Experiment 16: Istio on k3d

In this experiment, we will deploy Istio and access on K3d.

Create a cluster without traefik, since there are known issues in k3d with istio and traefik

$ k3d cluster create istio-demo –api-port 6660 -agents 2 –server-arg –no-deploy –server-arg traefik

Generate config
$ export KUBECONFIG=$(k3d get-kubeconfig)

# Checklus
$ kubectl get pod,svc -A
NAMESPACE NAME READY STATUS RESTARTS AGE
kube-system pod/local-path-provisioner-58fb86bdfd-h6npn 1/1 Running 0 13m
kube-system pod/coredns-57d8bbb86-zkjkq 1/1 Running 0 13m

NAMESPACE NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
default service/kubernetes ClusterIP 10.43.0.1 <none> 443/TCP 13m
kube-system service/kube-dns ClusterIP 10.43.0.10 <none> 53/UDP,53/TCP,9153/TCP 13m

Now I'm ready for installing Istio on it.

Install Istio

We will use a recent release of 1.6 for Istio to utilize a widely used release version, rather than the newer 1.7 or very new 1.8 versions

Download Istio from here:

https://github.com/istio/istio/releases/tag/1.6.14

For MacOS:

https://github.com/istio/istio/releases/download/1.6.14/istio-1.6.14-osx.tar.gz is the target so we'll use

$ curl -L https://istio.io/downloadIstio | ISTIO_VERSION=1.6.14 sh -

For Windows:

Download and unzip

https://github.com/istio/istio/releases/download/1.6.14/istio-1.6.14-win.zip

Or if you have Unix tools on windows

```
$ curl -L https://istio.io/downloadIstio | ISTIO_VERSION=1.6.8
TARGET_ARCH=x86_64 s
```

For additional information on Istio setup we could reference:

https://istio.io/docs/setup/install/helm/

We already installed Helm and we'll use the template for Istio

Create a namespace istio-system for Istio components:
$ kubectl create namespace istio-system

Install the Istio base chart which contains cluster-wide resources used by the Istio control plane:

$ helm install -n istio-system istio-base manifests/charts/base

Install the Istio discovery chart which deploys the istiod service:
$ helm install --namespace istio-system istiod manifests/charts/istio-control/istio-discovery --set global.hub="docker.io/istio" --set global.tag="1.6.14"

Install the Istio ingress chart which contains the ingress gateway components:

$ helm install --namespace istio-system istio-ingress manifests/charts/gateways/istio-ingress --set global.hub="docker.io/istio" --set global.tag="1.6.14"

Install the Istio egress chart which contains the egress gateway components:

$ helm install --namespace istio-system istio-egress manifests/charts/gateways/istio-egress --set global.hub="docker.io/istio" --set global.tag="1.6.14"

Optimistically there will be no errors. Not let's check the deployment.

$ kubectl get svc,pod -n istio-system

```
NAME                         TYPE          CLUSTER-IP      EXTERNAL-IP    PORT(S)
AGE
service/istio-galley         ClusterIP     10.43.10.191    <none>
443/TCP,15014/TCP,9901/TCP
2m21s
service/istio-policy         ClusterIP     10.43.86.131    <none>
9091/TCP,15004/TCP,15014/TCP
2m21s
service/istio-telemetry      ClusterIP     10.43.11.107    <none>
9091/TCP,15004/TCP,15014/TCP,42422/TCP
2m21s
service/istio-pilot          ClusterIP     10.43.126.19    <none>
15010/TCP,15011/TCP,8080/TCP,15014/TCP
2m21s
service/prometheus           ClusterIP     10.43.41.148    <none>        9090/TCP
2m21s
service/istio-citadel        ClusterIP     10.43.91.217    <none>
8060/TCP,15014/TCP
2m21s
service/istio-sidecar-injector   ClusterIP     10.43.117.133   <none>
443/TCP,15014/TCP
2m21s
service/istio-ingressgateway     LoadBalancer   10.43.69.0      192.168.96.2
15020:30845/TCP,80:31380/TCP,443:31390/TCP,31400:31400/TCP,15029:31842/TCP
,15030:32247/TCP,15031:32685/TCP,15032:31093/TCP,15443:30499/TCP   2m21s

NAME                                  READY   STATUS      RESTARTS   AGE
pod/istio-init-crd-10-1.3.5-28hj7     0/1     Completed   0          5m40s
pod/istio-init-crd-11-1.3.5-vmwmw     0/1     Completed   0          5m40s
pod/istio-init-crd-12-1.3.5-84q77     0/1     Completed   0          5m40s
pod/istio-security-post-install-1.3.5-jb66j   0/1     Completed   0          2m21s
pod/svclb-istio-ingressgateway-ww22d          9/9     Running     0          2m21s
pod/istio-citadel-5c67db5cb-hmhvb     1/1     Running     0          2m20s
pod/prometheus-6f74d6f76d-tpjpc       1/1     Running     0          2m20s
pod/istio-policy-66d87c756b-hf4wx     2/2     Running     3          2m21s
pod/istio-galley-56b9fb859d-7jmsq     1/1     Running     0          2m21s
pod/istio-sidecar-injector-5d65cfcd79-lhh6k   1/1     Running     0          2m20s
pod/istio-pilot-64478c6886-9xm7b      2/2     Running     0          2m20s
pod/istio-telemetry-5d4c4bfbbf-g4ccz          2/2     Running     4          2m20s
pod/istio-ingressgateway-7b766b6685-5vwg5     1/1     Running     0          2m21s
```

Next, we will run a sample application on our Istio configuration on k3d.

