Deep Dive into Gateway API BackendTLSPolicy

(in 5 minutes)

31/03/2025 - Rejekts - London

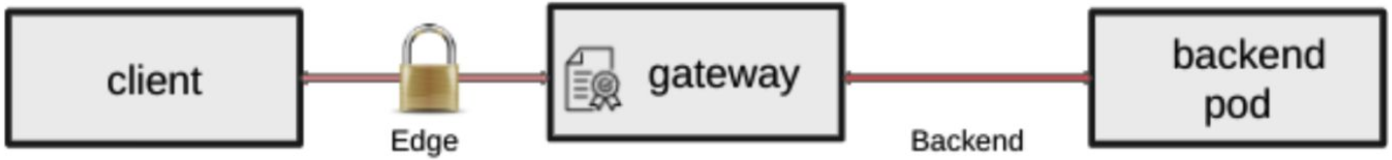
Nicolas Mengin

Traefik
Maintainer

Traefik Labs
Head of Support
(OSS & Enterprise)

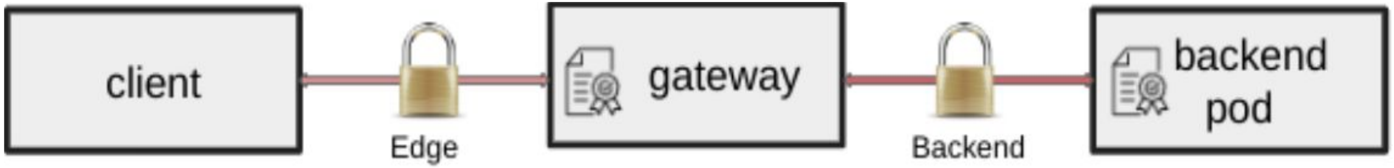
End to End Transport Layer Security

- Encrypt data transmitted…



End to End Transport Layer Security

- Encrypt data transmitted…
- …from the client to the backend

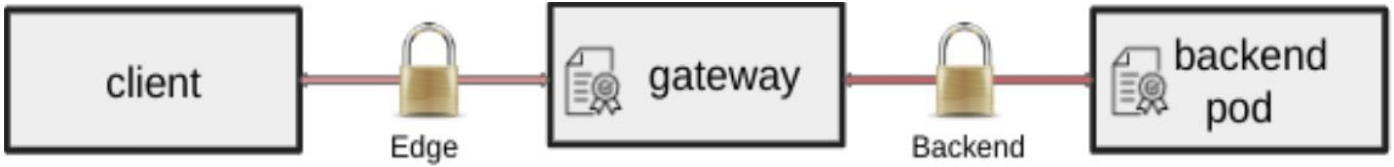


End to End Transport Layer Security

- Why a E2E Connection?
  - Zero Trust Security
  - HTTP2
  - gRPC
  - etc

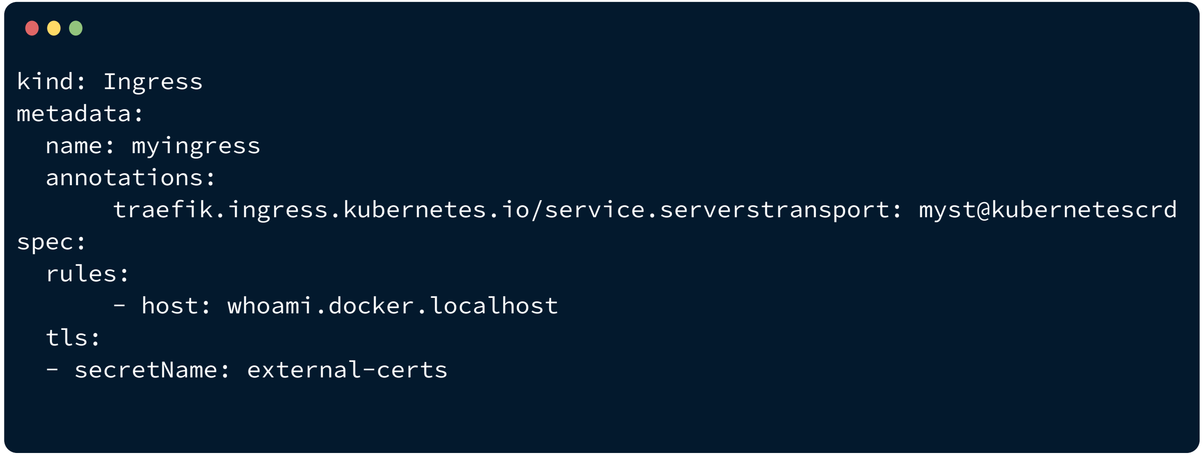
End to End Transport Layer Security

- How to do it?



End to End Transport Layer Security

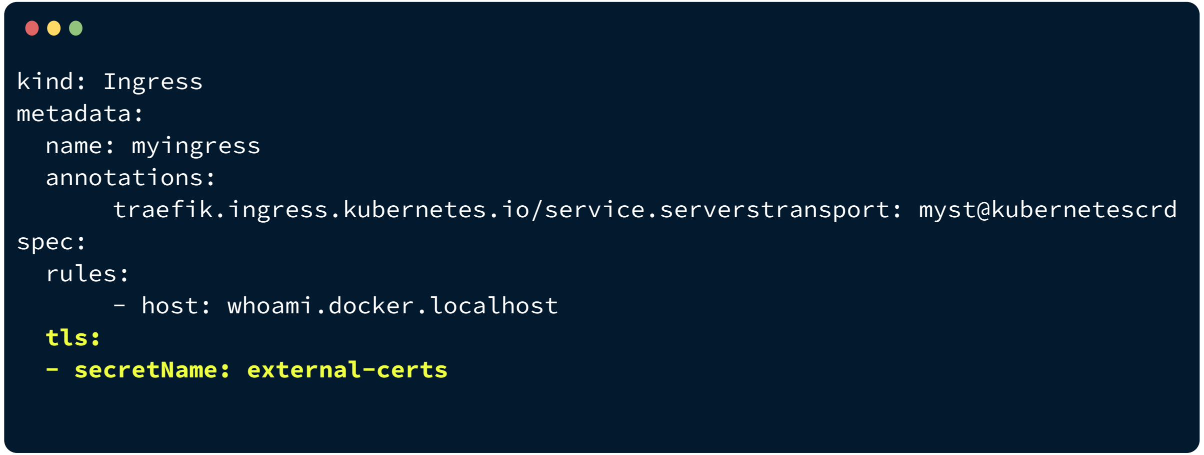
- How we do it in Kubernetes



```
kind: Ingress
metadata:
  name: myingress
  annotations:
      traefik.ingress.kubernetes.io/service.serverstransport: myst@kubernetescrd
spec:
  rules:
      - host: whoami.docker.localhost
  tls:
  - secretName: external-certs
```

