Politechnika Śląska

Micorservice orchestration platforms using Kubernetes

Minikube

Łukasz Zientek

At first I need to state im using VM from my friend since it was not working on my configuration.

To instal minikube we need to write:

```
curl -LO
https://storage.googleapis.com/minikube/releases/latest/minikube_lat
est_amd64.deb
```

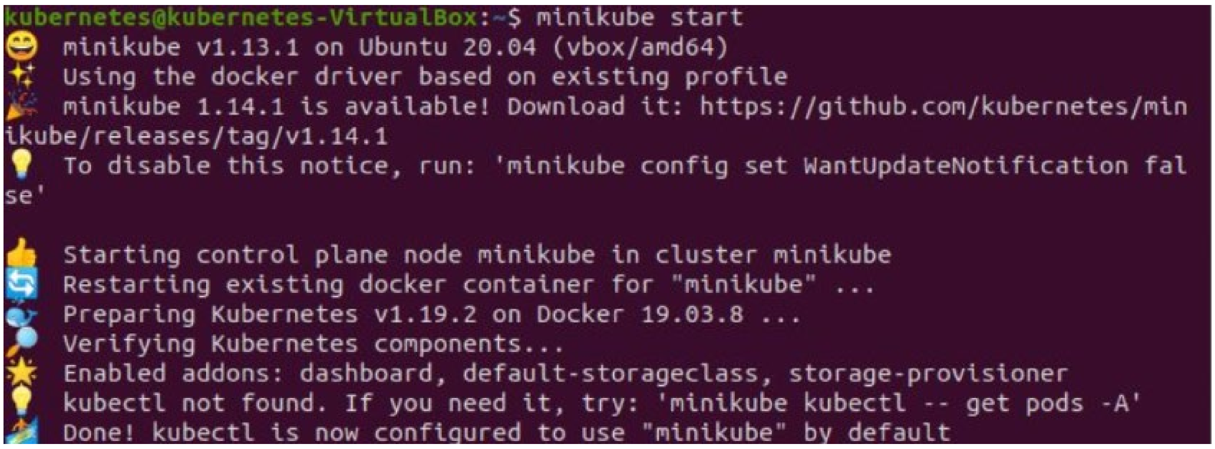
then we use:

```
sudo dpkg -i minikube_latest_amd64.deb
```

To start the minikube we write:

```
Minikube start
```

This will be the result

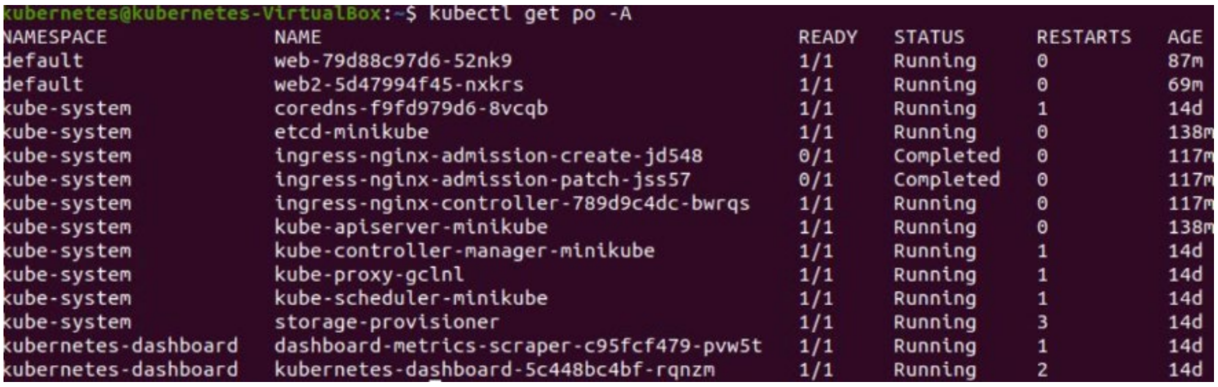


```
kubernetes@kubernetes-VirtualBox:~$ minikube start
    minikube v1.13.1 on Ubuntu 20.04 (vbox/amd64)
    Using the docker driver based on existing profile
    minikube 1.14.1 is available! Download it: https://github.com/kubernetes/min
ikube/releases/tag/v1.14.1
    To disable this notice, run: 'minikube config set WantUpdateNotification fal
se'

    Starting control plane node minikube in cluster minikube
    Restarting existing docker container for "minikube" ...
    Preparing Kubernetes v1.19.2 on Docker 19.03.8 ...
    Verifying Kubernetes components...
    Enabled addons: dashboard, default-storageclass, storage-provisioner
    kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
    Done! kubectl is now configured to use "minikube" by default
```

