# Service Ingress

1. Enable the NGINX Ingress controller:

$ minikube addons enable ingress  
💡 After the addon is enabled, please run "minikube tunnel" and your ingress resources would be available at "127.0.0.1"  
 ▪ Using image k8s.gcr.io/ingress-nginx/controller:v0.44.0  
 ▪ Using image docker.io/jettech/kube-webhook-certgen:v1.5.1  
 ▪ Using image docker.io/jettech/kube-webhook-certgen:v1.5.1  
🔎 Verifying ingress addon...  
🌟 The 'ingress' addon is enabled

1. Verify that the NGINX Ingress controller is running:

$ kubectl get pods -n ingress-nginx  
NAME READY STATUS RESTARTS AGE  
ingress-nginx-admission-create-t6vhl 0/1 Completed 0 16s  
ingress-nginx-admission-patch-vq8zk 0/1 Completed 0 15s  
ingress-nginx-controller-5d88495688-qw6cj 1/1 Running 0 16s

1. Create Deployments:

$ kubectl create deployment web1 --image=gcr.io/google-samples/hello-app:1.0  
deployment.apps/web1 created  
$ kubectl create deployment web2 --image=gcr.io/google-samples/hello-app:2.0  
deployment.apps/web2 created

1. Expose the Deployments:

$ kubectl expose deployment web1 --port=8080  
service/web1 exposed  
$ kubectl expose deployment web2 --port=8080  
service/web2 exposed

1. Verify the Service is created and is available on a node port:

$ kubectl get service web1 web2  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
web1 NodePort 10.101.151.71 <none> 8080:30577/TCP 32s  
web2 NodePort 10.109.3.191 <none> 8080:30370/TCP 8s

1. Visit the service via NodePort:

$ minikube service web1 --url  
http://127.0.0.1:57855  
  
$ curl http://127.0.0.1:57855  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-dpdwb

Do the same for service web2.

1. Run workaround

$ kubectl delete -A ValidatingWebhookConfiguration ingress-nginx-admission  
validatingwebhookconfiguration.admissionregistration.k8s.io "ingress-nginx-admission" deleted

to prevent:

$ kubectl apply -f services/service-ingress.yaml  
Error from server (InternalError): error when creating "services/service-ingress.yaml": Internal error occurred: \  
failed calling webhook "validate.nginx.ingress.kubernetes.io": an error on the server ("") has prevented the request from succeeding

1. Create ingress controller:

$ kubectl apply -f services/service-ingress.yaml  
ingress.networking.k8s.io/myingress created  
  
$ kubectl get ingress  
NAME CLASS HOSTS ADDRESS PORTS AGE  
myingress <none> hello-world.info 192.168.49.2 80 45s

1. Create a tunnel into minikube cluster *in separate terminal* (or run it in background):

$ minikube tunnel

1. Curl the server:

$ minikube ssh  
docker@minikube:~$ curl -H 'Host: hello-world.info' http://192.168.49.2/  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-x255w  
docker@minikube:~$ curl -H 'Host: hello-world.info' http://192.168.49.2/v2  
Hello, world!  
Version: 2.0.0  
Hostname: web2-5d47994f45-zx2f4  
docker@minikube:~$ curl -H 'Host: hello-world.error' http://192.168.49.2/v2  
<html>  
<head><title>404 Not Found</title></head>  
<body>  
<center><h1>404 Not Found</h1></center>  
<hr><center>nginx</center>  
</body>  
</html>  
docker@minikube:~$ exit

1. Stop tunneling by pressing Ctrl+C
2. Clean up:

$ kubectl delete ingress/myingress svc/web1 svc/web2 deploy/web1 deploy/web2  
ingress.networking.k8s.io "myingress" deleted  
service "web1" deleted  
service "web2" deleted  
deployment.apps "web1" deleted  
deployment.apps "web2" deleted

