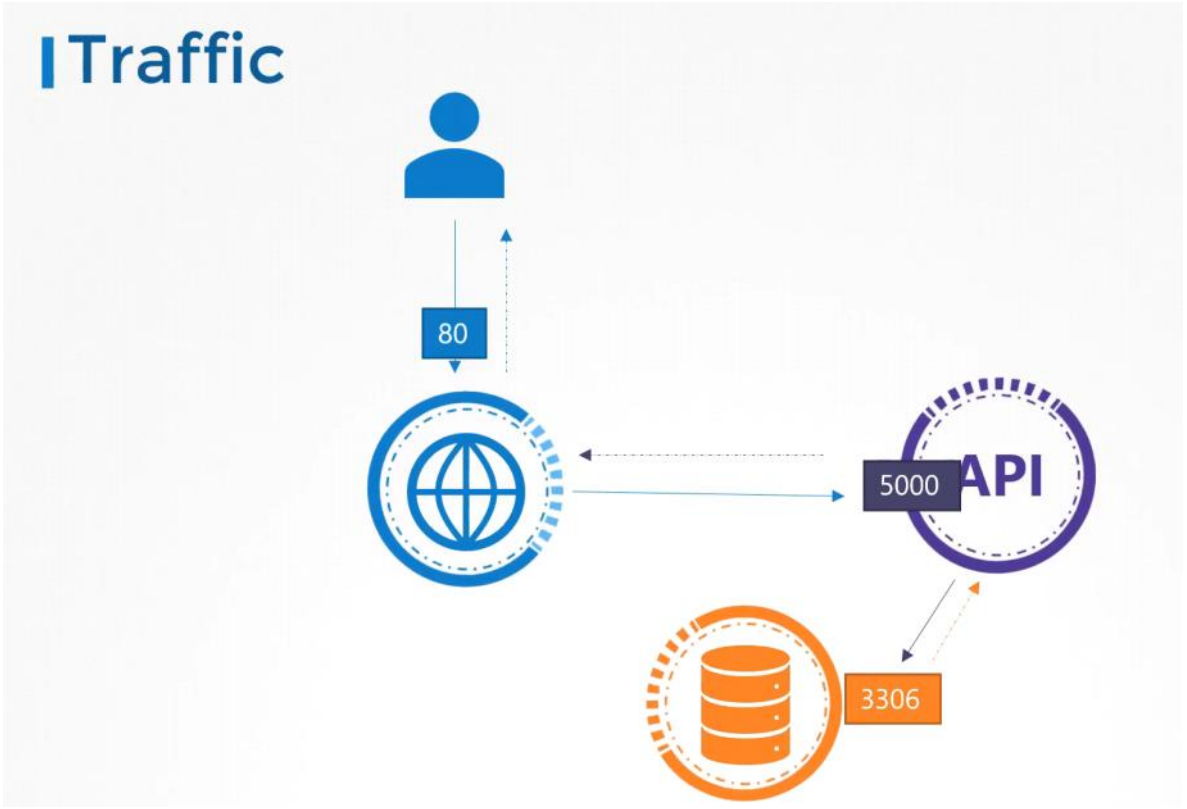
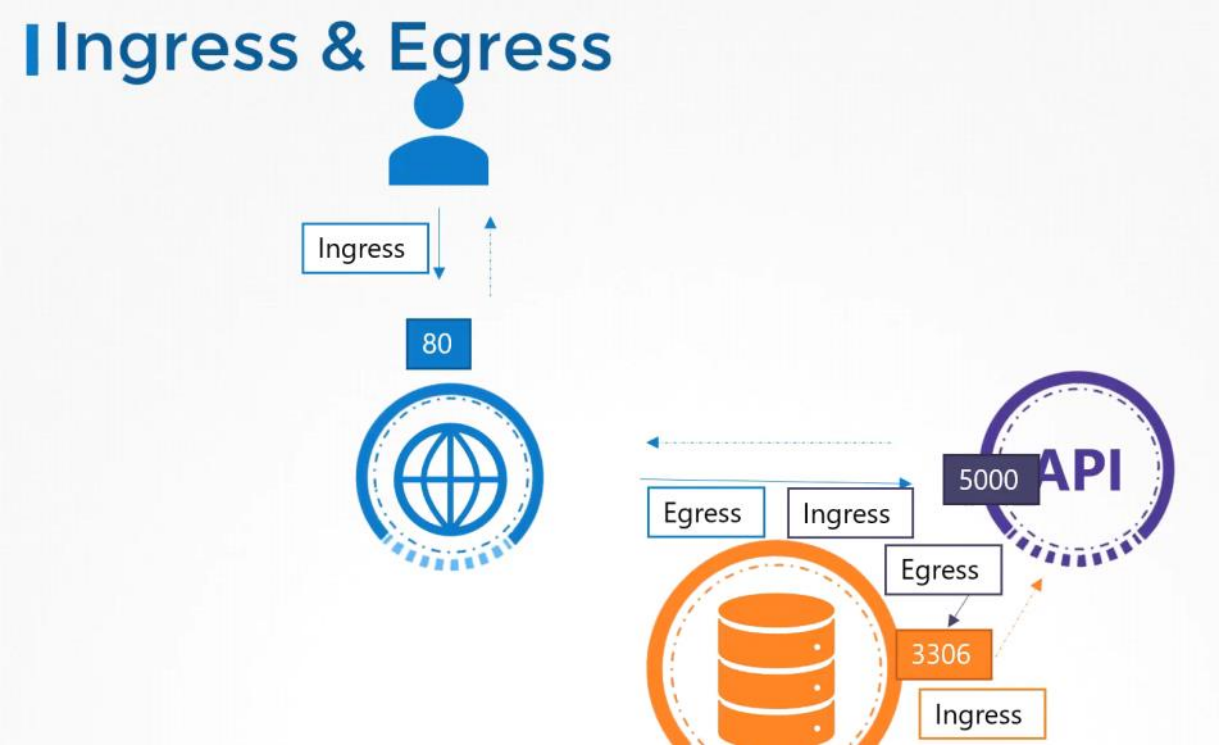
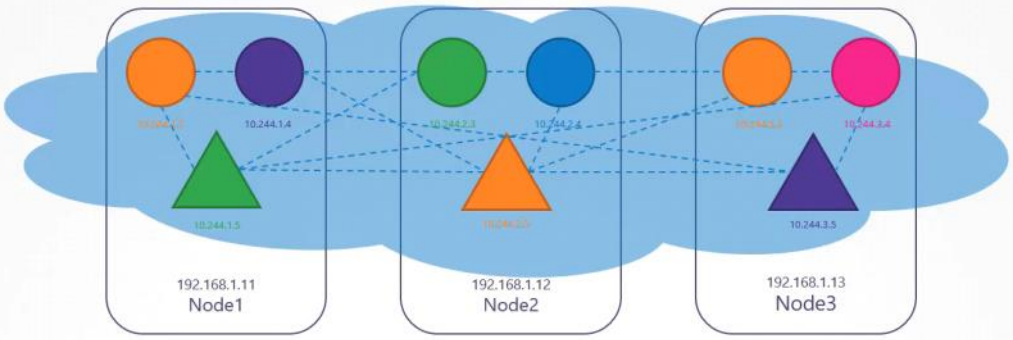
# Network Policies