Next to interact with the cluster we write:

```
kubectl get po -A
```



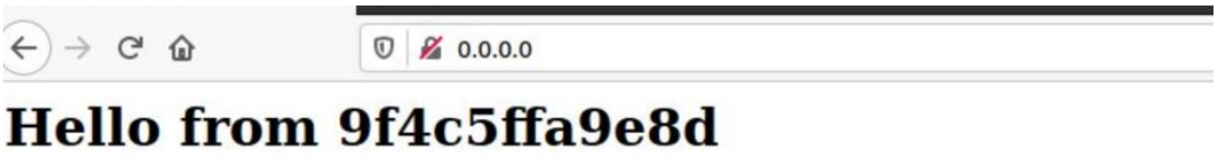
```
kubernetes@kubernetes-VirtualBox:~$ kubectl get po -A
NAMESPACE              NAME                                        READY   STATUS      RESTARTS   AGE
default                web-79d88c97d6-52nk9                        1/1     Running     0          87m
default                web2-5d47994f45-nxkrs                       1/1     Running     0          69m
kube-system            coredns-f9fd979d6-8vcqb                     1/1     Running     1          14d
kube-system            etcd-minikube                               1/1     Running     0          138m
kube-system            ingress-nginx-admission-create-jd548        0/1     Completed   0          117m
kube-system            ingress-nginx-admission-patch-jss57         0/1     Completed   0          117m
kube-system            ingress-nginx-controller-789d9c4dc-bwrqs    1/1     Running     0          117m
kube-system            kube-apiserver-minikube                     1/1     Running     0          138m
kube-system            kube-controller-manager-minikube            1/1     Running     1          14d
kube-system            kube-proxy-gclnl                            1/1     Running     1          14d
kube-system            kube-scheduler-minikube                     1/1     Running     1          14d
kube-system            storage-provisioner                         1/1     Running     3          14d
kubernetes-dashboard   dashboard-metrics-scraper-c95fcf479-pvw5t   1/1     Running     1          14d
kubernetes-dashboard   kubernetes-dashboard-5c448bc4bf-rqnzm       1/1     Running     2          14d
```

Next, we had to install strm/helloworld-http, to do it we had to write

```
Docker pull strm/helloworld-http
```

Then we had to run it:

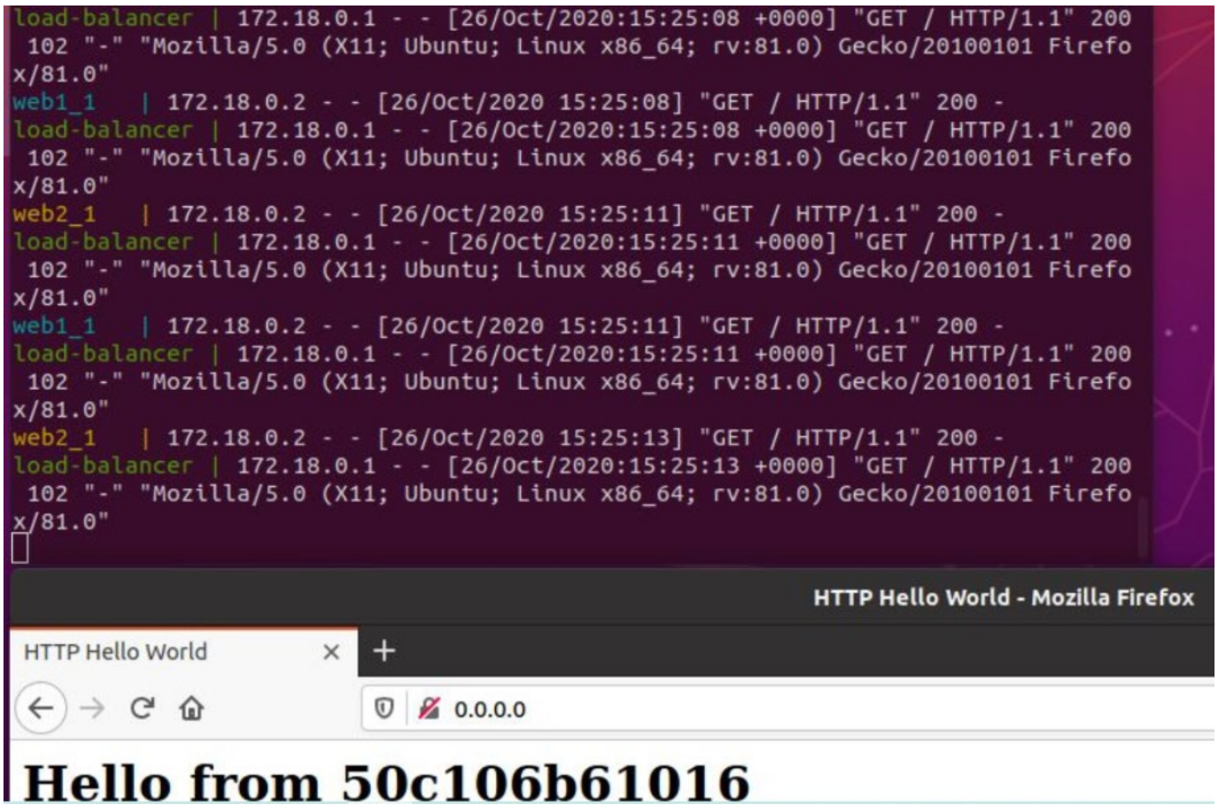
```
Docker run -rm -it -p 80:80 strm/helloworld-http
```



