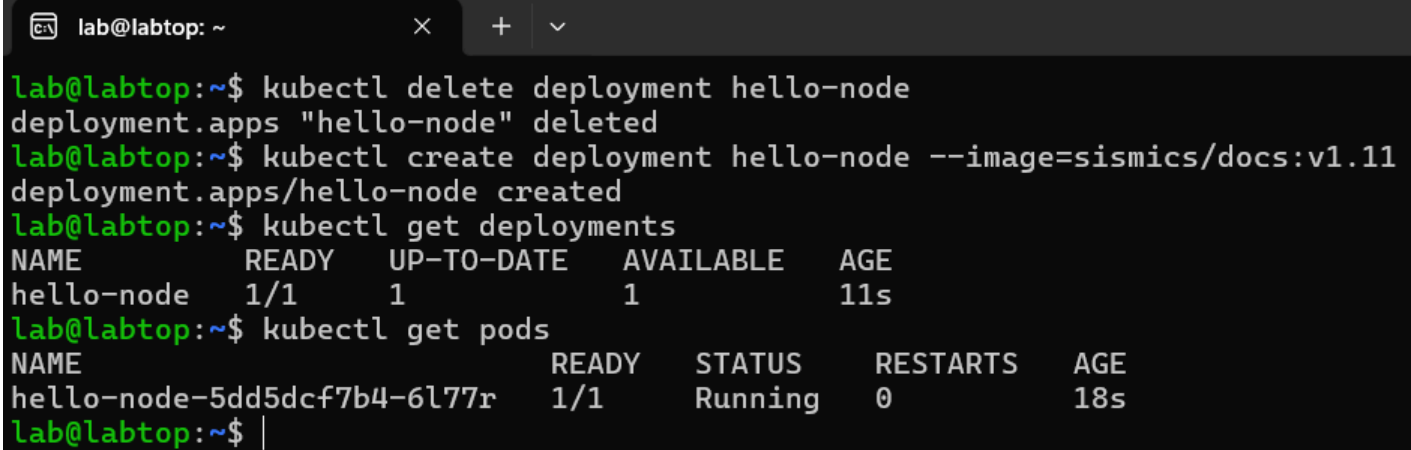
```
kubectl create deployment hello-node --image=sismics/docs:v1.11
```

2. View the Deployment:

```
kubectl get deployments
```

3. View the Pod:

```
kubectl get pods
```



```
lab@labtop: ~  
lab@labtop:~$ kubectl delete deployment hello-node  
deployment.apps "hello-node" deleted  
lab@labtop:~$ kubectl create deployment hello-node --image=sismics/docs:v1.11  
deployment.apps/hello-node created  
lab@labtop:~$ kubectl get deployments  
NAME          READY   UP-TO-DATE   AVAILABLE   AGE  
hello-node    1/1     1            1           11s  
lab@labtop:~$ kubectl get pods  
NAME                                READY   STATUS    RESTARTS   AGE  
hello-node-5dd5dcf7b4-6l77r        1/1     Running   0          18s  
lab@labtop:~$ |
```

4. View cluster events:

```
kubectl get events
```

5. View the kubectl configuration:

```
kubectl config view
```

6. View application logs for a container in a pod (replace pod name with the name returned by step 3).

```
kubectl logs hello-node-5dd5dcf7b4-6l77r
```

If you see satisfying result here, which means you see "Running" status when you do "kubectl get pods", you can proceed with "Create a Service". But some of you may see errors like "ImagePullBackOff" status due to minikube's not able to download sismic/docs image. Do the following if necessary:

```
docker pull sismics/docs:v1.11  
minikube image load sismics/docs:v1.11
```

Then delete the deployment and recreate it:

```
kubectl delete deployment hello-node
kubectl create deployment hello-node --image=sismics/docs:v1.11
```

Create a Service

However, the running container can only be accessed through a K8s virtual network. We need to create a service for it.