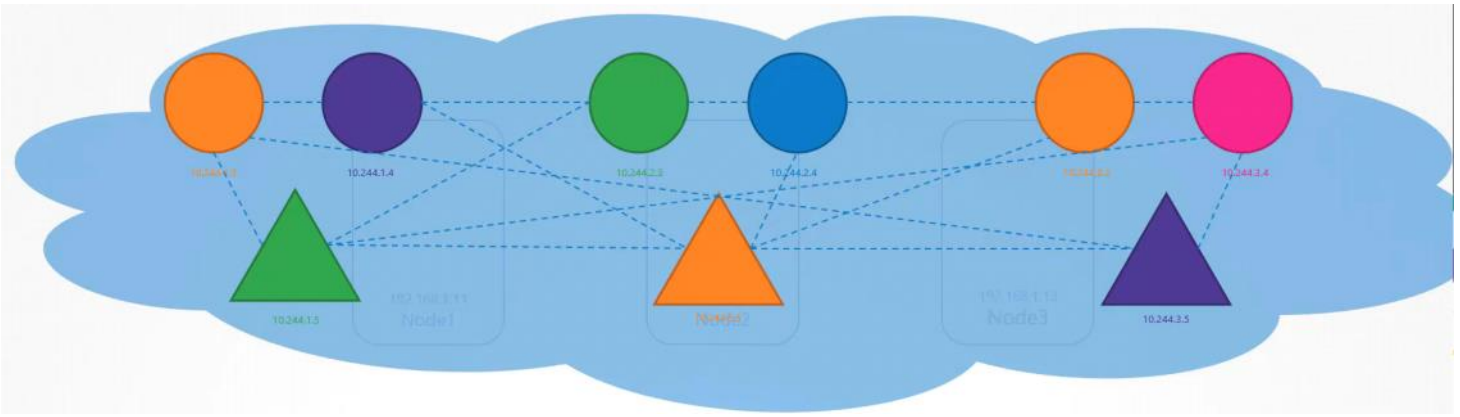



Ingress & Egress
Ingress
80
5000
API
Egress
Ingress
Egress
3306
Ingress


Traffic
1
Ingress
80
2
Egress
5000
3
Ingress
5000
4
Egress
3306
5
Ingress
3306
API

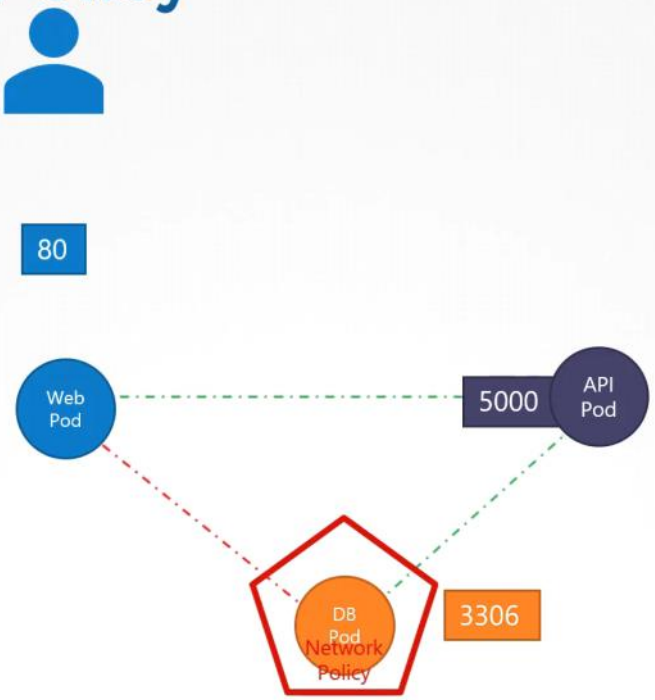
# Network Security



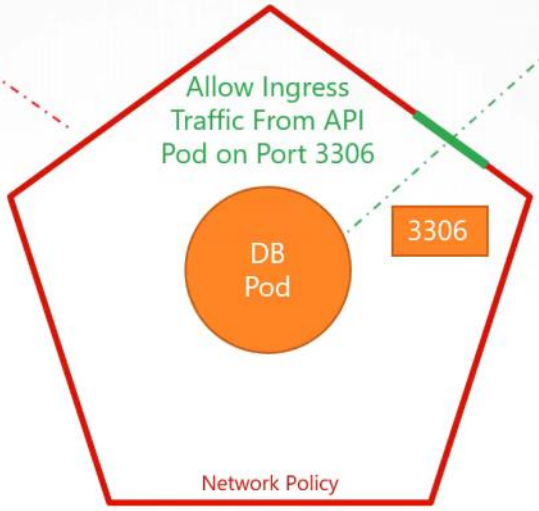




"All Allow"