Hello from 9f4c5ffa9e8d

Next we had to create a load balancer, after doing it we run following command:

```
Docker-compose up
```

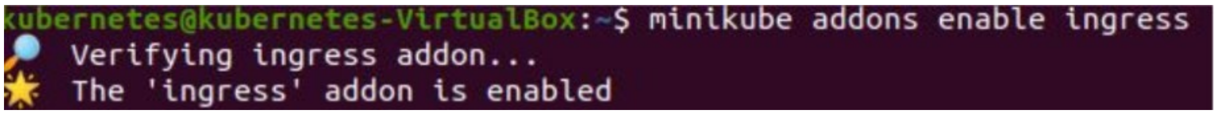


```
load-balancer | 172.18.0.1 - - [26/Oct/2020:15:25:08 +0000] "GET / HTTP/1.1" 200
 102 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefo
x/81.0"
web1_1     | 172.18.0.2 - - [26/Oct/2020 15:25:08] "GET / HTTP/1.1" 200 -
load-balancer | 172.18.0.1 - - [26/Oct/2020:15:25:08 +0000] "GET / HTTP/1.1" 200
 102 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefo
x/81.0"
web2_1     | 172.18.0.2 - - [26/Oct/2020 15:25:11] "GET / HTTP/1.1" 200 -
load-balancer | 172.18.0.1 - - [26/Oct/2020:15:25:11 +0000] "GET / HTTP/1.1" 200
 102 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefo
x/81.0"
web1_1     | 172.18.0.2 - - [26/Oct/2020 15:25:11] "GET / HTTP/1.1" 200 -
load-balancer | 172.18.0.1 - - [26/Oct/2020:15:25:11 +0000] "GET / HTTP/1.1" 200
 102 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefo
x/81.0"
web2_1     | 172.18.0.2 - - [26/Oct/2020 15:25:13] "GET / HTTP/1.1" 200 -
load-balancer | 172.18.0.1 - - [26/Oct/2020:15:25:13 +0000] "GET / HTTP/1.1" 200
 102 "-" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefo
x/81.0"
```

Hello from 50c106b61016

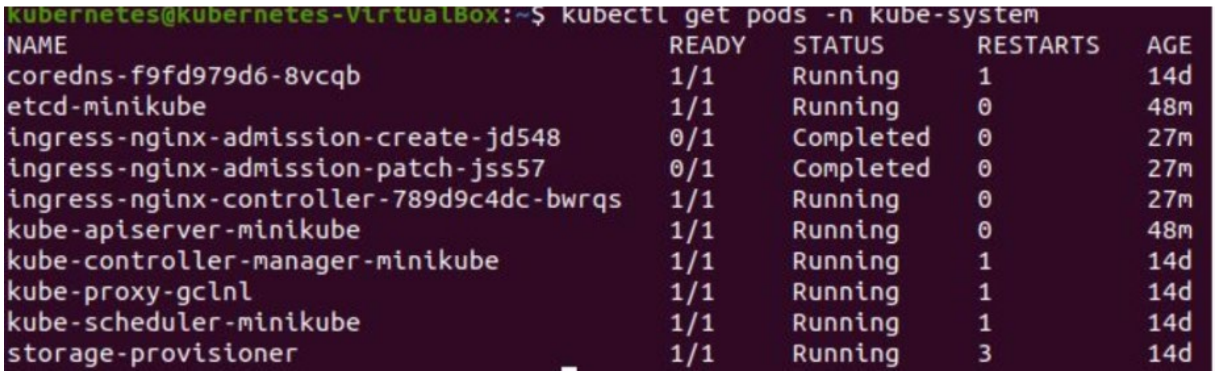
Next we had to set up a Ingress on minikube with the NGINX ingress controller:

```
Minikube addons enable ingress
```



