#KubernetesIn30Days challenge:-

#Day11 :-

complete Info about 'Ingress'

Ingress :-

→ It is a API object that manages external access to the services in the cluster typically HTTP and HTTPS.

It may provide

① Load Balancing

② SSL Termination

③ name based virtual Hosting

Q. When we have service to Access the cluster/App. What is the Need of this Ingress?

sol:- Question is valid because when we have Load Balancer service is there for external Access. Why we need this Ingress? so Here are the few reason why we need Ingress

① services lacking the commercial Enterprise level Load Balancers organizations such as F5 LB, NGINX... . These convential LB are providing a lot of commercial features such as host based, path based, ratio based, WAF etc..

(2) Lets assume we have 50 services in an Application so for each service of type LB should have LB. this means 50 services needs 50 Load Balancers. practically this will make more cost to client from CSPs.

Intial version of K8 doesn't have the concept of "Ingress". the importance of this Ingress came after people adopted K8 for production.

Openshift (based kubernetes) is developed by Redhat (IBM) is the first one to interduce Ingress kind of feature i.e. Routes.

Later then K8 admitted the importance of the usage of commercial LB infront of cluster.

At that point K8, Interduced Ingress to cluster. which is a resource of kubernetes cluster we can create.

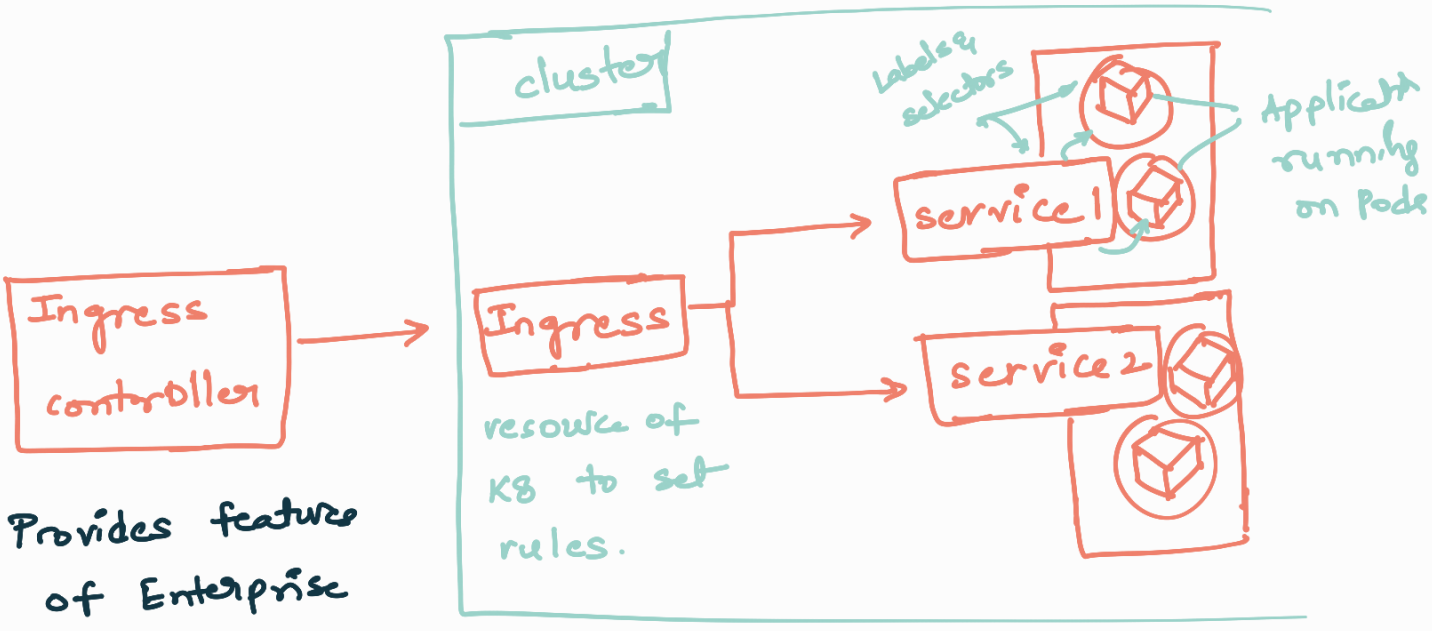
Ingress controller - It is that where we can manage all the routing policies like a commercial LB. These are developed by the respective organisation to be used for K8 cluster.