Deploy bookinfo sample application

To verify, we will deploy the bookinfo sample application included in Istio. We can reference additional detail at

https://istio.io/latest/docs/examples/bookinfo/

Since BookInfo is included in Istio, we'll have that with our installation

Enable automatic sidecar injection
$ kubectl label namespace default istio-injection=enabled

# Deploy apps
$ kubectl apply -f samples/bookinfo/platform/kube/bookinfo.yaml

# Wait for the deployment finished for example using watch
$ kubectl get pods -w
```
NAME                          READY   STATUS            RESTARTS   AGE
details-v1-78d78fbddf-5db8b       0/2     PodInitializing   0          37s
reviews-v1-7bb8ffd9b6-rdgjc       0/2     PodInitializing   0          37s
ratings-v1-6c9dbf6b45-p7567       0/2     PodInitializing   0          36s
productpage-v1-596598f447-nj6wx   0/2     PodInitializing   0          36s
reviews-v3-68964bc4c8-qrhc4       0/2     PodInitializing   0          37s
reviews-v2-d7d75fff8-65f4q        0/2     PodInitializing   0          37s
```

# Create ingress gateway for bookinfo
$ kubectl apply -f samples/bookinfo/networking/bookinfo-gateway.yaml

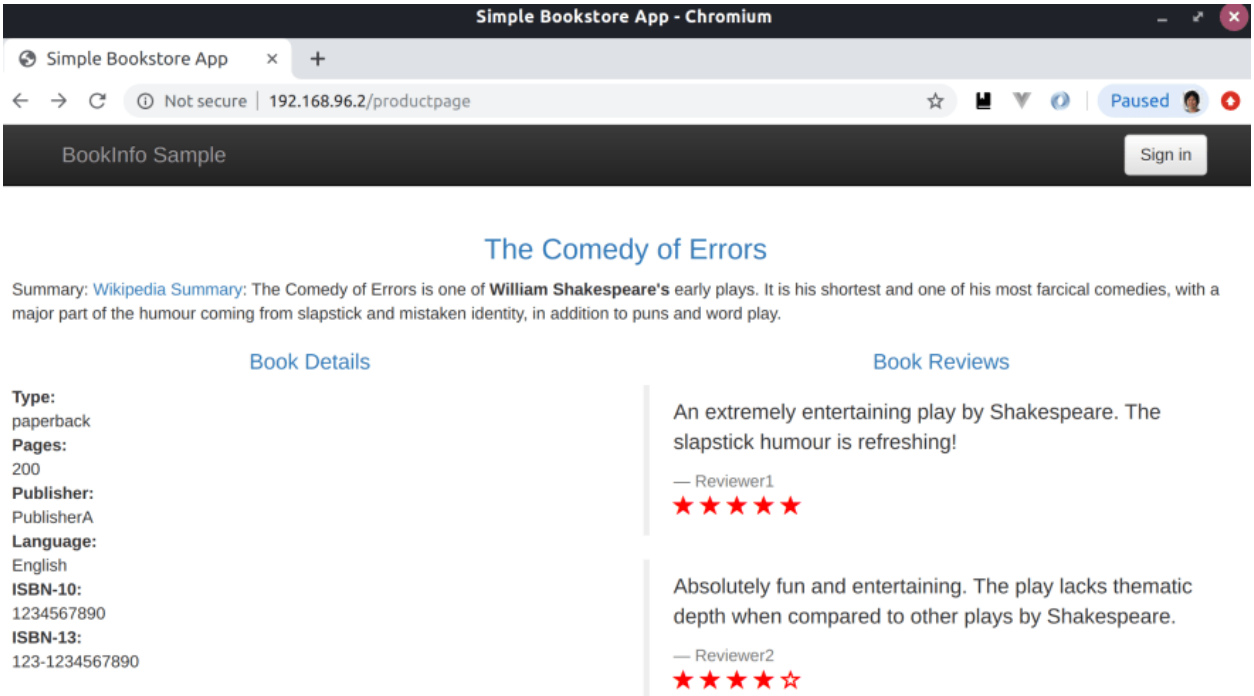
After that, we confirm the external IP of LoadBalancer service:

$ kubectl get svc -n istio-system istio-ingressgateway -o
jsonpath='{.status.loadBalancer.ingress[0].ip}'
192.168.96.2

and opened the following URL with the IP:

http://{The IP Address}/productpage

We should see the following



The memory usage of the container with bookinfo was around 2GiB:

$ docker stats --no-stream
CONTAINER ID NAME CPU % MEM USAGE / LIMIT MEM % NET I/O BLOCK I/O PIDS
598bd6d07c85 k3d-k3s-default-server 52.24% 1.909GiB / 15.4GiB 12.40% 819MB / 21.7MB 1.41MB / 818MB 899