1. Expose the Pod to the public internet using the kubectl expose command:

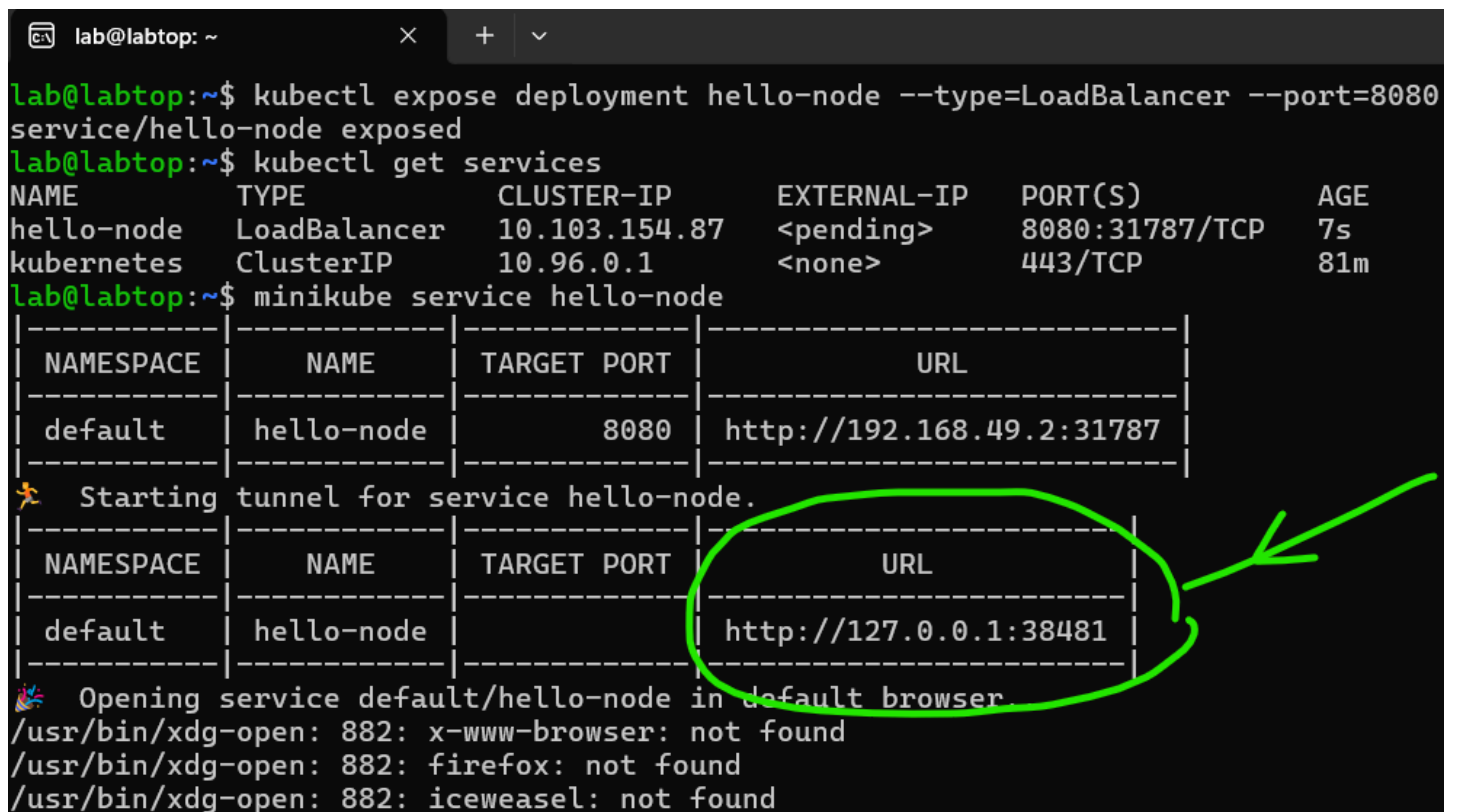
```
kubectl expose deployment hello-node --type=LoadBalancer --port=8080
```

2. View the Service you created:

```
kubectl get services
```

3. Run the following command to show your application in the browser:

```
minikube service hello-node
```



```
lab@labtop: ~
lab@labtop:~$ kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed
lab@labtop:~$ kubectl get services
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
hello-node    LoadBalancer 10.103.154.87  <pending>      8080:31787/TCP   7s
kubernetes    ClusterIP      10.96.0.1     <none>         443/TCP          81m
lab@labtop:~$ minikube service hello-node
|-----|
| NAMESPACE | NAME      | TARGET PORT | URL                               |
|-----|
| default   | hello-node | 8080        | http://192.168.49.2:31787       |
|-----|
🚀 Starting tunnel for service hello-node.
|-----|
| NAMESPACE | NAME      | TARGET PORT | URL                               |
|-----|
| default   | hello-node |             | http://127.0.0.1:38481         |
|-----|
🌐 Opening service default/hello-node in default browser.
/usr/bin/xdg-open: 882: x-www-browser: not found
/usr/bin/xdg-open: 882: firefox: not found
/usr/bin/xdg-open: 882: iceweasel: not found
```

Sometimes it fails to start the browser. Then you open your browser manually, and copy the link into your browser.