End to End Transport Layer Security

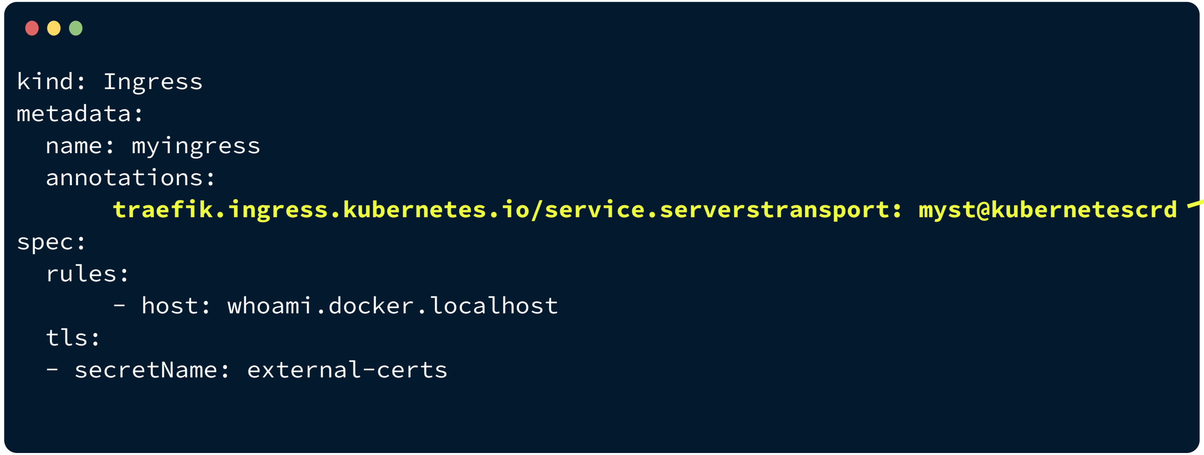
- How we do it in Kubernetes



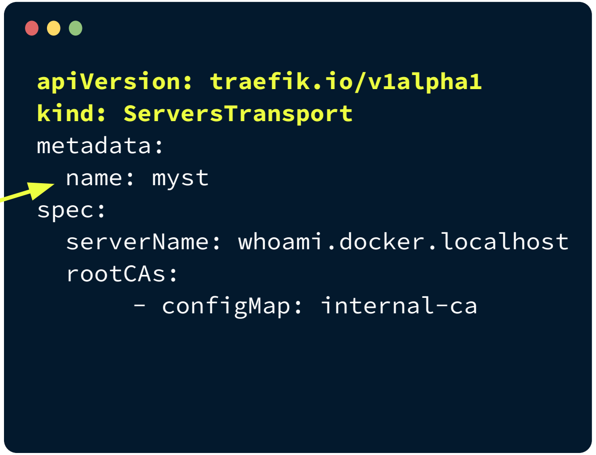
```
kind: Ingress
metadata:
  name: myingress
  annotations:
       traefik.ingress.kubernetes.io/service.serverstransport: myst@kubernetescrd
spec:
  rules:
       - host: whoami.docker.localhost
  tls:
  - secretName: external-certs
```

End to End Transport Layer Security

- How we do it in Kubernetes



```yaml
kind: Ingress
metadata:
  name: myingress
  annotations:
      traefik.ingress.kubernetes.io/service.serverstransport: myst@kubernetescrd
spec:
  rules:
      - host: whoami.docker.localhost
  tls:
  - secretName: external-certs
```



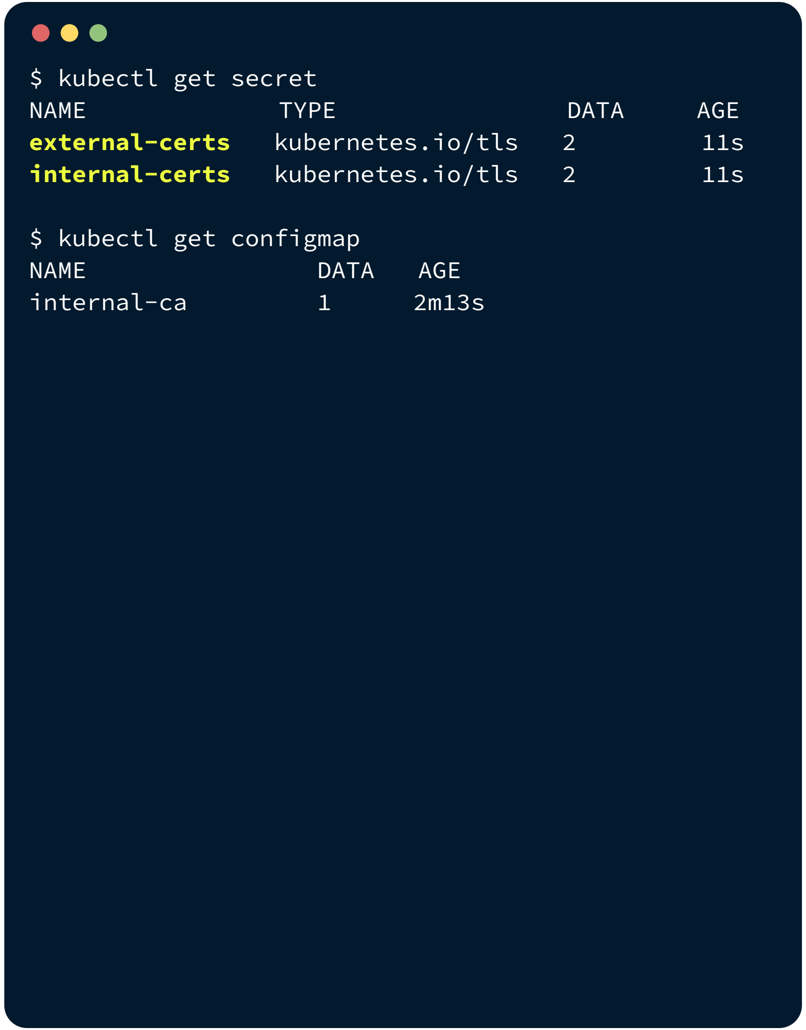
```yaml
apiVersion: traefik.io/v1alpha1
kind: ServersTransport
metadata:
  name: myst
spec:
  serverName: whoami.docker.localhost
  rootCAs:
      - configMap: internal-ca
```

BackendTLSPolicy at the rescue!

