Goal

Create a Kubernetes Cluster service and access it from other Pods in the cluster

Note: Run the lab from the same directory where you found this instructions file

Steps

Create Namespaces

We will create one namespace that hosts a service for firewall documentation and another that hosts a service for ExpressRoute documentation and label these services

//Firewall

```
kubectl create namespace firewall
kubectl label namespace/firewall purpose=firewall-content
```

//ExpressRoute

```
kubectl create namespace expressroute
kubectl label namespace/expressroute purpose=expressroute-content
```

Create Deployments

We will create a firewall content deployment in the firewall namespace and an ExpresRoute content deployment in the ExpressRoute namespace. These are Kubernetes deployments consisting of 3 Pods each serving the content. The content on each Pod is accessible on port 80. The yaml files for these deployment specifications are in the "Exercise3 – Create K8S Service" directory

1. Create Firewall content Deployment

```
kubectl apply -f deployment-firewall.yaml -n firewall
```

To check the status of all the containers in the deployment run the following

```
kubectl get deployments -n firewall
```

To check all the pods that have been created run the following

```
kubectl get pods -n firewall
```

2. Create ExpressRoute content deployment

```
kubectl apply -f deployment-expressroute.yaml -n expressroute
```

To check the status of all the containers in the deployment run the following

```
kubectl get deployments -n expressroute
```

To check all the pods that have been created run the following

```
kubectl get pods -n expressroute
```

Create Kubernetes Cluster Services

Now we will create a firewall-content service and an ExpressRoute content services from these deployments

1. Create Firewall content service

The following command creates a service serving on port 80 and connects to the backend firewall content pods on port 80.

```
kubectl expose deployment firewall-content-demo-deployment -n firewall
--name=firewall-content-service --port=80 --target-port=80
```

To check the cluster services that you created run the following

```
kubectl get services -n firewall
```

2. Create ExpressRoute content service

The following command creates a service serving on port 80 and connects to the backend ExpressRoute content pods on port 80.

```
kubectl expose deployment expressroute-content-demo-deployment -n
expressroute --name=expressroute-content-service --port=80 --target-
port=80
```

To check the cluster services that you created run the following

```
kubectl get services -n expressroute
```

Access these Cluster Services

We will now create an Alpine Pod in the Firewall namespace and access both the above services from it

1. Create sample Pod in Firewall namespace

   Create an Alpine Pod and get shell access.

   ```
   kubectl run samplepod --rm -it --image=alpine --namespace firewall -
   -generator=run-pod/v1
   ```

2. Access the Services from this pod

   wget downloads the http file of the website

   ```
   wget -qO- http://firewall-content-service
   wget -qO- http://expressroute-content-service.expressroute
   ```

   The services can be similarly accessed from a Pod in the ExpressRoute namespace

3. Exit the Pod

   This is done by typing “exit” at the shell prompt

Create Kubernetes Loadbalancer Service

Now we will create a loadbalancer service for the firewall-content deployment. The following command creates a service serving on port 80 and connects to the backend firewall content pods on port 80.

```
kubectl apply -f firewall-content-lbservice.yaml
```

To check the loadbalancer service that you created run the following

```
kubectl get services -n firewall
```

You will see a new service of type LoadBalancer. The status of “External-IP” might say “pending”. This is because Kubernetes is in the process of provisioning the Azure SLB and fetch a public IP for this service. It will take a couple of minutes to get the public IP.

Accessing the Service

Once you get the Public IP make a note of it. You can access the service from your browser by going to http:<PublicIP>