Kubernetes

Services Challenge Solutions

3. CHALLENGE - Service selectors

Without changing any of the running resources, create a new deployment that runs the `httpd` image with 2 replicas such that the service will send traffic to it as well.

Test your service to verify proper operation using curl in a test container.

Create the new deployment imperatively:

```
~/svc$ kubectl create deploy httpd --image=httpd --dry-run=client --
port=80 --replicas 2 -o yaml > httpd.yaml

~/svc$
```

Edit the manifest:

- Change the value of the `app:` key to `testweb`

```
~/svc$ nano httpd.yaml && cat httpd.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: testweb              # Change this
  name: httpd
spec:
  replicas: 2
  selector:
    matchLabels:
      app: testweb            # Change this
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
```

```
        app: testweb          # Change this
    spec:
      containers:
      - image: httpd:latest
        name: httpd
        ports:
        - containerPort: 80
        resources: {}
status: {}
```

```
~/svc$
```

Apply the manifest:

```
~/svc$ kubectl apply -f httpd.yaml

deployment.apps/httpd created

~/svc$
```

Retrieve pods labeled `app=testweb`:

```
~/svc$ kubectl get pods -l app=testweb

NAME                          READY   STATUS    RESTARTS   AGE
bigwebstuff                   1/1     Running   0          2m26s
bigwebstuff-cb98f5c6c-4tjqt   1/1     Running   0          59s
bigwebstuff-cb98f5c6c-blh7m   1/1     Running   0          59s
bigwebstuff-cb98f5c6c-vdcpr   1/1     Running   0          59s
httpd-df4bfd447-22lr6         1/1     Running   0          7s
httpd-df4bfd447-bkjt2         1/1     Running   0          7s

~/svc$
```

Check the total endpoints:

```
~/svc$ kubectl get ep testweb

NAME      ENDPOINTS                                               AGE
testweb   10.32.0.10:80,10.32.0.5:80,10.32.0.6:80 + 3 more...   3m23s

~/svc$
```

You should have 6 running pods and six total endpoints for your service. Now test w/ our client pod.

Attach to the testclient pod:

```
~/svc$ kubectl exec testclient -c testclient -- wget -O - $SVC
```

Or if it isn't running, run it:

```
~/svc$ kubectl run -it testclient --image=busybox:1.27 --env SVC=$SVC
```

Curl (or wget) the Service IP; you should receive the nginx welcome page sometimes and other times the apache "It works!" page. If you don't see each one right away, keep trying (remember, the iptables load balancing is based on a randomizer).

```
/ # wget -O - $SVC

Connecting to 10.102.255.100 (10.102.255.100:80)
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed
and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
-                    100%
|*******************************************************|   615   0:00:00 ETA

/ # wget -O - $SVC

Connecting to 10.102.255.100 (10.102.255.100:80)
<html><body><h1>It works!</h1></body></html>

/ # exit
```

```
Session ended, resume using 'kubectl attach testclient -c testclient -i -
t' command when the pod is running

~/svc$
```

- Clean up the related resources once finished

```
~/svc$ kubectl delete -f httpd.yaml

deployment.apps "httpd" deleted

~/svc$
```