BackendTLSPolicy Recipe

BackendTLSPolicy Recipe

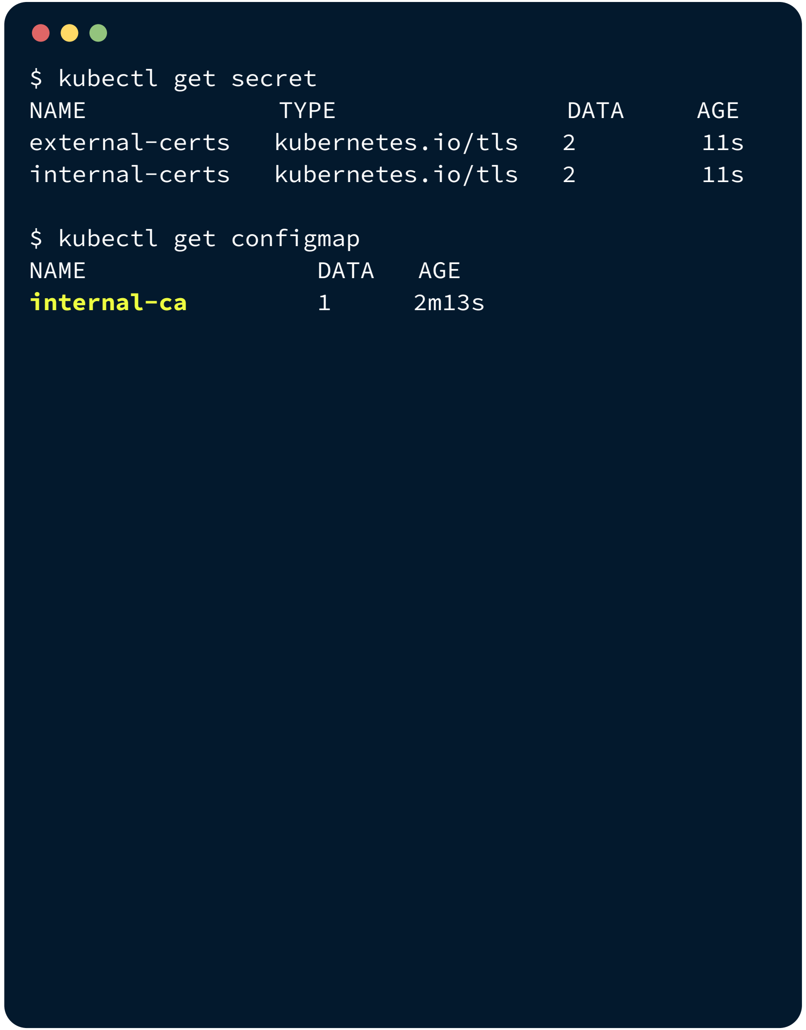
1. **TLS certificates + internal CA**
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)



```
$ kubectl get secret
NAME              TYPE                 DATA      AGE
external-certs    kubernetes.io/tls    2          11s
internal-certs    kubernetes.io/tls    2          11s

$ kubectl get configmap
NAME                    DATA    AGE
internal-ca             1      2m13s
```

BackendTLSPolicy Recipe

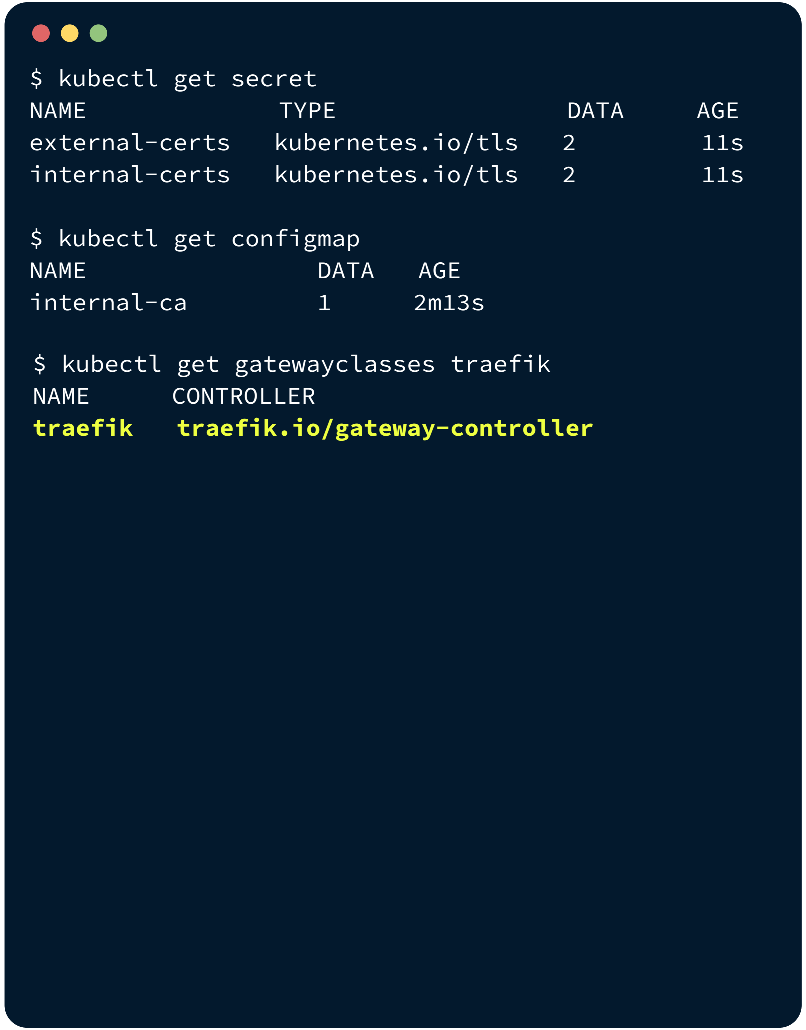
1. **TLS certificates + internal CA**
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)



```
$ kubectl get secret
NAME             TYPE                 DATA     AGE
external-certs   kubernetes.io/tls    2        11s
internal-certs   kubernetes.io/tls    2        11s

$ kubectl get configmap
NAME                DATA   AGE
internal-ca         1      2m13s
```

