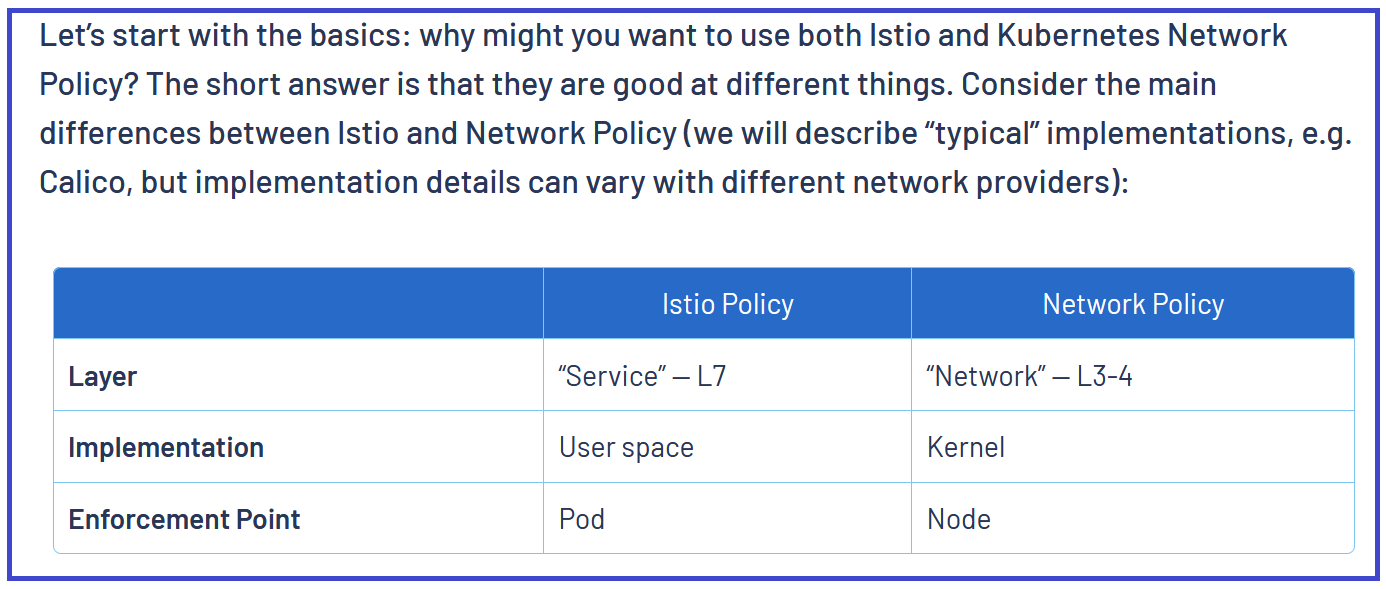
**Filtering network traffic**

First of all, you need to think about what kind of network traffic you want to filter. The easiest way filtering traffic is on Layer 3 and 4, which is at the IP address and port level. If you’d like to filter on http/https hostname, then you’ll need Layer 7 filtering, which is currently not supported by Kubernetes Network Policies. For that, you would need a separate proxy server within your cluster.

We’ll keep Layer 7 filtering for another blog post, as this is currently much more complicated, because you cannot use Network Policies for that. You would need an ingress/egress proxy that understands Layer 7 (http/https) traffic, like Envoy proxy. More traditional reverse proxies like haproxy/nginx could also be used for this.



#!/bin/bash

#Deploying in multiple Availability Zones will deploy NAT gateway in each zone & also a Load Balancer for the Master Server. Be careful....

kops create cluster --name=trainingk8s.xyz \

--state=s3://sreek8s --zones=us-east-1a \

--node-count=3 --node-size=t3.medium --master-size=t3.medium \

--master-volume-size 10 --node-volume-size 10 \

--topology private --networking calico \

--dns-zone=trainingk8s.xyz --yes

https://docs.giantswarm.io/guides/limiting-pod-communication-with-network-policies/

<https://docs.projectcalico.org/security/kubernetes-default-deny>

<https://medium.com/flant-com/calico-for-kubernetes-networking-792b41e19d69>

###Very Good Repo For Practice

https://github.com/ahmetb/kubernetes-network-policy-recipes

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: web-deny-all

namespace: development

spec:

podSelector:

matchLabels:

env: dev

ingress: []

---

#Block Ingress & Engress

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: web-deny-all

spec:

podSelector:

matchLabels:

env: frontend

policyTypes:

- Egress

- Ingress

ingress: []

egress: []

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: web-allow-from-default

namespace: testing

spec:

podSelector:

matchLabels:

env: testing

ingress:

- from:

- podSelector:

matchLabels:

env: default

- namespaceSelector:

matchLabels:

env: default

ports:

- protocol: TCP

port: 8000

- protocol: TCP

port: 8888

#Block All Ingress

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: app1-deny-all

namespace: app1

spec:

podSelector:

matchLabels:

env: nginx

ingress: []

#Allow all PODS in the namespace app1 to

#communicate with app1 pod in same namesapce

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: allow-local-ns-pods-only #Make sure the app pod has label access1: local1

namespace: app1

spec:

podSelector:

matchLabels:

access1: local1

ingress:

- from:

- podSelector:

matchLabels:

access1: local1

#Allowing namespace test and pods with label access1: local1 to

#communicate with app1 pod in namespace app1

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: allow-from-ns-test

namespace: app1

spec:

podSelector:

matchLabels:

access1: local1

ingress:

- from:

- namespaceSelector:

matchLabels:

env: test

podSelector:

matchLabels:

access1: local1

#Allowing namespace prod and pods with label access1: local1 to

#communicate with app1 pod in namespace app1

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: allow-from-ns-prod

namespace: app1

spec:

podSelector:

matchLabels:

access1: local1

ingress:

- from:

- namespaceSelector:

matchLabels:

env: prod

podSelector:

matchLabels:

access1: local1

#====================EGRESS================================#

#Block All Egress

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: allow-egress

namespace: prod

spec:

podSelector:

matchLabels:

env: prod

policyTypes:

- Egress

egress: []

---

kind: NetworkPolicy

apiVersion: networking.k8s.io/v1

metadata:

name: allow-egress-from-prod-to-app1

namespace: prod

spec:

podSelector:

matchLabels:

access1: local1

policyTypes:

- Egress

egress:

- to:

- namespaceSelector:

matchLabels:

env: app1

podSelector:

matchLabels:

access1: local1

ports:

- protocol: TCP

port: 8000

- protocol: TCP

port: 80

---

#Allow communication inside namesapce

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: allow-same-namespace

namespace: test

spec:

podSelector: {}

ingress:

- from:

- namespaceSelector:

matchLabels:

name: test

egress:

- to:

- namespaceSelector:

matchLabels:

name: test

---

#Allow communication inside namespace

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: allow-same-namespace

namespace: prod

spec:

podSelector: {}

ingress:

- from:

- namespaceSelector:

matchLabels:

name: prod

egress:

- to:

- namespaceSelector:

matchLabels:

name: prod