## Solution for minikube

🧨

1. Enable the NGINX Ingress controller:

$ minikube addons enable ingress  
💡 After the addon is enabled, please run "minikube tunnel" and your ingress resources would be available at "127.0.0.1"  
 ▪ Using image k8s.gcr.io/ingress-nginx/controller:v0.44.0  
 ▪ Using image docker.io/jettech/kube-webhook-certgen:v1.5.1  
 ▪ Using image docker.io/jettech/kube-webhook-certgen:v1.5.1  
🔎 Verifying ingress addon...  
🌟 The 'ingress' addon is enabled

1. Verify that the NGINX Ingress controller is running:

$ kubectl get pods -n ingress-nginx  
NAME READY STATUS RESTARTS AGE  
ingress-nginx-admission-create-zzzk5 0/1 Completed 0 68s  
ingress-nginx-admission-patch-7nrjr 0/1 Completed 0 68s  
ingress-nginx-controller-5d88495688-k7xrw 1/1 Running 0 68s

1. Create Deployments:

$ kubectl create deployment web1 --image=gcr.io/google-samples/hello-app:1.0  
deployment.apps/web1 created  
  
$ kubectl create deployment web2 --image=gcr.io/google-samples/hello-app:2.0  
deployment.apps/web2 created

1. Expose the Deployments:

$ kubectl expose deployment web1 --port=8080  
service/web1 exposed  
$ kubectl expose deployment web2 --port=8080  
service/web2 exposed

1. Verify the Service is created and is available on a node port:

$ kubectl get service web1 web2  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
web1 ClusterIP 10.110.50.25 <none> 8080/TCP 10s  
web2 ClusterIP 10.106.106.3 <none> 8080/TCP 7s

1. Visit the services via NodePort:

$ minikube service web1 --url  
http://127.0.0.1:57418  
  
$ curl http://127.0.0.1:57418  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-xgr25  
  
$ minikube service web2 --url  
http://127.0.0.1:60532  
  
$ curl http://127.0.0.1:60532  
Hello, world!  
Version: 2.0.0  
Hostname: web2-5d47994f45-j6twv

1. Run workaround

$ kubectl delete -A ValidatingWebhookConfiguration ingress-nginx-admission  
validatingwebhookconfiguration.admissionregistration.k8s.io "ingress-nginx-admission" deleted

1. Create ingress controller:

$ kubectl apply -f services/service-ingress.yaml  
ingress.networking.k8s.io/myingress created  
  
$ kubectl get ingress  
NAME CLASS HOSTS ADDRESS PORTS AGE  
myingress <none> hello-world.info 80 42s

It may take a couple of minutes to get IP address:

$ kubectl get ingress  
NAME CLASS HOSTS ADDRESS PORTS AGE  
myingress <none> hello-world.info 192.168.49.2 80 3m40s

1. Create a tunnel into minikube cluster *in separate terminal* (or run it in background):

$ minikube service list  
|---------------|------------------------------------|--------------|---------------------------|  
| NAMESPACE | NAME | TARGET PORT | URL |  
|---------------|------------------------------------|--------------|---------------------------|  
| default | kubernetes | No node port |  
| default | web1 | No node port |  
| default | web2 | No node port |  
| ingress-nginx | ingress-nginx-controller | http/80 | http://192.168.49.2:30386 |  
| | | https/443 | http://192.168.49.2:32505 |  
| ingress-nginx | ingress-nginx-controller-admission | No node port |  
| kube-system | kube-dns | No node port |  
|---------------|------------------------------------|--------------|---------------------------|  
  
$ minikube service ingress-nginx-controller -n ingress-nginx --url  
🏃 Starting tunnel for service ingress-nginx-controller.  
|---------------|--------------------------|-------------|------------------------|  
| NAMESPACE | NAME | TARGET PORT | URL |  
|---------------|--------------------------|-------------|------------------------|  
| ingress-nginx | ingress-nginx-controller | | http://127.0.0.1:56960 |  
| | | | http://127.0.0.1:56961 |  
|---------------|--------------------------|-------------|------------------------|  
http://127.0.0.1:56960  
http://127.0.0.1:56961  
❗ Because you are using a Docker driver on windows, the terminal needs to be open to run it.

