## Exercise 11.1: Service Mesh

If you have a large number of services to expose outside of the cluster, or to expose a low-number port on the host node you can deploy an ingress controller. While nginx and GCE have controllers mentioned a lot in Kubernetes.io, there are many to chose from. Even more functionality and metrics come from the use of a service mesh, such as Istio, Linkerd, Contour, Aspen, or several others.

1. We will install linkerd using their own scripts. There is quite a bit of output. Instead of showing all of it the output has been omitted. Look through the output and ensure that everything gets a green check mark. Some steps may take a few minutes to complete. Each command is listed here to make install easier. As well these steps are in the `setupLinkerd.txt` file.

```
student@cp:˜$ curl -sL run.linkerd.io/install-edge | sh

student@cp:˜$ export PATH=$PATH:/home/student/.linkerd2/bin

student@cp:˜$ echo "export PATH=$PATH:/home/student/.linkerd2/bin" >> $HOME/.bashrc

student@cp:˜$ linkerd check --pre

student@cp:˜$ kubectl apply -f https://github.com/kubernetes-sigs/gateway-api/releases/download/v1.2.1/standard-i

student@cp:˜$ linkerd install --crds | kubectl apply -f -

student@cp:˜$ linkerd install | kubectl apply -f -

student@cp:˜$ linkerd check

student@cp:˜$ linkerd viz install | kubectl apply -f -

student@cp:˜$ linkerd viz check

student@cp:˜$ linkerd viz dashboard &
```

2. By default the GUI is on available on the localhost. We will need to edit the service and the deployment to allow outside access, in case you are using a cloud provider for the nodes. Edit to remove all characters after equal sign for `-enforced-host`, which is around line 59.

```
student@cp:˜$ kubectl -n linkerd-viz edit deploy web
```

YAML
```yaml
spec:
      containers:
      - args:
        - -linkerd-controller-api-addr=linkerd-controller-api.linkerd.svc.cluster.local:8085
        - -linkerd-metrics-api-addr=metrics-api.linkerd-viz.svc.cluster.local:8085
        - -cluster-domain=cluster.local
        - -grafana-addr=grafana.linkerd-viz.svc.cluster.local:3000
        - -controller-namespace=linkerd
        - -viz-namespace=linkerd-viz
        - -log-level=info
#         - -enforced-host=                                 #<-- Comment the line by adding #
        image: cr.l5d.io/linkerd/web:stable-2.11.1
        imagePullPolicy: IfNotPresent
```