BackendTLSPolicy Recipe

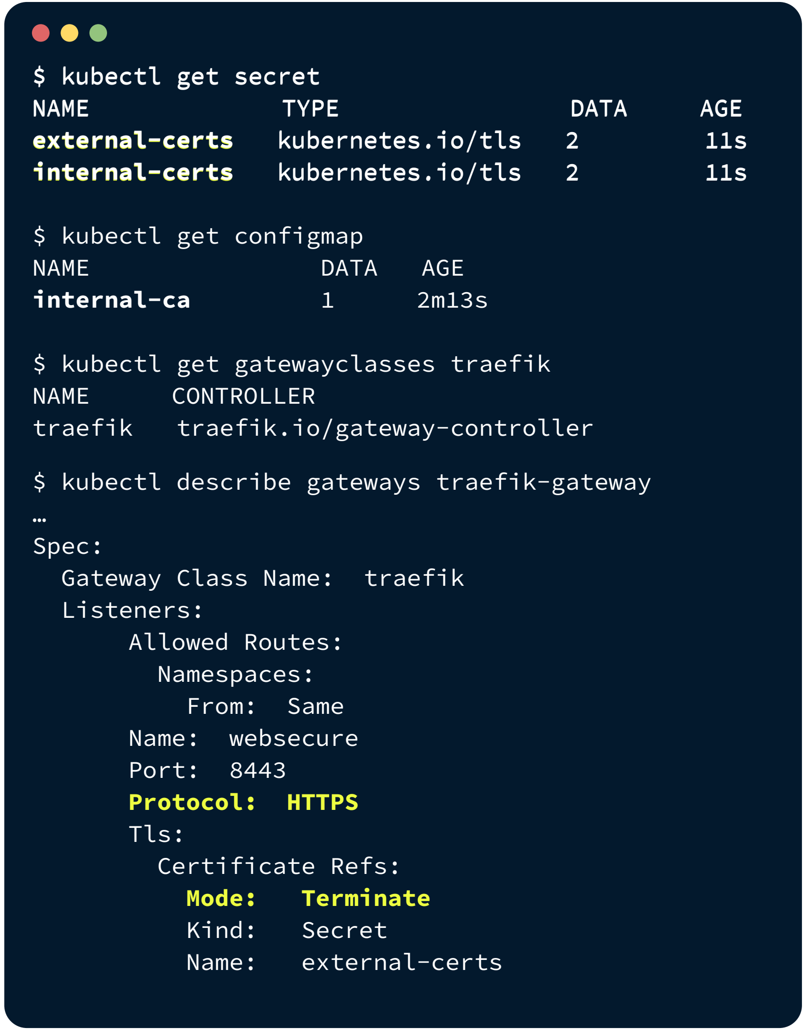
1. TLS certificates + internal CA
2. **Gateway Controller + Gateway Class**
3. Gateway (with HTTPS Listener)



```
$ kubectl get secret
NAME              TYPE                 DATA      AGE
external-certs    kubernetes.io/tls    2          11s
internal-certs    kubernetes.io/tls    2          11s

$ kubectl get configmap
NAME                  DATA    AGE
internal-ca           1      2m13s

$ kubectl get gatewayclasses traefik
NAME       CONTROLLER
traefik    traefik.io/gateway-controller
```

BackendTLSPolicy Recipe

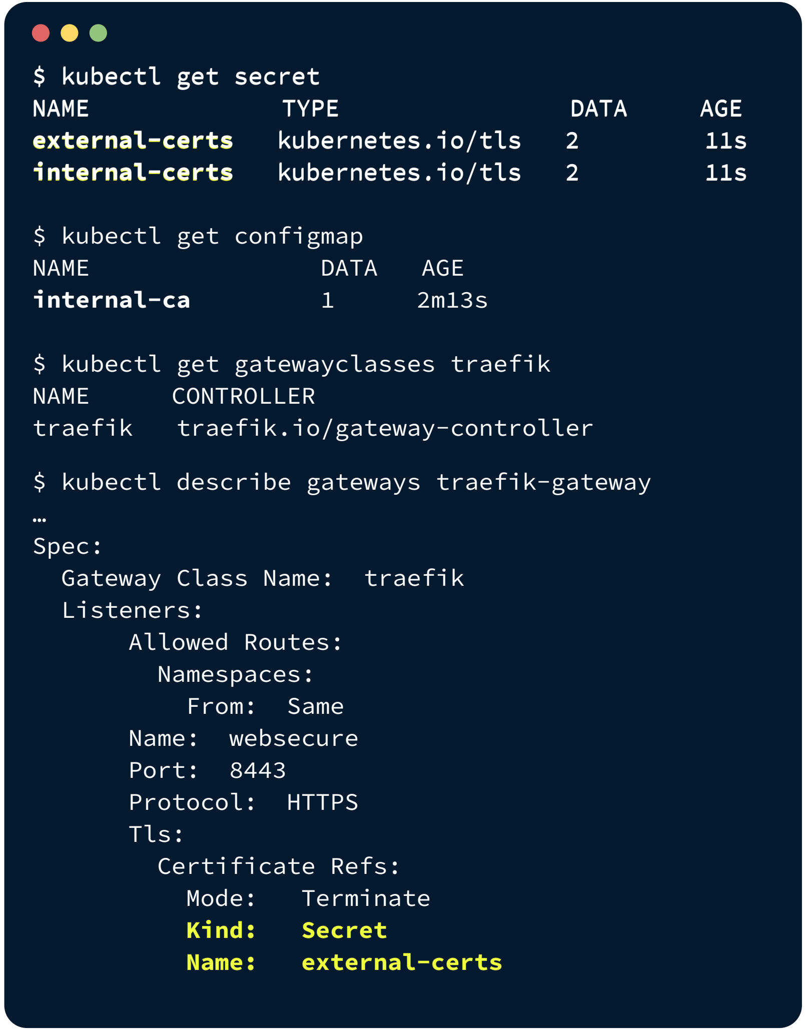
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. **Gateway (with HTTPS Listener)**



```
$ kubectl get secret
NAME             TYPE                 DATA    AGE
external-certs   kubernetes.io/tls   2         11s
internal-certs   kubernetes.io/tls   2         11s

$ kubectl get configmap
NAME                DATA   AGE
internal-ca         1      2m13s

$ kubectl get gatewayclasses traefik
NAME      CONTROLLER
traefik   traefik.io/gateway-controller

$ kubectl describe gateways traefik-gateway
…
Spec:
  Gateway Class Name:  traefik
  Listeners:
      Allowed Routes:
        Namespaces:
          From:  Same
      Name:  websecure
      Port:  8443
      Protocol:  HTTPS
      Tls:
        Certificate Refs:
          Mode:   Terminate
          Kind:   Secret
          Name:   external-certs
```