Update your app

Suppose you usually use Jenkins to manage your project. You have just pushed your code to Github. Then Github action run Jenkins and push new Docker image to DockerHub. Then what you can do with your local machine with K8s running the old image on it?

1. Tell your deployments to change to a new image. Note that "docs" is the container name here. If you don't know your container name, you can run "kubectl describe pod your-pod-id"

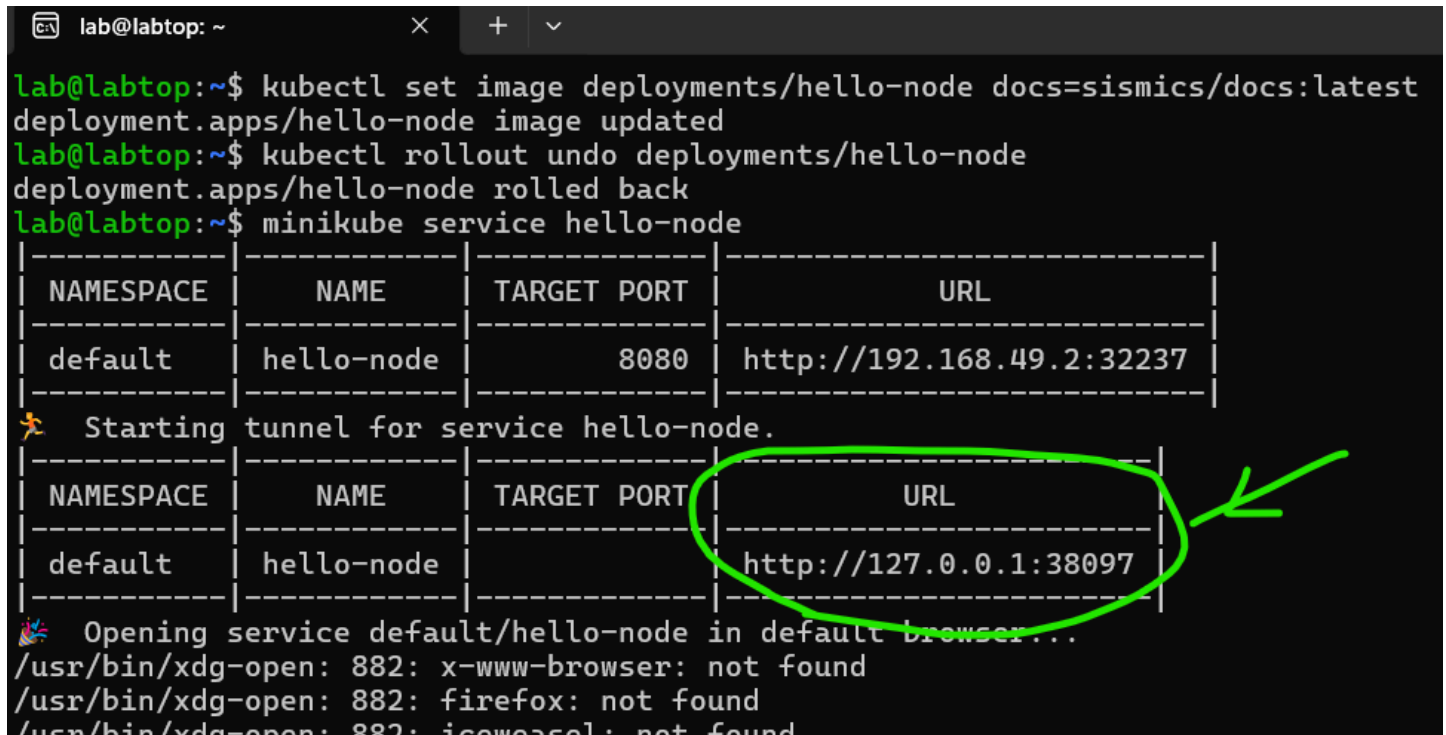
```
kubectl set image deployments/hello-node docs=sismics/docs:latest
```

2. If you are not satisfied with the new version, you can roll back:

```
kubectl rollout undo deployments/hello-node
```

3. After rollback, I failed to access Teedy from web browser, I have to redo create service:

```
minikube service hello-node
```



