Create an Ingress Resource for External Access to Your Application

Table of Contents

Introduction

In Kubernetes, an **Ingress** resource defines rules that allow external access to services within a cluster. It acts as a gateway, managing traffic and routing requests based on defined rules. In this lab, you will set up an Ingress resource in a Minikube environment to provide external access to a sample web application.

Problem Statement

As applications within a Kubernetes cluster grow, the need for external access becomes paramount. Traditional methods, such as NodePort Services, can expose services, but they lack advanced routing capabilities. Ingress solves this problem by providing a single entry point to your applications, allowing for better management and easier access.

Prerequisites

Completion of all previous lab guides (up to Lab Guide-06) is required before proceeding with Lab Guide-07.

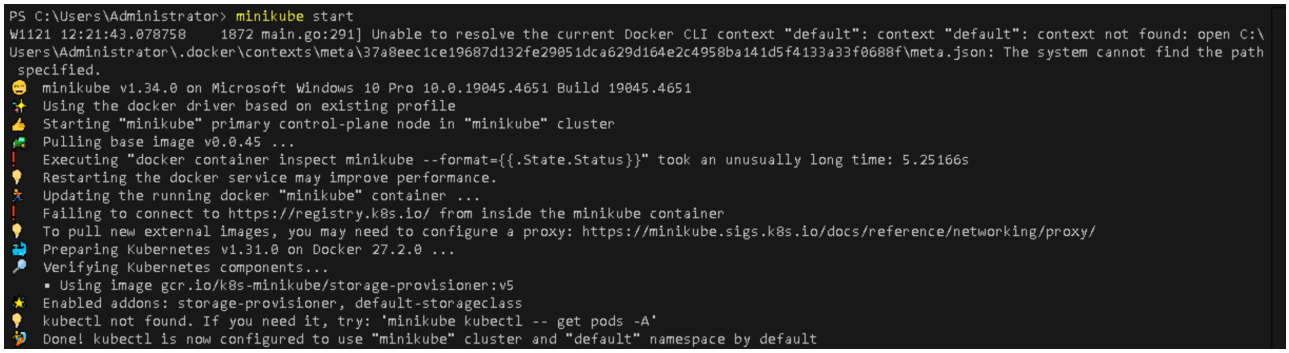
- **Minikube** installed on your Windows machine.
- **kubectl** command-line tool installed and configured to communicate with your Minikube cluster.
- Ensure your Minikube version is compatible (at least Kubernetes v1.19).

Setup Instructions

Step 1: Create a Minikube Cluster

1. **Start Minikube**
   Open PowerShell or Command Prompt as Administrator and run:

```
minikube start
```

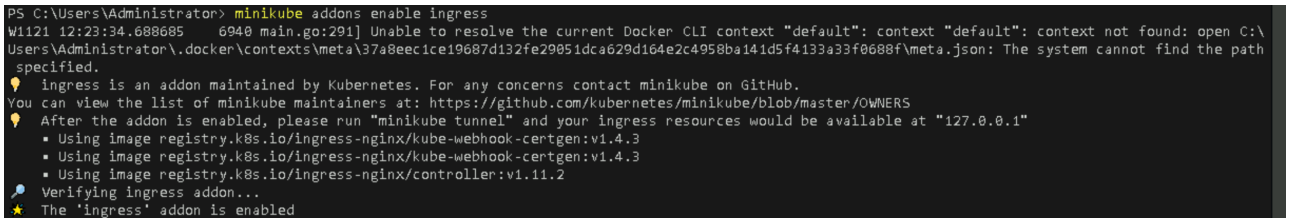


This command will create a local Kubernetes cluster.

Step 2: Enable the Ingress Controller

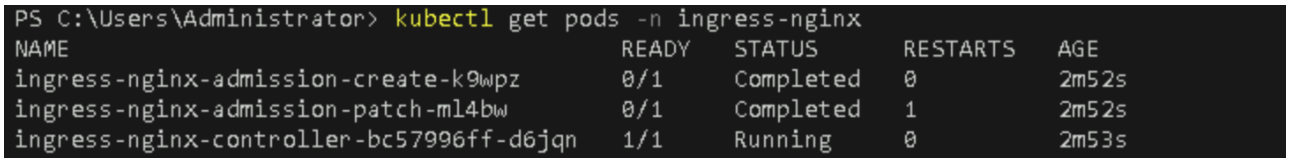
1. **Enable NGINX Ingress Controller**
   Run the following command in your terminal:

```
minikube addons enable ingress
```



2. **Verify the Ingress Controller is Running**
   After a few moments, verify that the Ingress controller is running with:

```
kubectl get pods -n ingress-nginx
```



You should see the NGINX Ingress controller listed and running.