BackendTLSPolicy Recipe

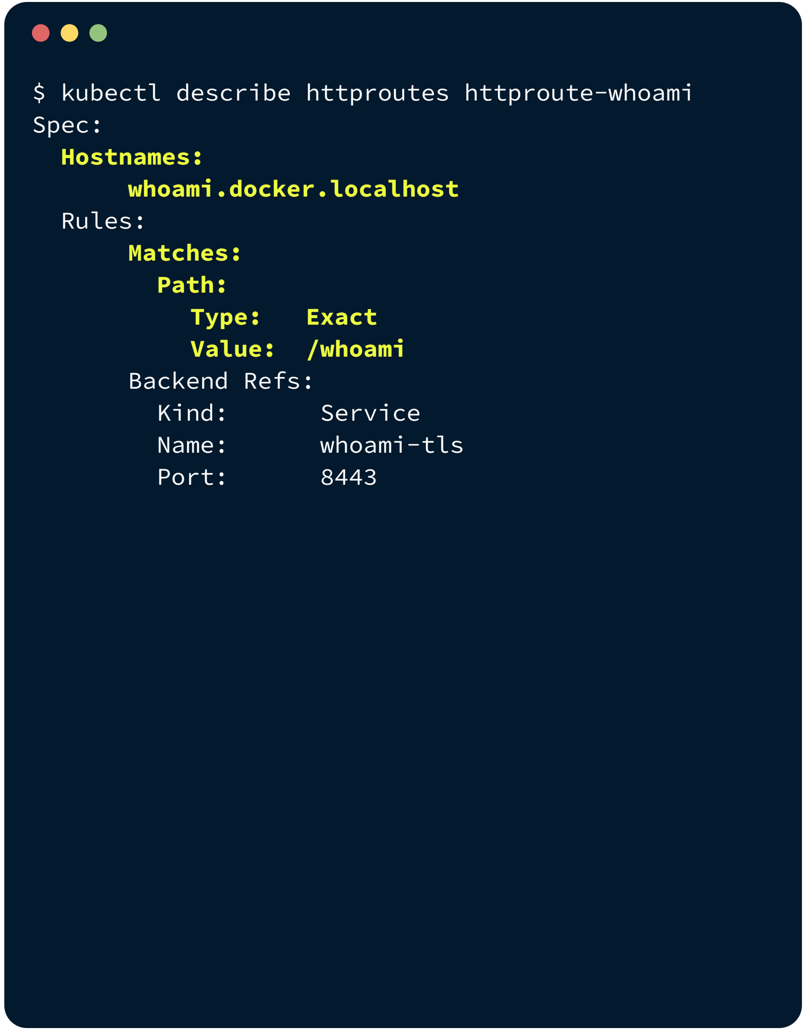
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. **Gateway (with HTTPS Listener)**



```
$ kubectl get secret
NAME              TYPE                 DATA      AGE
external-certs    kubernetes.io/tls    2         11s
internal-certs    kubernetes.io/tls    2         11s

$ kubectl get configmap
NAME                    DATA    AGE
internal-ca             1       2m13s

$ kubectl get gatewayclasses traefik
NAME       CONTROLLER
traefik    traefik.io/gateway-controller

$ kubectl describe gateways traefik-gateway
…
Spec:
  Gateway Class Name:   traefik
  Listeners:
       Allowed Routes:
         Namespaces:
           From:   Same
       Name:   websecure
       Port:   8443
       Protocol:   HTTPS
       Tls:
         Certificate Refs:
           Mode:    Terminate
           Kind:    Secret
           Name:    external-certs
```

BackendTLSPolicy Recipe

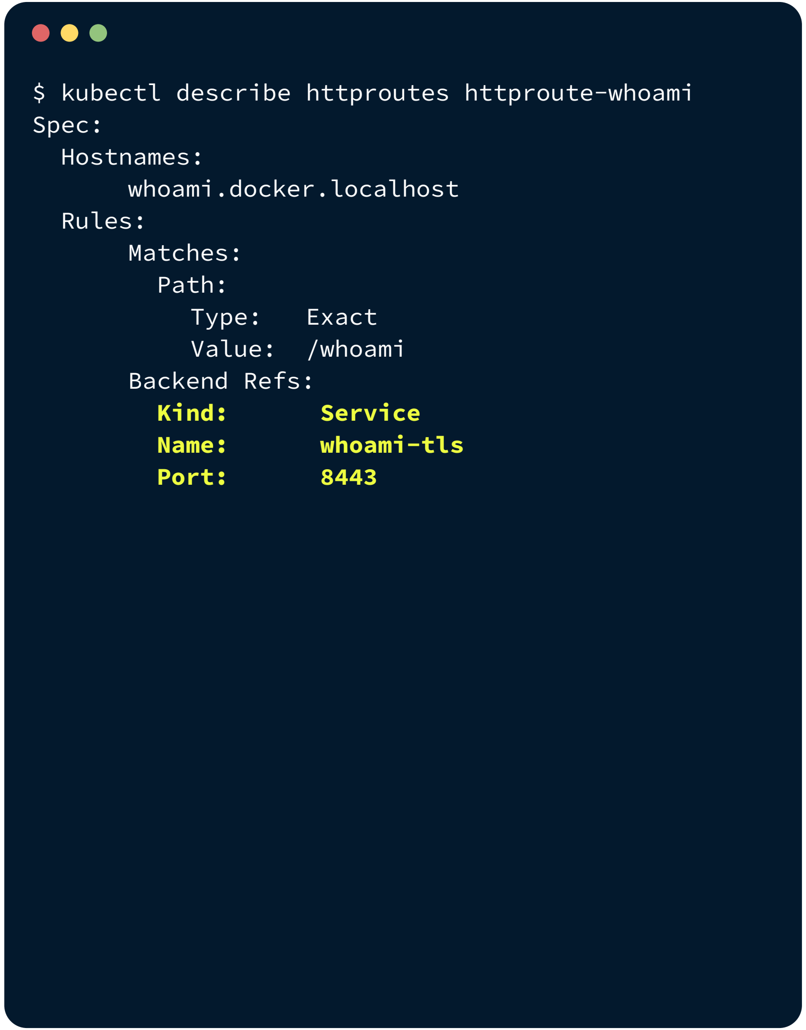
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. **HTTPRoute**
5. TLS Application (with a Service)



```
$ kubectl describe httproutes httproute-whoami
Spec:
  Hostnames:
       whoami.docker.localhost
  Rules:
       Matches:
         Path:
           Type:    Exact
           Value:   /whoami
       Backend Refs:
         Kind:        Service
         Name:        whoami-tls
         Port:        8443
```