# Network Policy

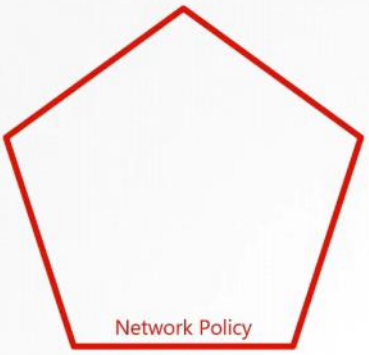


# Network Policy




# Network Policy - Selectors

Allow Ingress
Traffic From API
Pod on Port 3306





```
podSelector:
  matchLabels:
    role: db
```

```
labels:
  role: db
```

# Network Policy - Rules

```
policyTypes:
- Ingress
ingress:
- from:
  - podSelector:
      matchLabels:
        name: api-pod
  ports:
  - protocol: TCP
    port: 3306
```

Allow Ingress Traffic From API Pod on Port 3306

# Network Policy

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: db-policy
spec:
```

```
podSelector:
  matchLabels:
    role: db
```

```
policyTypes:
- Ingress
ingress:
- from:
  - podSelector:
      matchLabels:
        name: api-pod
  ports:
  - protocol: TCP
    port: 3306
```

# Network Policy

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: db-policy
spec:
   podSelector:
     matchLabels:
       role: db
   policyTypes:
   - Ingress
   ingress:
   - from:
     - podSelector:
         matchLabels:
           name: api-pod
     ports:
     - protocol: TCP
       port: 3306
```

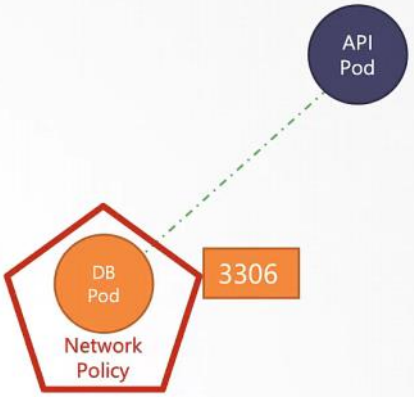
# Note

Solutions that Support Network Policies:
- Kube-router
- Calico
- Romana
- Weave-net

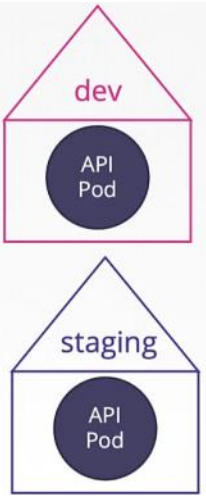
Solutions that DO NOT Support Network Policies:
- Flannel

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: db-policy
spec:
  podSelector:
    matchLabels:
      role: db

  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          name: api-pod
    ports:
    - protocol: TCP
      port: 3306
```

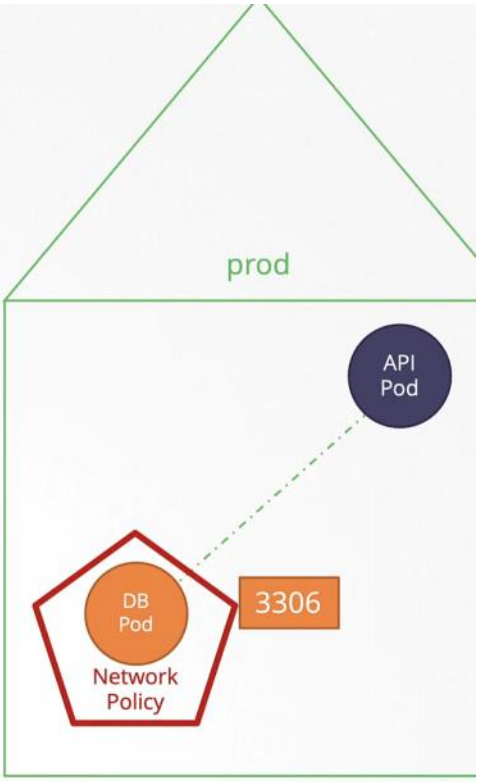