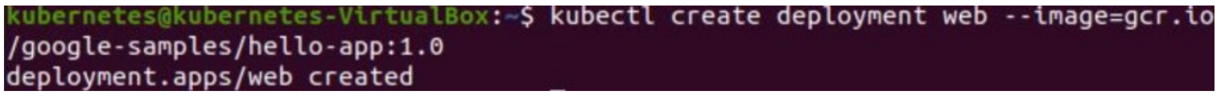
```
kubernetes@kubernetes-VirtualBox:~$ minikube addons enable ingress
🔎  Verifying ingress addon...
🌟  The 'ingress' addon is enabled
```

```
Kubectl get pods –n kube-system
```



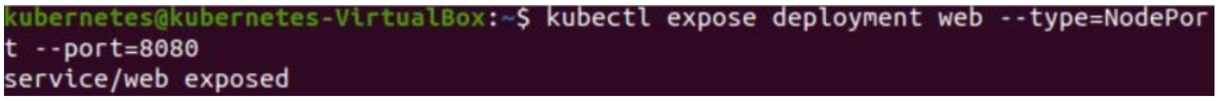
```
kubernetes@kubernetes-VirtualBox:~$ kubectl get pods -n kube-system
NAME                                        READY   STATUS      RESTARTS   AGE
coredns-f9fd979d6-8vcqb                     1/1     Running     1          14d
etcd-minikube                               1/1     Running     0          48m
ingress-nginx-admission-create-jd548        0/1     Completed   0          27m
ingress-nginx-admission-patch-jss57         0/1     Completed   0          27m
ingress-nginx-controller-789d9c4dc-bwrqs    1/1     Running     0          27m
kube-apiserver-minikube                     1/1     Running     0          48m
kube-controller-manager-minikube            1/1     Running     1          14d
kube-proxy-gclnl                            1/1     Running     1          14d
kube-scheduler-minikube                     1/1     Running     1          14d
storage-provisioner                         1/1     Running     3          14d
```

```
Kubectl create deployement web –image=gcr.io/google-samples/hello-
app:1.0
```



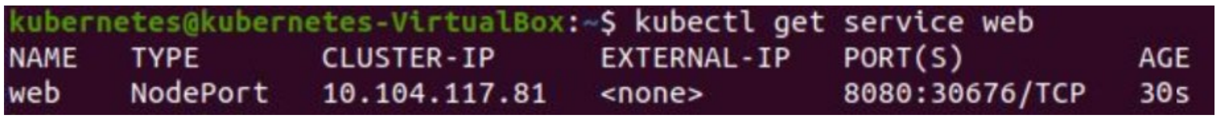
```
kubernetes@kubernetes-VirtualBox:~$ kubectl create deployment web --image=gcr.io
/google-samples/hello-app:1.0
deployment.apps/web created
```

```
Kubectl expose deployment web –type=NodePort –port=8080
```



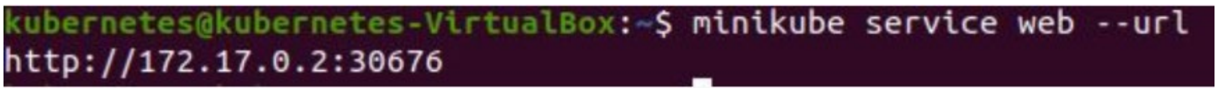
```
kubernetes@kubernetes-VirtualBox:~$ kubectl expose deployment web --type=NodePor
t --port=8080
service/web exposed
```

```
Kubectl get service web
```