BackendTLSPolicy Recipe

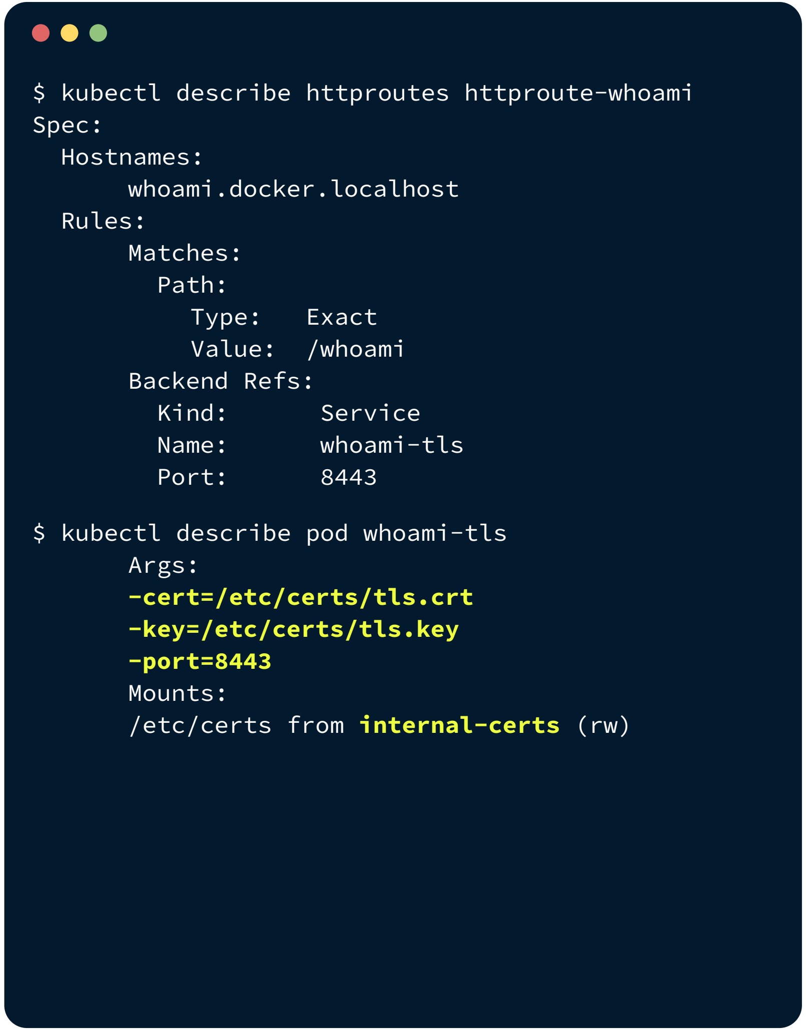
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. **HTTPRoute**
5. TLS Application (with a Service)



```
$ kubectl describe httproutes httproute-whoami
Spec:
  Hostnames:
       whoami.docker.localhost
  Rules:
       Matches:
         Path:
           Type:    Exact
           Value:   /whoami
       Backend Refs:
         Kind:       Service
         Name:       whoami-tls
         Port:       8443
```

BackendTLSPolicy Recipe

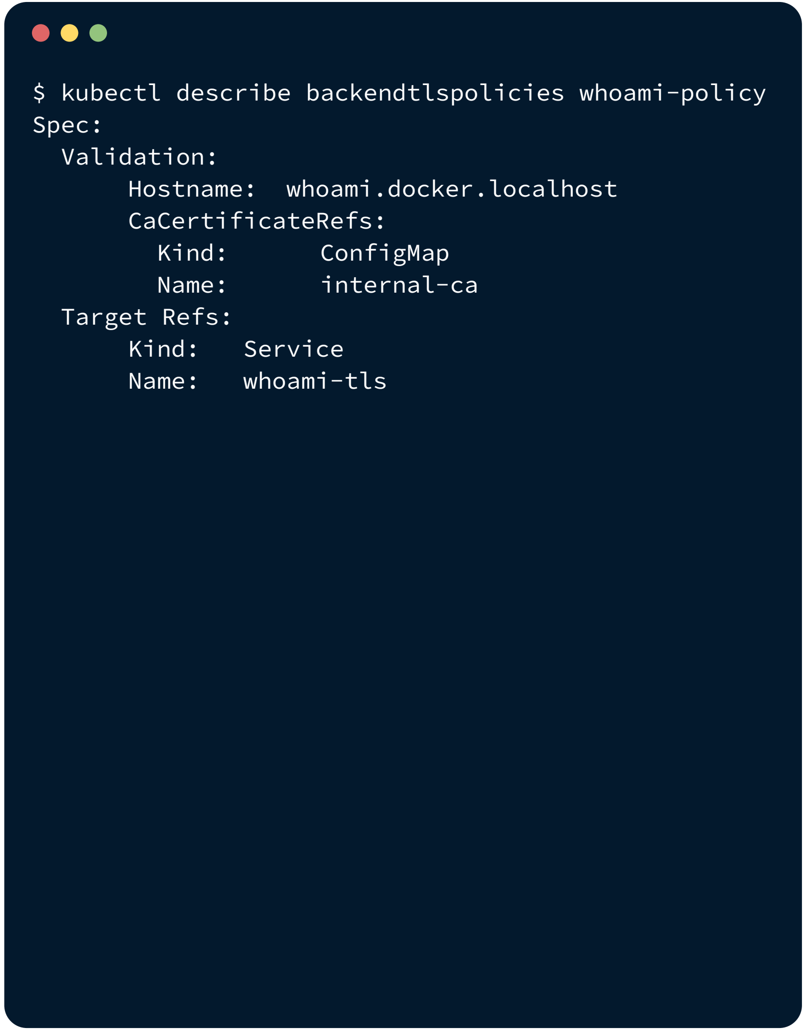
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. HTTPRoute
5. **TLS Application (with a Service)**



```
$ kubectl describe httproutes httproute-whoami
Spec:
  Hostnames:
      whoami.docker.localhost
  Rules:
      Matches:
        Path:
          Type:   Exact
          Value:  /whoami
      Backend Refs:
        Kind:      Service
        Name:      whoami-tls
        Port:      8443

$ kubectl describe pod whoami-tls
      Args:
      -cert=/etc/certs/tls.crt
      -key=/etc/certs/tls.key
      -port=8443
      Mounts:
      /etc/certs from internal-certs (rw)
```

