Applying the NetworkPolicy on Docker Desktop Kubernetes

1. Enable Kubernetes on Docker Desktop

1. Open Docker Desktop.
2. Go to Settings → Kubernetes.
3. Check Enable Kubernetes.
4. Wait until Kubernetes finishes starting (it may take a few minutes).
5. Verify with:

```
kubectl cluster-info

kubectl get nodes
```

You should see a single node Kubernetes cluster ready.

---

2. Create a Namespace

Keep everything organized by creating a dedicated namespace:

```
kubectl create namespace local-test
```

---

3. Deploy a Sample Pod (platform-certificate-core)

Since you don't have the production pod locally, deploy a simple test pod to represent platform-certificate-core:

```
apiVersion: v1

kind: Pod

metadata:

  name: platform-certificate-core

  namespace: local-test
```

```
  labels:
    service: platform-certificate-core
spec:
  containers:
    - name: nginx
      image: nginx:latest
      ports:
        - containerPort: 80
```

Apply it:

```
kubectl apply -f platform-certificate-core.yaml
```

Check it:

```
kubectl get pods -n local-test --show-labels
```

---

4. Create the NetworkPolicy

Apply the NetworkPolicy you designed. Save it as certificate-core-network-policy.yaml:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: certificate-core-network-policy
  namespace: local-test
  labels:
    app: platform-certificate-core
```

```
    purpose: network-policy
    environment: local
spec:
  podSelector:
    matchLabels:
      service: platform-certificate-core
  policyTypes:
    - Ingress
  ingress:
    - from:
        - podSelector:
            matchLabels:
              app.kubernetes.io/name: ps-core
```

Apply it:

```
kubectl apply -f certificate-core-network-policy.yaml
```

Verify:

```
kubectl describe networkpolicy certificate-core-network-policy -n local-test
```

---

5. Test the NetworkPolicy

Allowed Pod

Create a test pod with the correct ps-core label:

```
apiVersion: v1
```

```
kind: Pod

metadata:

  name: test-allowed-pod

  namespace: local-test

  labels:

    app.kubernetes.io/name: ps-core

spec:

  containers:

    - name: busybox

      image: busybox

      command: ["sleep", "3600"]
```

Apply:

```
kubectl apply -f test-allowed-pod.yaml
```

Test connectivity:

```
kubectl exec -it -n local-test test-allowed-pod -- wget -qO- http://platform-certificate-core:80
```

Expected: Success.

Denied Pod

Create a test pod without the required label:

```
apiVersion: v1

kind: Pod

metadata:
```

```
  name: test-denied-pod
  namespace: local-test
  labels:
    app.kubernetes.io/name: other-app
spec:
  containers:
    - name: busybox
      image: busybox
      command: ["sleep", "3600"]
```

Apply:

```
kubectl apply -f test-denied-pod.yaml
```

Test connectivity:

```
kubectl exec -it -n local-test test-denied-pod -- wget -qO- http://platform-certificate-core:80
```

Expected: Fail (connection refused/timeout).

---

6. Clean Up

After testing:

```
kubectl delete namespace local-test
```