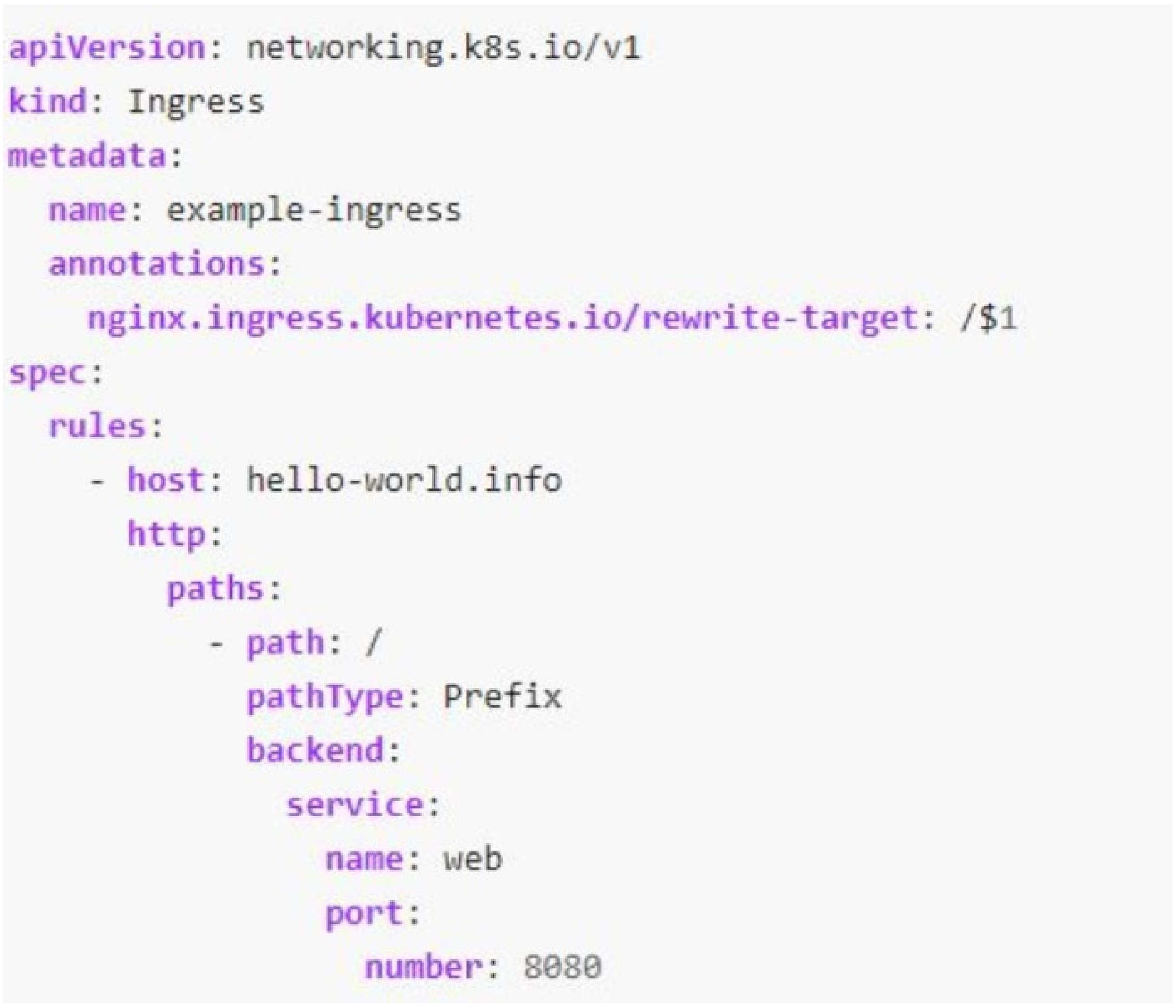
```
kubernetes@kubernetes-VirtualBox:~$ kubectl get service web
NAME   TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
web    NodePort   10.104.117.81   <none>        8080:30676/TCP   30s
```

```
minikube service web –url
```



```
kubernetes@kubernetes-VirtualBox:~$ minikube service web --url
http://172.17.0.2:30676
```