BackendTLSPolicy Recipe

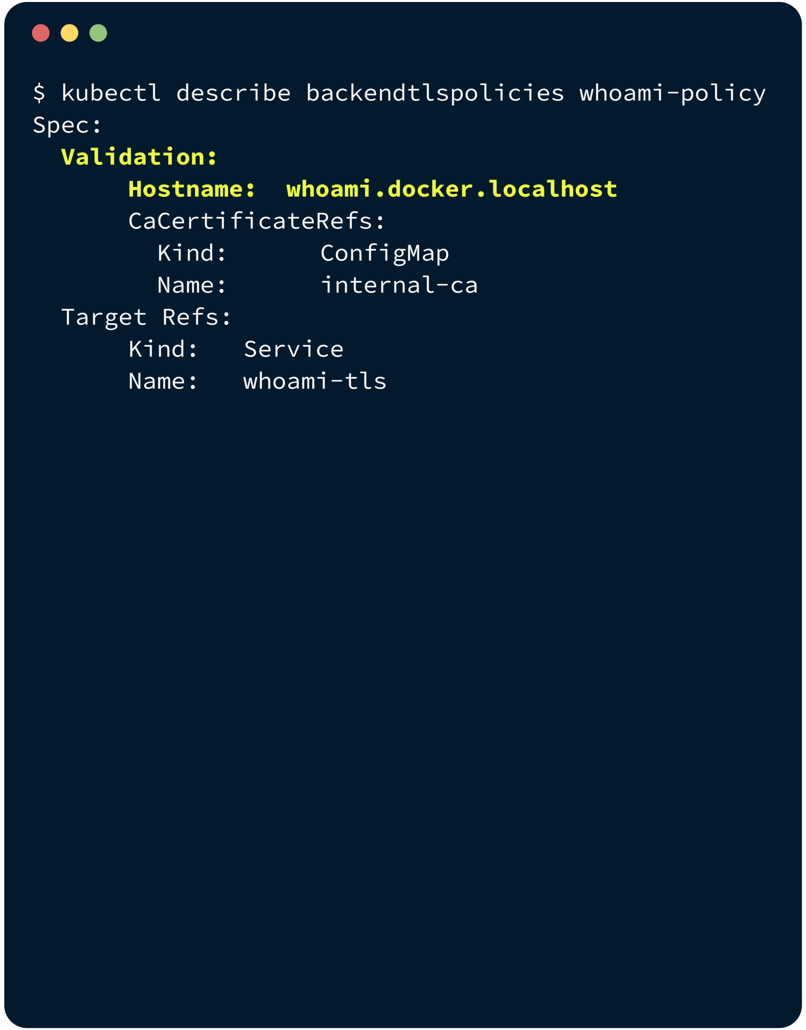
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. HTTPRoute
5. TLS Application (with a Service)
6. **BackendTLSPolicy**
    - Hostname (SNI)
    - CA Root
    - Service / Pod



```
$ kubectl describe backendtlspolicies whoami-policy
Spec:
  Validation:
       Hostname:  whoami.docker.localhost
       CaCertificateRefs:
         Kind:      ConfigMap
         Name:      internal-ca
  Target Refs:
       Kind:   Service
       Name:   whoami-tls
```

BackendTLSPolicy Recipe

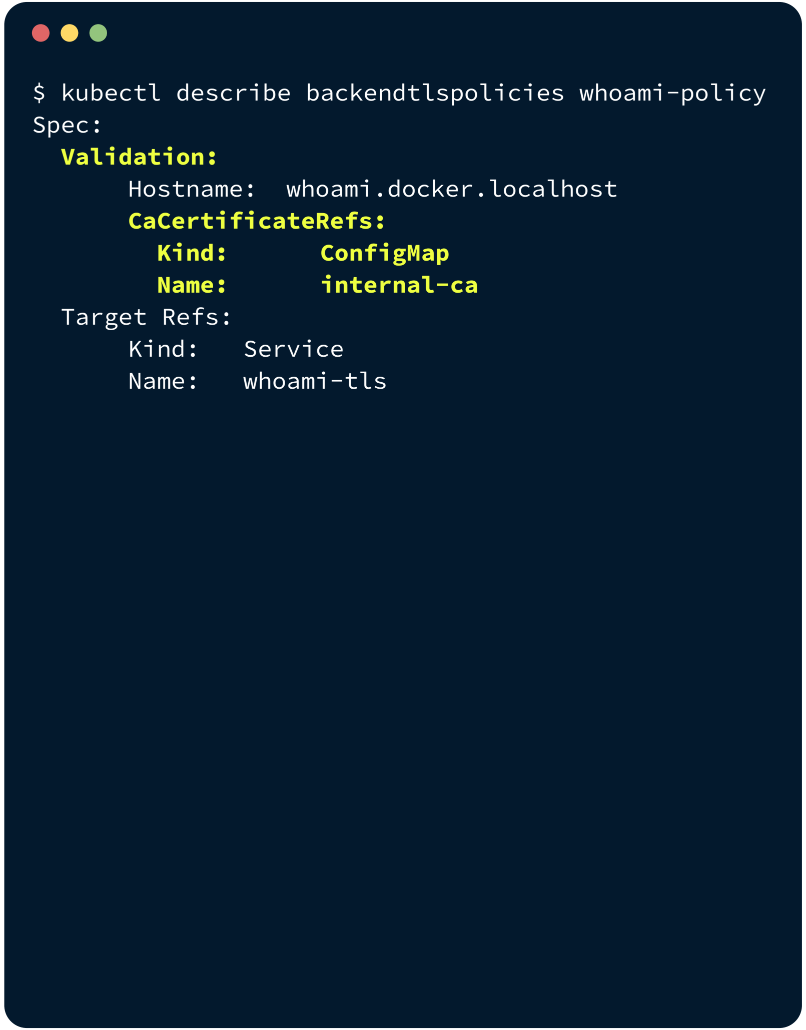
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. HTTPRoute
5. TLS Application (with a Service)
6. **BackendTLSPolicy**
    - **Hostname (SNI)**
    - CA Root
    - Service / Pod



```
$ kubectl describe backendtlspolicies whoami-policy
Spec:
  Validation:
       Hostname:  whoami.docker.localhost
       CaCertificateRefs:
         Kind:       ConfigMap
         Name:       internal-ca
  Target Refs:
       Kind:   Service
       Name:   whoami-tls
```