```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: db-policy
  namespace: prod
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          name: api-pod
    ports:
    - protocol: TCP
      port: 3306
```

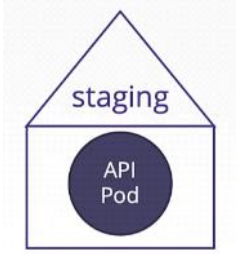


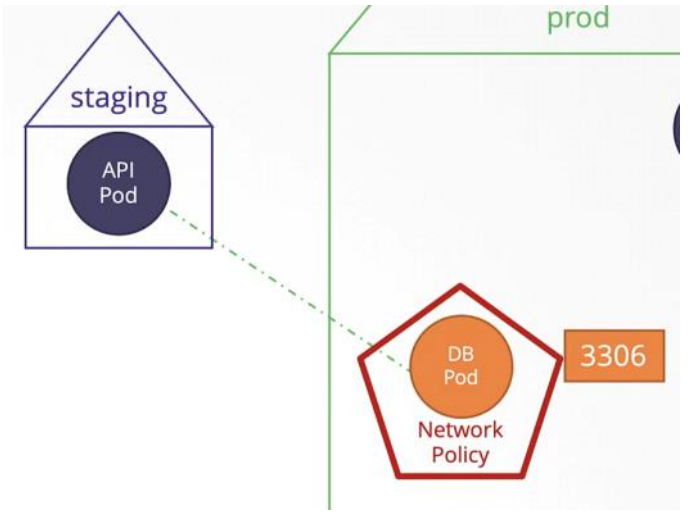




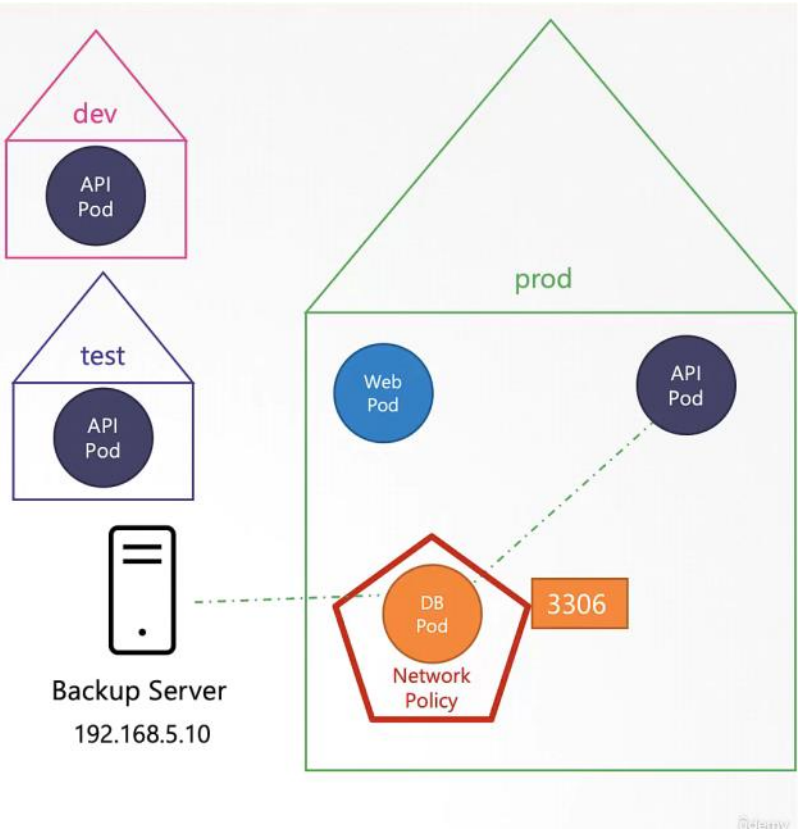