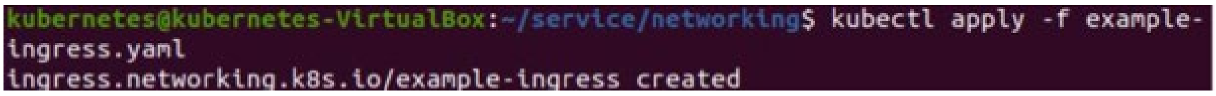
Next we create ingress recource



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
  rules:
    - host: hello-world.info
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 8080
```

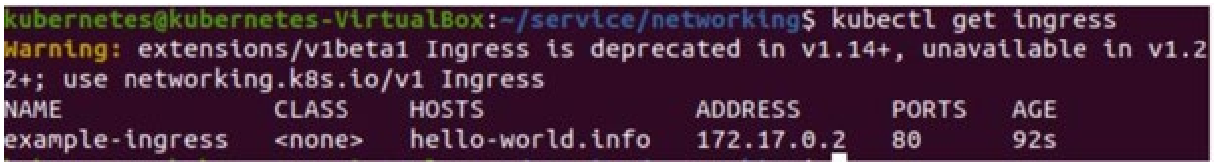
Next we write

```
Kubectl apply –f https://k8s.io/examples/service/networking/example-ingess.yaml
```



```
kubernetes@kubernetes-VirtualBox:~/service/networking$ kubectl apply -f example-
ingress.yaml
ingress.networking.k8s.io/example-ingress created
```

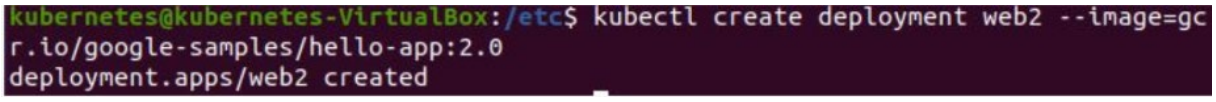
```
Kubectl get ingeress
```



```
kubernetes@kubernetes-VirtualBox:~/service/networking$ kubectl get ingress
Warning: extensions/v1beta1 Ingress is deprecated in v1.14+, unavailable in v1.2
2+; use networking.k8s.io/v1 Ingress
NAME              CLASS    HOSTS              ADDRESS      PORTS   AGE
example-ingress   <none>   hello-world.info   172.17.0.2   80      92s
```

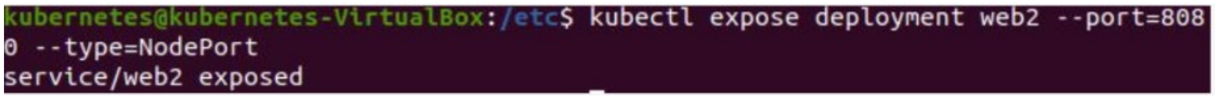
Next we had to modify the file (etc/hosts) and add 172.17.0.2 hello-world.info. After this we create a deployment

```
Kubectl create deployment web2 --image=gcr.io/google-samples/hello-app:2.0
```



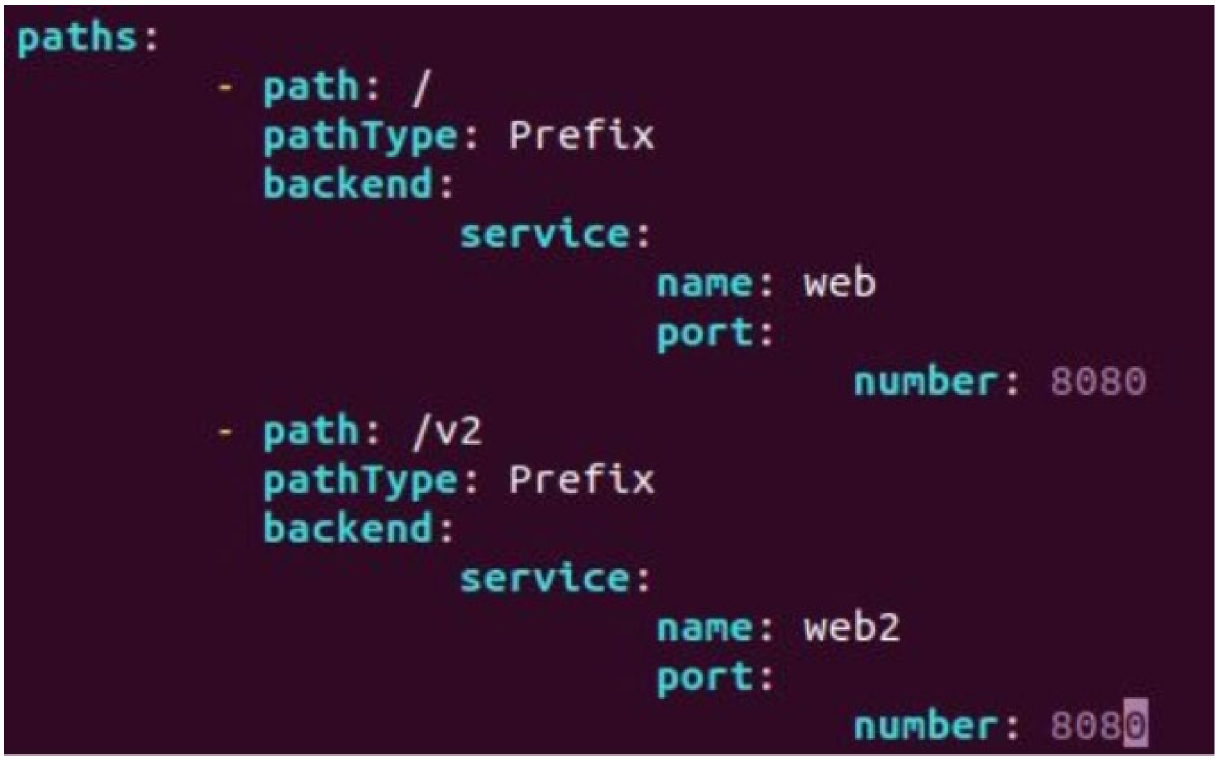
```
kubernetes@kubernetes-VirtualBox:/etc$ kubectl create deployment web2 --image=gc
r.io/google-samples/hello-app:2.0
deployment.apps/web2 created
```