Kubernetes with Jenkins

Now we want to update K8s each time we run our Jenkins pipeline. Update our Jenkins file from lab 11:

```

pipeline {
    agent any

    environment {
        DEPLOYMENT_NAME = "your-deployment"
        CONTAINER_NAME = "your-container"
        IMAGE_NAME = "your-dockerhub-id/your-image:version"
    }

    stages {
        stage('Start Minikube') {
            steps {
                sh '''
                    if ! minikube status | grep -q "Running"; then
                        echo "Starting Minikube..."
                        minikube start
                    else
                        echo "Minikube already running."
                    fi
                '''
            }
        }

        stage('Set Image') {
            steps {
                sh '''
                    echo "Setting image for deployment..."
                    kubectl set image deployment/${DEPLOYMENT_NAME} ${CONTAINER_NAME}=${IMAGE_NAME}
                '''
            }
        }

        stage('Verify') {
            steps {
                sh 'kubectl rollout status deployment/${DEPLOYMENT_NAME}'
                sh 'kubectl get pods'
            }
        }
    }
}

```

Remember to modify your container name and image id. Jenkins will update your K8s image each time we run the pipeline.

There are plenty of features to explore with Kubernetes plugin for Jenkins: [Kubernetes plugin for Jenkins](#). Typically, a complete CI/CD system involves DockerHub, Kubernetes, Jenkins and Github all together, but today we only deal with K8s. You can think about how to put them together.

Clean up

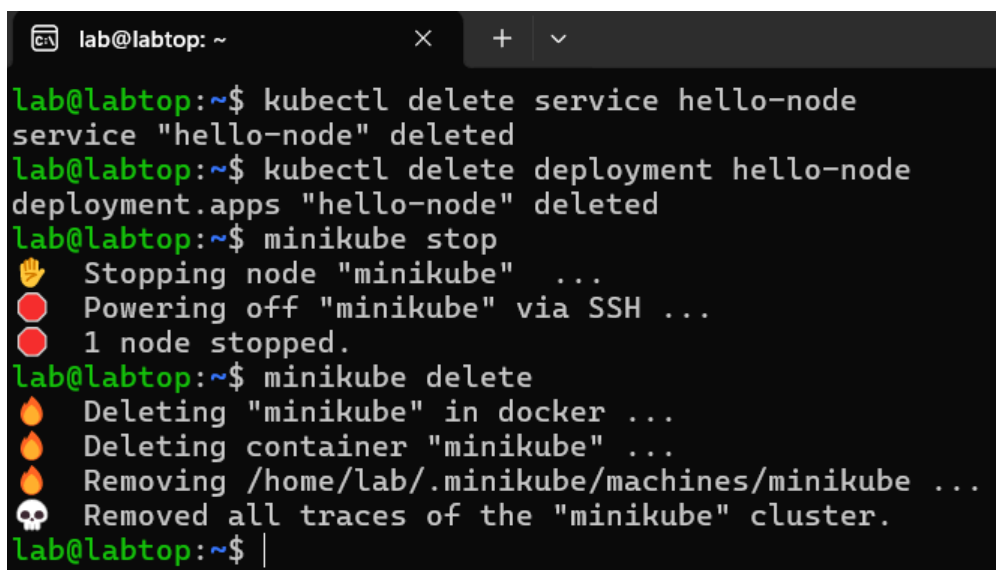
Now that you have successfully run Teedy project in the browser. Let's clean it up.

1. Remove the service and the deployment. Stop minikube.

```
kubectl delete service hello-node
kubectl delete deployment hello-node
minikube stop
```

2. If you don't use it again today, delete the virtual cluster.

```
minikube delete
```

A terminal window titled 'lab@labtop: ~' with window control buttons (close, maximize, minimize) in the title bar. The terminal shows the following commands and output:

```
lab@labtop:~$ kubectl delete service hello-node
service "hello-node" deleted
lab@labtop:~$ kubectl delete deployment hello-node
deployment.apps "hello-node" deleted
lab@labtop:~$ minikube stop
👋 Stopping node "minikube" ...
🔴 Powering off "minikube" via SSH ...
🔴 1 node stopped.
lab@labtop:~$ minikube delete
🔥 Deleting "minikube" in docker ...
🔥 Deleting container "minikube" ...
🔥 Removing /home/lab/.minikube/machines/minikube ...
💀 Removed all traces of the "minikube" cluster.
lab@labtop:~$ |
```