NGINX Ingress controller → Developed by NGINX

F5 Ingress controller → Developed by F5.

Similarly remaining will also develop their Ingress controllers to be a part of K8 cluster.

For Better Understanding observe the flow.



Path based Routing Example :-

/login → service1

/Dashboard → service2

/checkout → service3

/feed → service4.

Now it's time to do Handson this Ingress.

Iam using minikube -cluster. The process might be depends on type of platform yow using.

In Kubernetes Documentation, Browse to "setup Ingress on Minikube with NGINX Ingress controller"

① Enable Ingress-controller (NGINX here)

#minikube addon enable Ingress

verify that the NGINX Ingress controller is running using below cmd:

#Kubectl get pods -n Ingress-nginx

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ minikube addons enable ingress
* ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
  - Using image registry.k8s.io/ingress-nginx/controller:v1.9.4
* Verifying ingress addon...
* The 'ingress' addon is enabled

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl get pods -n ingress-nginx
NAME                                        READY   STATUS      RESTARTS   AGE
ingress-nginx-admission-create-p5pbj        0/1     Completed   0          23m
ingress-nginx-admission-patch-5hnt8         0/1     Completed   1          23m
ingress-nginx-controller-7c6974c4d8-41vfv   1/1     Running     0          23m
```

② Deploy the webapp1-sample using below cmd:

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl create deployment web --image=gcr.io/google-samples/hello-app:1.0
deployment.apps/web created

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl expose deployment web --type=NodePort --port=8080
service/web exposed

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ minikube service web --url
http://192.168.59.109:32403
```

③ create Ingress resource using below file except the /v2 → lines (which will be added after creating web2 svc & deploy)

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ cat example_ingress.yml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
  rules:
    - host: test.domain
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 8080
          - path: /v2
            pathType: Prefix
            backend:
              service:
                name: web2
                port:
                  number: 8080
```

④ Now verify the application Endpoints using below cmd: (Version 1.0.0)

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ curl --resolve "test.domain:80:$( minikube ip )" -i http://test.domain
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    60  100    60    0     0  15511      0 --:--:-- --:--:-- --:--:-- 20000HTTP/1.1 200 OK
Date: Mon, 18 Dec 2023 11:37:07 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 60
Connection: keep-alive

Hello, world!
Version: 1.0.0
Hostname: web-57f46db77f-8db5h
```

⑤ similarly create deployment & service for web2 as follows:- (Here we will add /v2 lines in ingress file)

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl create deployment web2 --image=gcr.io/google-samples/hello-app:2.0
deployment.apps/web2 created

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl get deploy
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
web     1/1     1            1           111m
web2    1/1     1            1           6s

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl expose deployment web2 --port=8080 --type=NodePort
service/web2 exposed

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ vi example_ingress.yml

betha@Rajasekhar_PC MINGW64 /d/test/k8
$ kubectl apply -f example_ingress.yml
ingress.networking.k8s.io/example-ingress configured
```

⑥ Now, verify that web2 app is accessible at /v2 Endpoint Ending. (version 2.0.0)

```
betha@Rajasekhar_PC MINGW64 /d/test/k8
$ curl --resolve "test.domain:80:$( minikube ip )" -i http://test.domain/v2
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    61  100    61    0     0  13625      0 --:--:-- --:--:-- --:--:-- 15250HTTP/1.1 200 OK
Date: Mon, 18 Dec 2023 12:02:10 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 61
Connection: keep-alive

Hello, world!
Version: 2.0.0
Hostname: web2-866dc4bcc8-dlsvs
```