```
Kubectl expose deployment web2 –port=8080 –type=NodePort
```



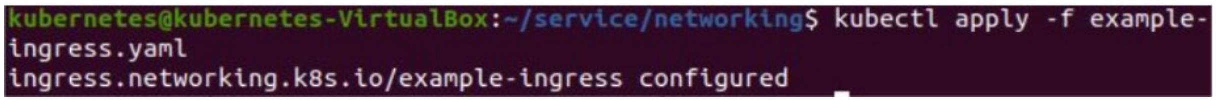
```
kubernetes@kubernetes-VirtualBox:/etc$ kubectl expose deployment web2 --port=808
0 --type=NodePort
service/web2 exposed
```

```
Next we update paths in the example ingress.yaml
```



```
paths:
        - path: /
          pathType: Prefix
          backend:
                service:
                        name: web
                        port:
                                number: 8080
        - path: /v2
          pathType: Prefix
          backend:
                service:
                        name: web2
                        port:
                                number: 8080
```

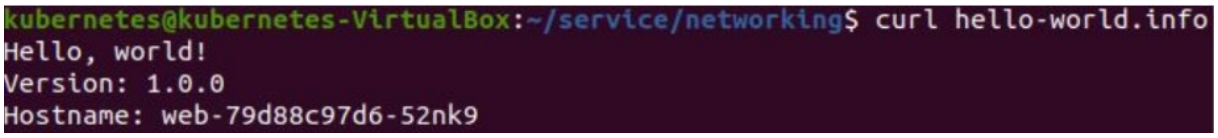
```
Kubeclt apply –f example-ingress.yaml
```



```
kubernetes@kubernetes-VirtualBox:~/service/networking$ kubectl apply -f example-
ingress.yaml
ingress.networking.k8s.io/example-ingress configured
```

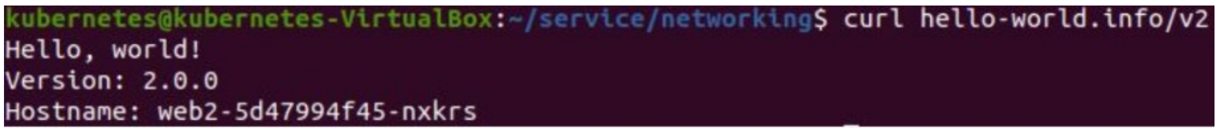
At the very end we test both deployments

```
Curl hello-world.info
```



```
kubernetes@kubernetes-VirtualBox:~/service/networking$ curl hello-world.info
Hello, world!
Version: 1.0.0
Hostname: web-79d88c97d6-52nk9
```

```
Curl hello-world.info/v2
```



```
kubernetes@kubernetes-VirtualBox:~/service/networking$ curl hello-world.info/v2
Hello, world!
Version: 2.0.0
Hostname: web2-5d47994f45-nxkrs
```