1. Curl the server:

$ curl -H 'Host: hello-world.info' http://127.0.0.1:56960/  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-xgr25  
  
$ curl -H 'Host: hello-world.info' http://127.0.0.1:56960/v2  
Hello, world!  
Version: 2.0.0  
Hostname: web2-5d47994f45-j6twv  
  
$ curl -H 'Host: hello-world.error' http://127.0.0.1:56960/v2  
<html>  
<head><title>404 Not Found</title></head>  
<body>  
<center><h1>404 Not Found</h1></center>  
<hr><center>nginx</center>  
</body>  
</html>

1. Stop tunneling by pressing Ctrl+C
2. Clean up:

$ kubectl delete ingress/myingress svc/web1 svc/web2 deploy/web1 deploy/web2  
ingress.networking.k8s.io "myingress" deleted  
service "web1" deleted  
service "web2" deleted  
deployment.apps "web1" deleted  
deployment.apps "web2" deleted

## Solution for AWS

Change context to cluster in AWS:

$ kubectl config use-context aws  
Switched to context "aws".

1. Create Deployments:

$ kubectl create deployment web1 --image=gcr.io/google-samples/hello-app:1.0  
deployment.apps/web1 created  
  
$ kubectl create deployment web2 --image=gcr.io/google-samples/hello-app:2.0  
deployment.apps/web2 created

1. Expose the Deployments:

$ kubectl expose deployment web1 --port=8080  
service/web1 exposed  
  
$ kubectl expose deployment web2 --port=8080  
service/web2 exposed

1. Verify the Service is created and is available on a node port:

$ kubectl get service web1 web2  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
web1 ClusterIP 172.20.250.133 <none> 8080/TCP 19s  
web2 ClusterIP 172.20.242.35 <none> 8080/TCP 11s

1. Visit the service and curl the services:

kubectl run busybox --image yauritux/busybox-curl --rm -it  
If you don't see a command prompt, try pressing enter.  
  
/home # curl web1:8080  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-xdj6j  
  
/home # curl web2:8080  
Hello, world!  
Version: 2.0.0  
Hostname: web2-5d47994f45-vc2np  
  
/home # exit  
Session ended, resume using 'kubectl attach busybox -c busybox -i -t' command when the pod is running  
pod "busybox" deleted

1. Create ingress controller:

https://aws.amazon.com/ru/blogs/opensource/kubernetes-ingress-aws-alb-ingress-controller/

$ kubectl apply -f services/service-ingress.yaml  
ingress.networking.k8s.io/myingress created  
  
$ kubectl get ingress  
NAME CLASS HOSTS ADDRESS PORTS AGE  
myingress <none> hello-world.info 192.168.49.2 80 45s

1. Curl the server:

$ minikube ssh  
docker@minikube:~$ curl -H 'Host: hello-world.info' http://192.168.49.2/  
Hello, world!  
Version: 1.0.0  
Hostname: web1-7856799799-x255w  
docker@minikube:~$ curl -H 'Host: hello-world.info' http://192.168.49.2/v2  
Hello, world!  
Version: 2.0.0  
Hostname: web2-5d47994f45-zx2f4  
docker@minikube:~$ curl -H 'Host: hello-world.error' http://192.168.49.2/v2  
<html>  
<head><title>404 Not Found</title></head>  
<body>  
<center><h1>404 Not Found</h1></center>  
<hr><center>nginx</center>  
</body>  
</html>  
docker@minikube:~$ exit

1. Stop tunneling by pressing Ctrl+C
2. Clean up:

$ kubectl delete ingress/myingress svc/web1 svc/web2 deploy/web1 deploy/web2  
ingress.networking.k8s.io "myingress" deleted  
service "web1" deleted  
service "web2" deleted  
deployment.apps "web1" deleted  
deployment.apps "web2" deleted