3. Now edit the http nodePort and type to be a NodePort.

```
student@cp:˜$ kubectl edit svc web -n linkerd-viz
```

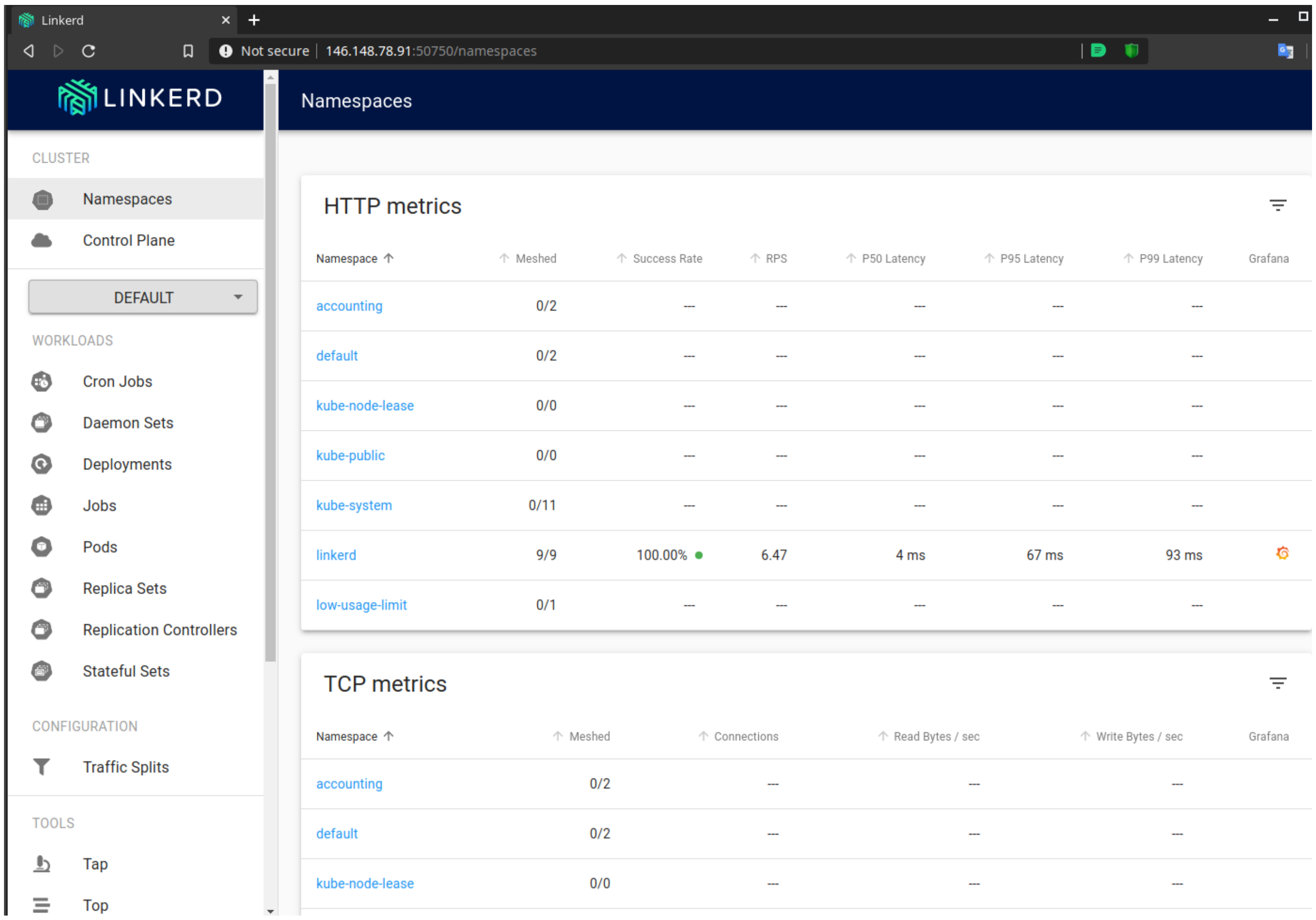
```yaml
....
ports:
  - name: http
    nodePort: 31500                                     #<-- Add line with an easy to remember port
    port: 8084
....
  sessionAffinity: None
  type: NodePort                                        #<-- Edit type to be NodePort
status:
  loadBalancer: {}
....
```

4. Test access using a local browser to your public IP. Your IP will be different than the one shown below.

```
student@cp:˜$ curl ifconfig.io
```

```
104.197.159.20
```

5. From you local system open a browser and go to the public IP and the high-number nodePort. Be aware the look of the web page may look slightly different as the software is regularly updated, for example Grafana is not longer fully integrated.



Namespaces

HTTP metrics

| Namespace | Meshed | Success Rate | RPS | P50 Latency | P95 Latency | P99 Latency | Grafana |
|---|---|---|---|---|---|---|---|
| accounting | 0/2 | — | — | — | — | — | |
| default | 0/2 | — | — | — | — | — | |
| kube-node-lease | 0/0 | — | — | — | — | — | |
| kube-public | 0/0 | — | — | — | — | — | |
| kube-system | 0/11 | — | — | — | — | — | |
| linkerd | 9/9 | 100.00% ● | 6.47 | 4 ms | 67 ms | 93 ms | |
| low-usage-limit | 0/1 | — | — | — | — | — | |

TCP metrics

| Namespace | Meshed | Connections | Read Bytes / sec | Write Bytes / sec | Grafana |
|---|---|---|---|---|---|
| accounting | 0/2 | — | — | — | |
| default | 0/2 | — | — | — | |
| kube-node-lease | 0/0 | — | — | — | |

Figure 11.3: **Main Linkerd Page**

6. In order for linkerd to pay attention to an object we need to add an annotation. The **linkerd inject** command will do this for us. Generate YAML and pipe it to **linkerd** then pipe again to **kubectl**. Expect an error about how the object was created, but the process will work. The command can run on one line if you omit the back-slash. Recreate the `nginx-one` deployment we worked with in a previous lab exercise.

```
student@cp:˜$ kubectl get ns accounting       ## Verify namespace exists

student@cp:˜$ kubectl label node worker<TAB> system=secondOne     ## Re-label the node

student@cp:˜$ vim nginx-one.yaml                ## Validate or correct containerPort: 80 (not 8080)

student@cp:˜$ kubectl apply -f nginx-one.yaml       ## Re-deploy nginx-one application

student@cp:˜$  kubectl -n accounting get deploy nginx-one -o yaml | \
     linkerd inject - | kubectl apply -f -
```

```
<output_omitted>
```