```yaml
ingress:
- from:
  - podSelector:
      matchLabels:
        name: api-pod
    namespaceSelector:
      matchLabels:
        name: staging
```



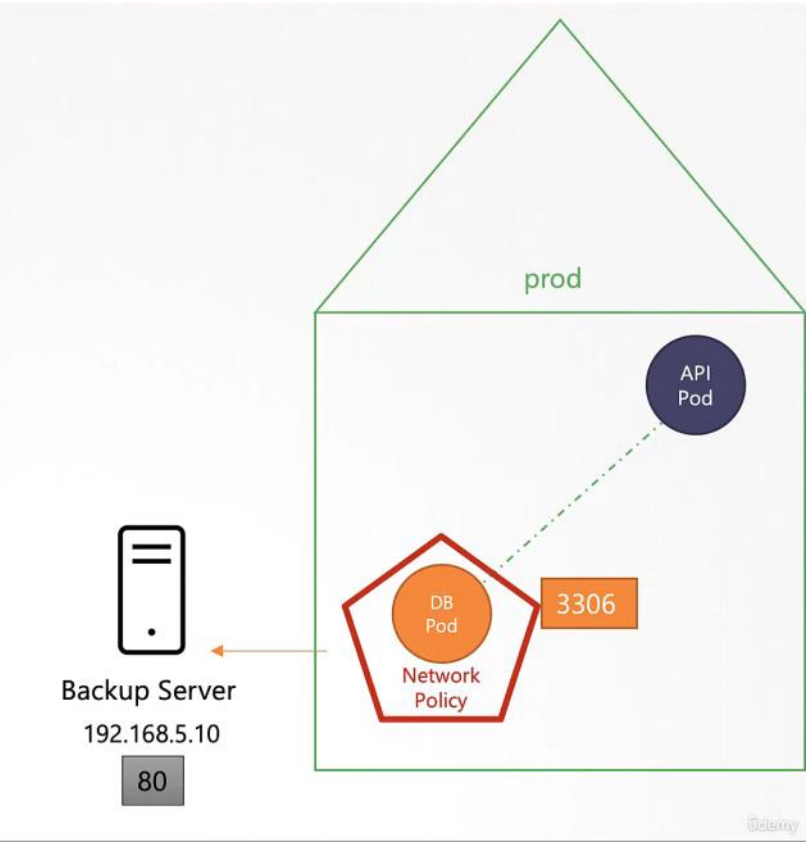



```yaml
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          name: api-pod
      namespaceSelector:
        matchLabels:
          name: prod
    - ipBlock:
        cidr: 192.168.5.10/32
    ports:
    - protocol: TCP
      port: 3306
```




```yaml
ingress:
- from:
  - podSelector:
      matchLabels:
        name: api-pod
  - namespaceSelector:
      matchLabels:
        name: prod
  - ipBlock:
      cidr: 192.168.5.10/32
```

```yaml
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          name: api-pod
    ports:
    - protocol: TCP
      port: 3306
  egress:
  - to:
    - ipBlock:
        cidr: 192.168.5.10/32
    ports:
    - protocol: TCP
      port: 80
```




**Kubectx and Kubens – Command line Utilities**
Through out the course, you have had to work on several different namespaces in the practice lab environments. In some labs, you also had to switch between several contexts.

While this is excellent for hands-on practice, in a real "live" kubernetes cluster implemented for production, there could be a possibility of often switching between a large number of namespaces and clusters.

This can quickly become and confusing and overwhelming task if you had to rely on kubectl alone.

This is where command line tools such as kubectx and kubens come in to picture.

Reference: https://github.com/ahmetb/kubectx

**Kubectx:**
With this tool, you don't have to make use of lengthy "kubectl config" commands to switch between contexts. This tool is particularly useful to switch context between clusters in a multi-cluster environment.

**Installation:**

```
sudo git clone https://github.com/ahmetb/kubectx /opt/kubectx
sudo ln -s /opt/kubectx/kubectx /usr/local/bin/kubectx
```

**Syntax:**

To list all contexts:

```
kubectx
```

To switch to a new context:

```
kubectx <context_name>
```

To switch back to previous context:

```
kubectx -
```

To see current context:

```
kubectx -c
```

**Kubens:**

This tool allows users to switch between namespaces quickly with a simple command.

**Installation:**

```
sudo git clone https://github.com/ahmetb/kubectx /opt/kubectx
sudo ln -s /opt/kubectx/kubens /usr/local/bin/kubens
```

**Syntax:**

To switch to a new namespace:

> kubens <new_namespace>

To switch back to previous namespace:

> kubens -