Experiment: 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 andtraefik

```
$ k3d cluster create istio-demo --servers 1 --agents 3 --port
9080:80@loadbalancer --port 9443:443@loadbalancer --api-port 6443 --
k3s-server-arg '--no-deploy=traefik'
```

Let's explain the above a little bit…

- --servers 1 simply means there will be one server node for the control plane.
- --agents 3 simply means we will have 3 nodes to run containers on
- --port 9080:80@loadbalancer simply means that the load balancer (in docker, which is exposed), will forward requests to port 9080 to 80 in the k8 cluster, you can check this out after creation by running docker ps
- --port 944:443@loadbalancer same as above, just for HTTPS (later)
- --api-port 6443 the k8 API port will be port 6443 instead of randomly generated
- --k3s-server-arg '--no-deploy=traefik' simply means k3d will not deploy the Traefik v1 ingress controller

Generate config

$ export KUBECONFIG=$(k3d kubeconfig get istio-demo)

Check our pods and services

$ 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  013m

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
```

Look at our cluster with Docker

$ docker ps --format 'table {{.ID}}\t{{.Image}}\t{{.Names}}\t{{.Ports}}'

Should look similar to the following

```
CONTAINER ID NAMES PORTS
33db5801a3fc k3d-istio-demo-serverlb 0.0.0.0:6443->6443/tcp,
0.0.0.0:9080->80/tcp, 0.0.0.0:9443->443/tcp
```

```
34407a2fbbb2 k3d-istio-demo-agent-2
14f720b707e8 k3d-istio-demo-agent-1
d4beb7b29e6b k3d-istio-demo-agent-0
742d2eb6999f k3d-my-multinode-cluster-server-0
```

Now we're ready for installing Istio on it.

Install Istio
We will use a recent release of 1.10 for Istio to utilize the most current version

We'll download Istio from the releases site: https://github.com/istio/istio/releases

For Windows and MacOS:

To download the latest version, run the following. This will download the archive and extract it for you.

$ curl -L https://istio.io/downloadIstio | sh -
once completed, move the folder to the desired location and navigate to it as follows.

Note: *Make sure to check which version you have*)

```
cd istio-1.10.3
```
then run the following to add it to your path…

```
$ export PATH=$PWD/bin:$PATH
```

Test it by running

```
istioctl version
```

We should see something like the following

```
no running Istio pods in "istio-system"
1.10.3
```

and that's it, we can now actually start deploying Istio

We'll use the default profile. A few useful commands to try out profiles are:

```
$ istioctl profile list # Will list available profiles
$ istioctl profile dump default # Will dump the default profile config
```

Some notes about the default profile…

- Ingress Gateway is **enabled**
- Egress Gateway is **disabled**
- Istiod is **enabled**

Let's get started, by installing Istio

$ **istioctl install --set profile=default**

you will see the following…

This will install the Istio default profile with ["Istio core" "Istiod" "Ingress gateways"] components into the cluster. Proceed? (y/N) y
✔ Istio core installed
✔ Istiod installed
✔ Ingress gateways installed
✔ Installation complete

You have successfully installed Istio.

To enable the automatic injection of Envoy sidecar proxies, run the following:
Note: Otherwise you will need to do this manually when you deploy your applications.

$ **kubectl label namespace default istio-injection=enabled**

Optimistically there will be no errors. Now let's check the deployment.

$ kubectl get svc,pod -n istio-system

```
NAME                TYPE          CLUSTER-IP     EXTERNAL-IP    PORT(S)
AGE
service/istio-      ClusterIP     10.43.10.191   <none>
galley
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

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
```

Note: Ensure the pods complete container creation or of course the application will not be visible

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}'

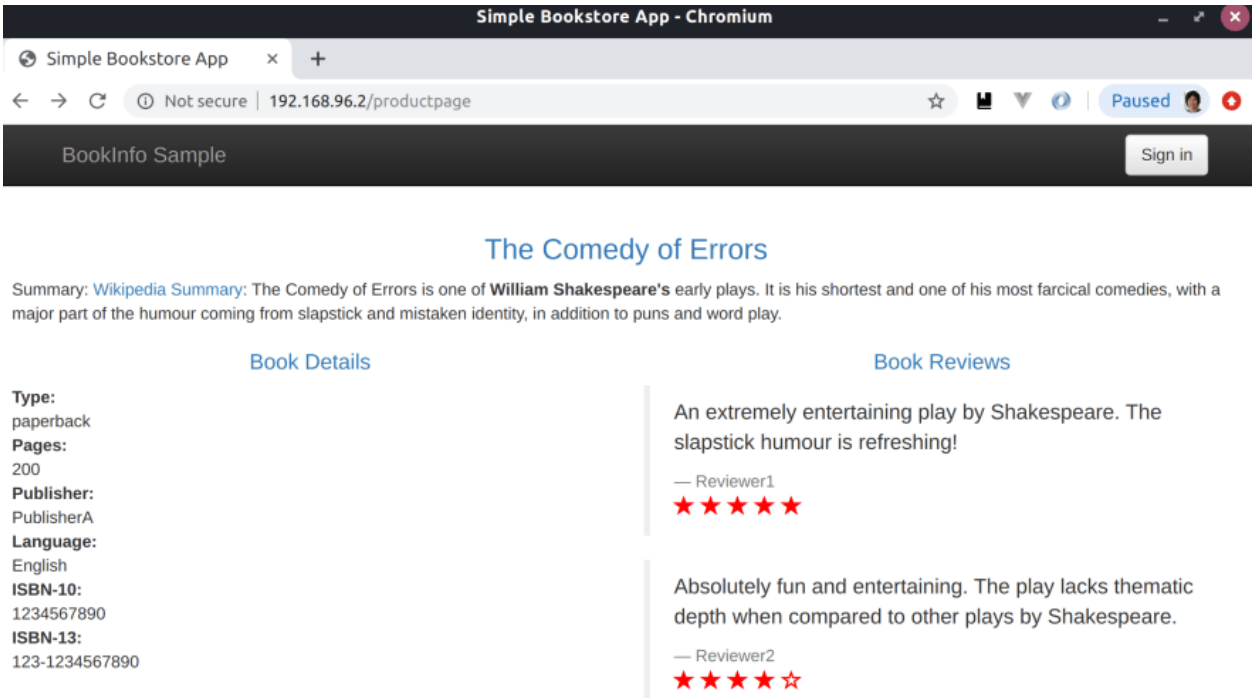
192.168.96.2

View application

Opened that IP in following URL for the bookinfo application

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
```

(Optional) Attach tcpdump to our container network and browse the application to see the communications that are occurring.