BackendTLSPolicy Recipe

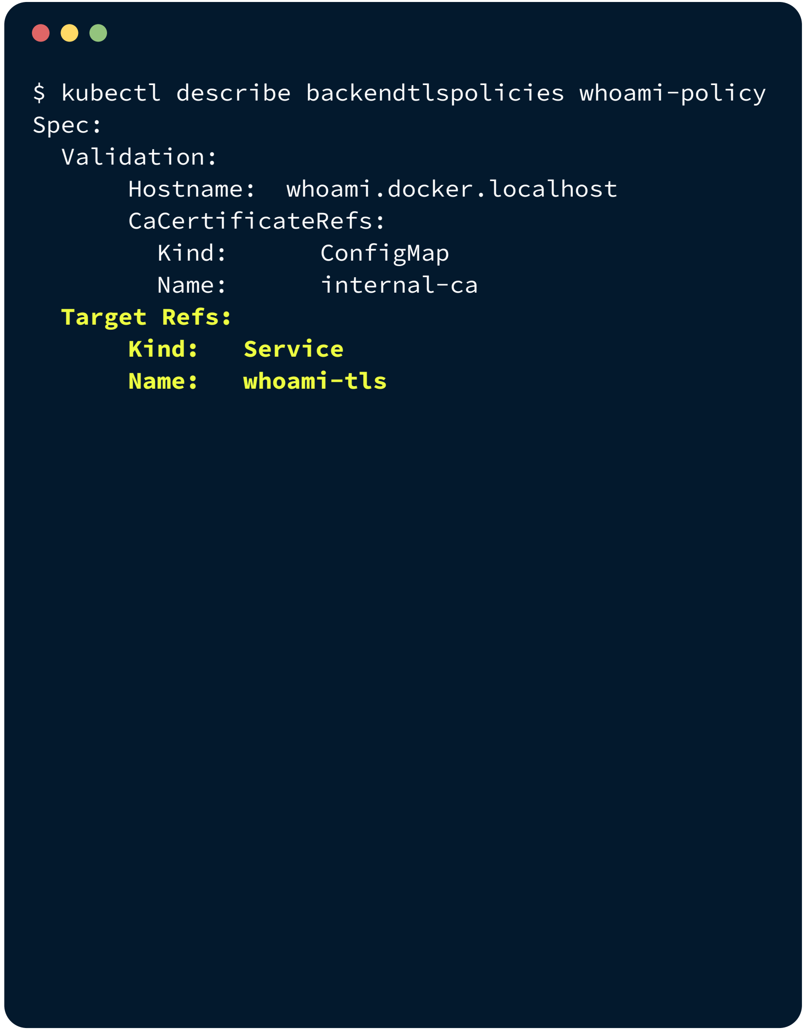
1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. HTTPRoute
5. TLS Application (with a Service)
6. **BackendTLSPolicy**
    - Hostname (SNI)
    - **CA Root**
    - Service / Pod



```
$ kubectl describe backendtlspolicies whoami-policy
Spec:
  Validation:
       Hostname:  whoami.docker.localhost
       CaCertificateRefs:
         Kind:       ConfigMap
         Name:       internal-ca
  Target Refs:
       Kind:    Service
       Name:    whoami-tls
```

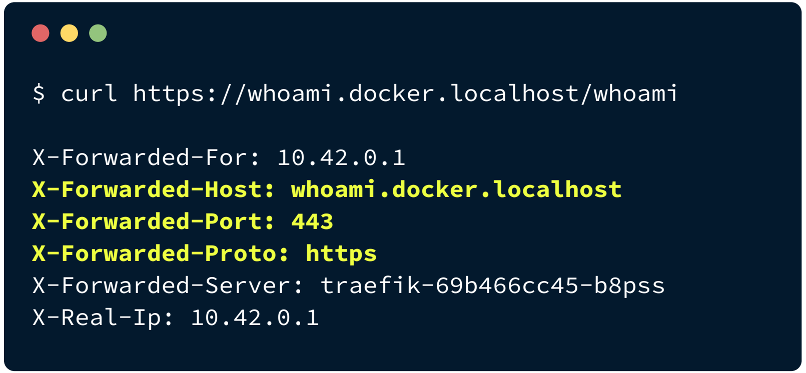
BackendTLSPolicy Recipe

1. TLS certificates + internal CA
2. Gateway Controller + Gateway Class
3. Gateway (with HTTPS Listener)
4. HTTPRoute
5. TLS Application (with a Service)
6. **BackendTLSPolicy**
    - Hostname (SNI)
    - CA Root
    - **Service / Pod**

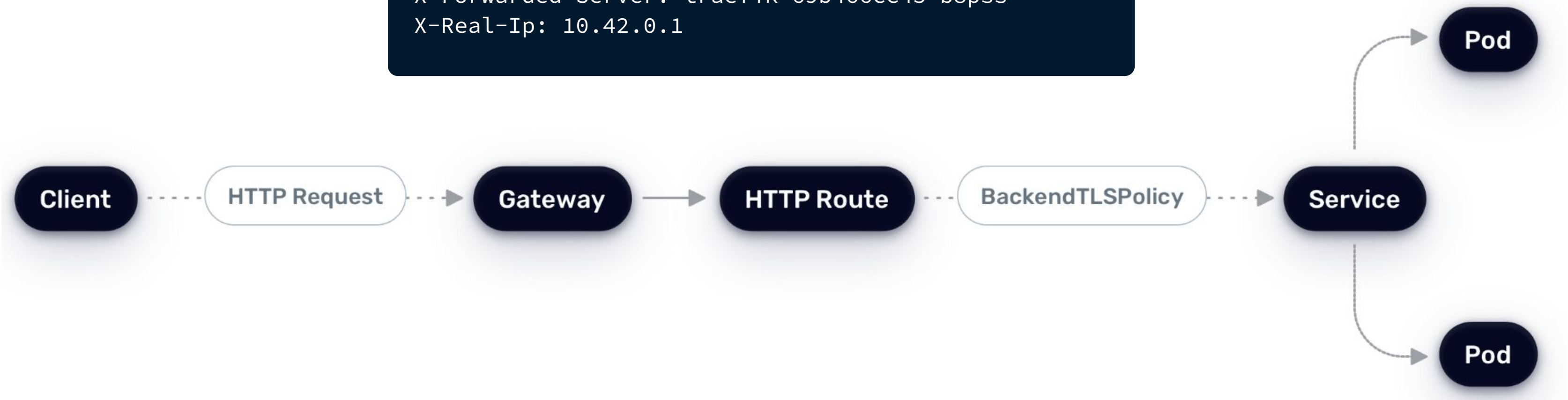


```
$ kubectl describe backendtlspolicies whoami-policy
Spec:
  Validation:
       Hostname:  whoami.docker.localhost
       CaCertificateRefs:
         Kind:       ConfigMap
         Name:       internal-ca
  Target Refs:
       Kind:    Service
       Name:    whoami-tls
```

E2E TLS Connection Ready to Serve!



```
$ curl https://whoami.docker.localhost/whoami

X-Forwarded-For: 10.42.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 443
X-Forwarded-Proto: https
X-Forwarded-Server: traefik-69b466cc45-b8pss
X-Real-Ip: 10.42.0.1
```



Thank you!

nmengin

KubeCon: Booth S650

træfiklabs