Step 3: Deploy a Hello World Application

1. **Create a Deployment**
   Deploy a sample Hello World application:

```
kubectl create deployment web --image=gcr.io/google-samples/hello-app:1.0
```

```
PS C:\Users\Administrator> kubectl create deployment web --image=gcr.io/google-samples/hello-app:1.0
deployment.apps/web created
```

Confirm the deployment is successful by running:

```
kubectl get deployment web
```

```
PS C:\Users\Administrator> kubectl get deployment web
NAME   READY   UP-TO-DATE   AVAILABLE   AGE
web    0/1     1            0           30s
```

2. **Expose the Deployment**
   Expose the deployment with NodePort:

```
kubectl expose deployment web --type=NodePort --port=8080
```

```
PS C:\Users\Administrator> kubectl expose deployment web --type=NodePort --port=8080
service/web exposed
```

3. **Verify the Service**
   Check the service to ensure it's available:

```
kubectl get svc web
```

```
PS C:\Users\Administrator> kubectl get svc web
NAME   TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
web    NodePort   10.107.128.213   <none>        8080:30308/TCP   4m11s
```

Note the `NodePort` assigned to the service.

Step 4: Create the Ingress Resource

1. **Create an Ingress YAML File**
   Create a file named `example-ingress.yaml` with the following content:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
```

```
    - host: hello-world.example
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 8080
```

2. **Apply the Ingress Resource**
   Use the following command to create the Ingress:

```
kubectl apply -f example-ingress.yaml
```

```
PS C:\Users\Administrator> kubectl apply -f example-ingress.yaml
ingress.networking.k8s.io/example-ingress created
```

3. **Verify the Ingress**
   Check the status of the Ingress:

