Goal

In this lab we will create two kinds of policies

1. A policy that ensures that the services created in the previous lab are only accessible from Pods in the same namespace as the service
2. A policy that allows the firewall-content service to be accessible from pods in the ExpressRoute namespace having a specific label

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

Service Accessible from Pods in its Namespace

Apply the Policies

1. Apply the Policy in Firewall Namespace

```
kubectl apply -f firewall-content-policy-SameNS.yaml -n firewall
```

To check the policy applied run the following

```
kubectl describe -f firewall-content-policy-SameNS.yaml -n firewall
```

2. Apply the Policy in ExpressRoute Namespace

```
kubectl apply -f expressroute-content-policy-SameNS.yaml -n expressroute
```

To check the policy applied run the following

```
kubectl describe -f expressroute-content-policy-SameNS.yaml -n expressroute
```

Access the Services from Pods in their respective namespace

We will now create an Alpine Pod in the Firewall namespace and try accessing the two services from it. The same steps can be repeated to test access from Alpine Pod in ExpressRoute namespace too.

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

   Run the following wget commands from the shell. wget downloads the http file of the website

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

Accessing firewall content service will pass but accessing expressroute content service will fail to get the content.

Cleanup

1. Exit the Pod
   This is done by typing “exit” at the shell prompt

2. Delete the Policies

```
kubectl delete -f firewall-content-policy-SameNS.yaml -n firewall

kubectl delete -f expressroute-content-policy-SameNS.yaml -n expressroute
```

Firewall Content Service Accessible from Pod in ExpressRoute Namespace

Apply the Policy in Firewall Namespace

This policy allows Pods with a label status=exception and running in ExpressRoute namespace to access the Firewall content service. It also allows Pods running in the Firewall namespace to access the content

```
kubectl apply -f firewall-content-policy-exception.yaml -n firewall
```

To check the policy applied run the following

```
kubectl describe -f firewall-content-policy-exception.yaml -n firewall
```

Accessing Firewall Content Service

1. Accessing from Pod in Firewall Namespace
   Create a sample Pod in firewall namespace

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

   Verify that you are able to access the firewall-content. Run the following wget command from the Alpine Pod shell

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

   Exit the Pod by typing “exit” at the shell

2. Accessing from Pod in ExpressRoute Namespace
   Create a sample Pod in expressroute namespace without the status=exception label

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

   Verify that you are not able to access the firewall-content. Observe that the following wget command from the Alpine Pod shell fails to get the content

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

   Exit the Pod by typing “exit” at the shell

3. Accessing from Pod with allowed label in ExpressRoute Namespace

Create a sample Pod in expressroute namespace with the status=exception label

```
kubectl run samplepod --rm -it --image=alpine --labels
status=exception --namespace expressroute --generator=run-pod/v1
```

Verify that you are now able to access the firewall-content. Run the following wget command from the Alpine Pod shell

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

Exit the Pod by typing “exit” at the shell

Cleanup

1. Delete the Policies

```
kubectl delete -f firewall-content-policy-exception.yaml -n firewall
```

2. Delete the Services

```
kubectl delete service firewall-content-service -n firewall
kubectl delete services firewall-content-lbservice -n firewall
kubectl delete service expressroute-content-service -n expressroute
```

3. Delete the Deployments

```
kubectl delete deployment firewall-content-demo-deployment -n
firewall
kubectl delete deployment expressroute-content-demo-deployment -n
expressroute
```

Appendix

If for some reason policies don't work then perform the following steps and try again

1. Delete all the policies – steps for this are covered above

2. Run the following command

   ```
   kubectl delete -f https://raw.githubusercontent.com/Azure/aks-engine/master/parts/k8s/addons/kubernetesmasteraddons-azure-npm-daemonset.yaml --grace-period=0 --force && kubectl apply -f https://raw.githubusercontent.com/Azure/aks-engine/master/parts/k8s/addons/kubernetesmasteraddons-azure-npm-daemonset.yaml
   ```

3. Re-apply the policy yamls and continue your lab exercises