7. Check the GUI, you should see that the `accounting` namespaces and pods are now meshed, and the name is a link.

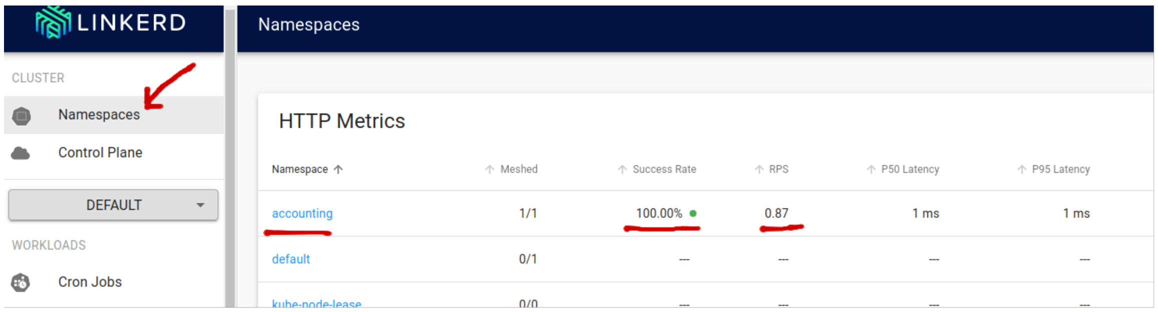
8. Generate some traffic to the pods, and watch the traffic via the GUI. Use the `service-lab` service.

```
student@cp:˜$ kubectl -n accounting get svc
```

```
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
nginx-one     ClusterIP   10.107.141.227  <none>        8080/TCP       5h15m
service-lab   NodePort    10.102.8.205    <none>        80:30759/TCP   5h14m
```

```
student@cp:~$ curl 10.102.8.205
```

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<output_omitted>
```



HTTP Metrics

| Namespace | Meshed | Success Rate | RPS | P50 Latency | P95 Latency |
|---|---|---|---|---|---|
| accounting | 1/1 | 100.00% ● | 0.87 | 1 ms | 1 ms |
| default | 0/1 | — | — | — | — |
| kube-node-lease | 0/0 | — | — | — | — |

Figure 11.4: **Now shows meshed**

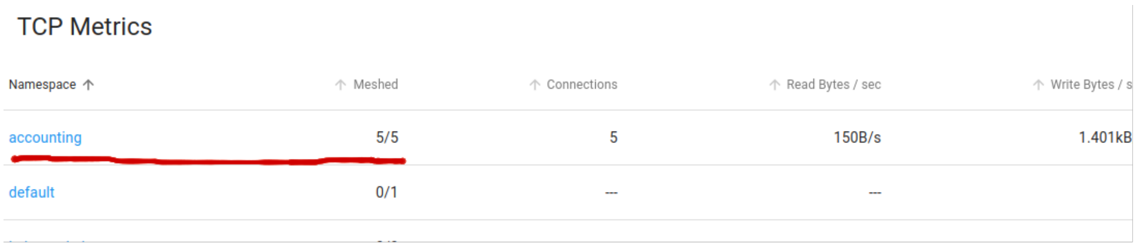
9. Scale up the `nginx-one` deployment. Generate traffic to get metrics for all the pods.

```
student@cp:~$ kubectl -n accounting scale deploy nginx-one --replicas=5
```

```
deployment.apps/nginx-one scaled
```

```
student@cp:~$ curl 10.102.8.205    #Several times
```

10. Explore some of the other information provided by the GUI. Note that the initial view is of the `default` namespaces. Change to `accounting` to see details of the nginx-one deployment.



TCP Metrics

| Namespace | Meshed | Connections | Read Bytes / sec | Write Bytes / s |
|---|---|---|---|---|
| accounting | 5/5 | 5 | 150B/s | 1.401kB |
| default | 0/1 | — | — | |

Figure 11.5: **Five meshed pods**