```
kubectl get ingress
```

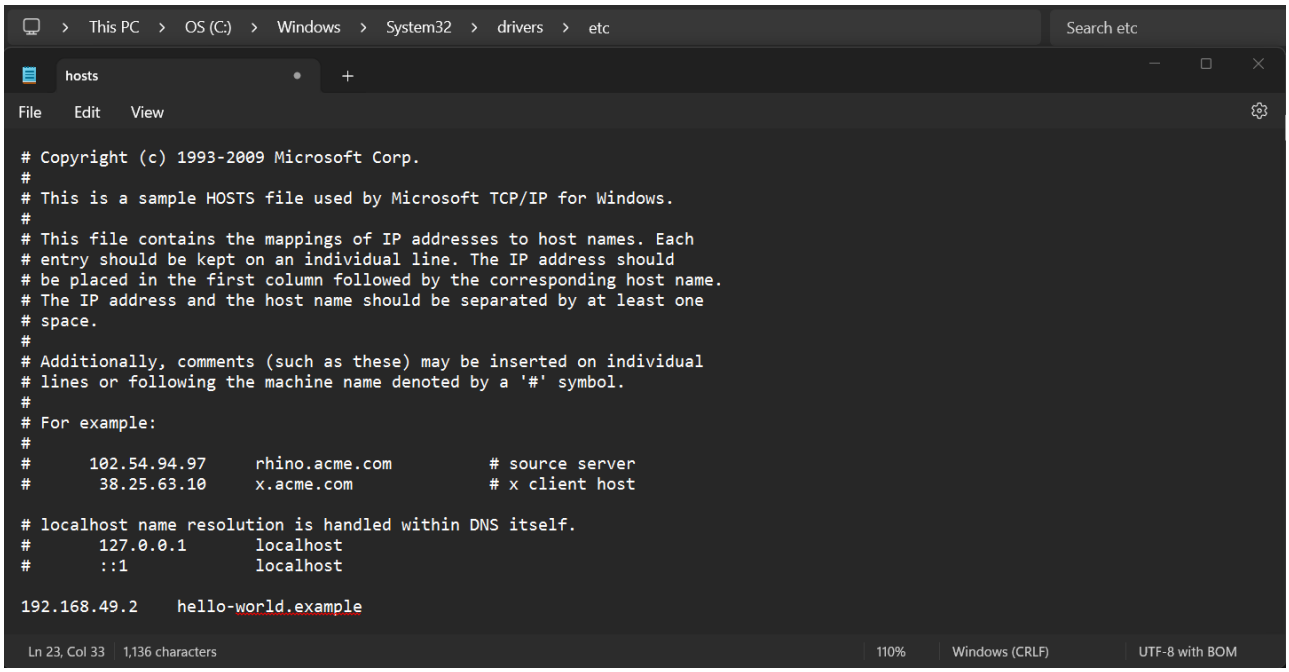
```
PS C:\Users\Administrator> kubectl get ingress
NAME              CLASS   HOSTS                ADDRESS   PORTS   AGE
example-ingress   nginx   hello-world.example             80      12s
```

Note: It may take a minute to assign an address.

Step 5: Test the Ingress

1. **Update the Hosts File**

   Open the `hosts` file with administrative rights (located at `C:\Windows\System32\drivers\etc\hosts`) and add the following line using the IP address returned by `minikube ip`:



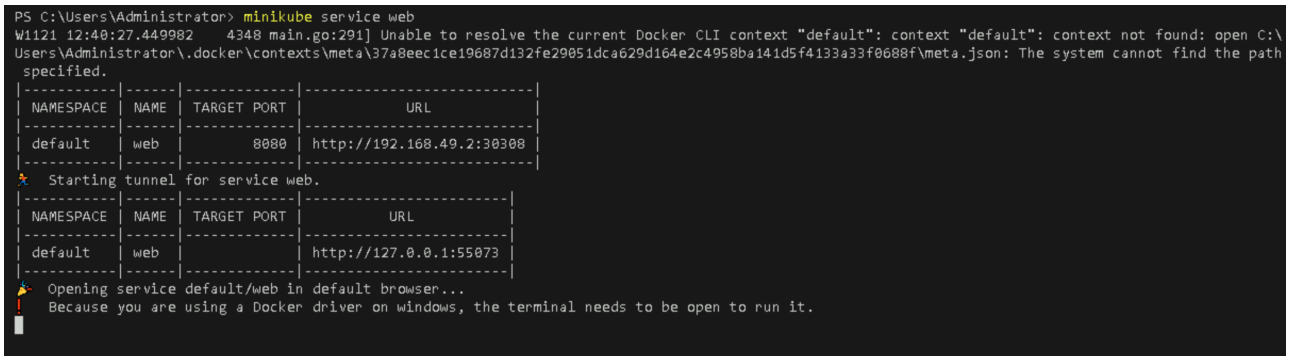

```
<minikube-ip> hello-world.example
```

Replace `<minikube-ip>` with the actual IP address from the previous command.

2. **Test the Ingress**

Visit the Service via NodePort, using the `minikube service` command:

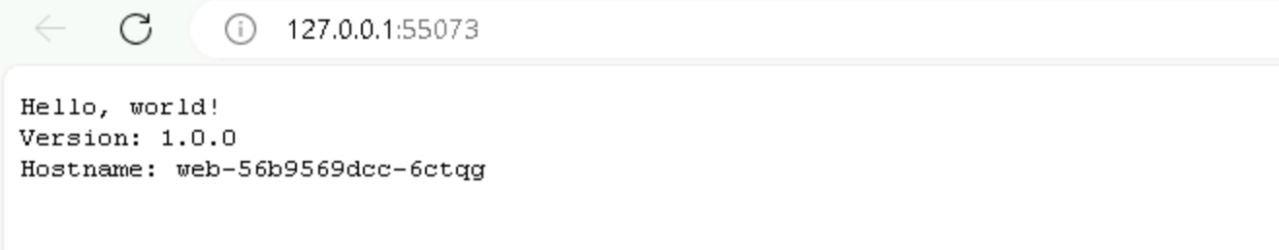
```
minikube service web
```



The above command will open the `sample application` .

The output is similar to:

```
Hello, world!
Version: 1.0.0
Hostname: web-56b9569dcc-48zbn
```



> **Note:** On Linux System we can test ingress using `Minikube IP` address instead of Localhost i.e., `127.0.0.1`

References

- Kubernetes Ingress Documentation
- Minikube Ingress Documentation
