



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

Bridging the Gap from Ingress to the Future

Gateway API Maintainer Track Update



Introductions



KubeCon



CloudNativeCon

Europe 2026



Katarzyna Lach
@kl52752



Norwin
Schnyder
@snrowin



Rostislav
Bobrovsky
@rostislavbobo



James Strong
@strongjz



Nick Young
@youngnick

Agenda



KubeCon



CloudNativeCon

Europe 2026

- Ingress-nginx deprecation and archival
- How Gateway is pushing to make ingress-nginx migration easier
- Release 1.5 Update
- Controller Matching Wizard



KubeCon



CloudNativeCon

Europe 2026

Ingress-nginx Deprecation And Archival



Ingres-nginx by the numbers



KubeCon



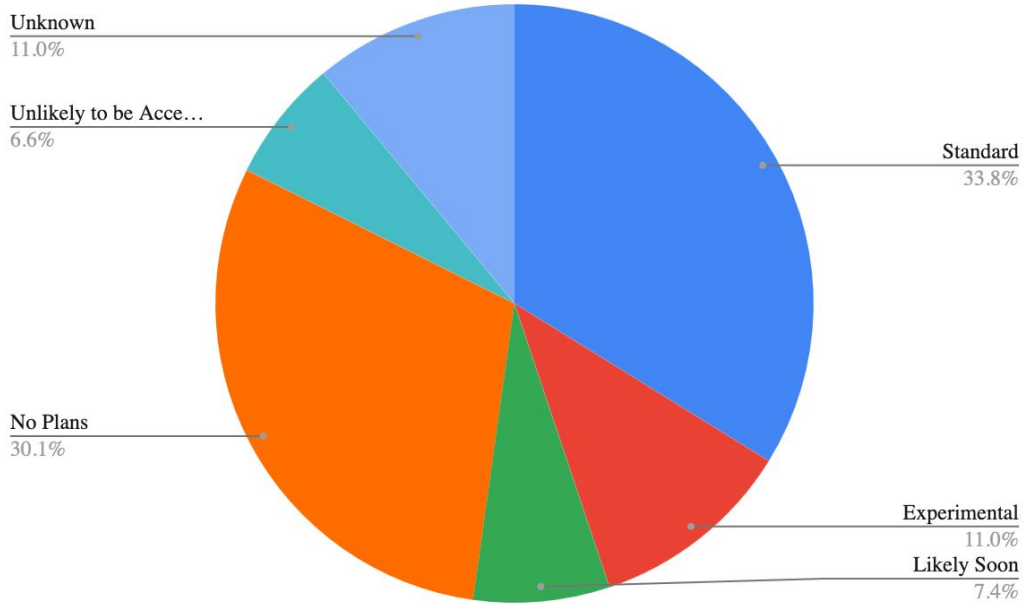
CloudNativeCon

Europe 2026

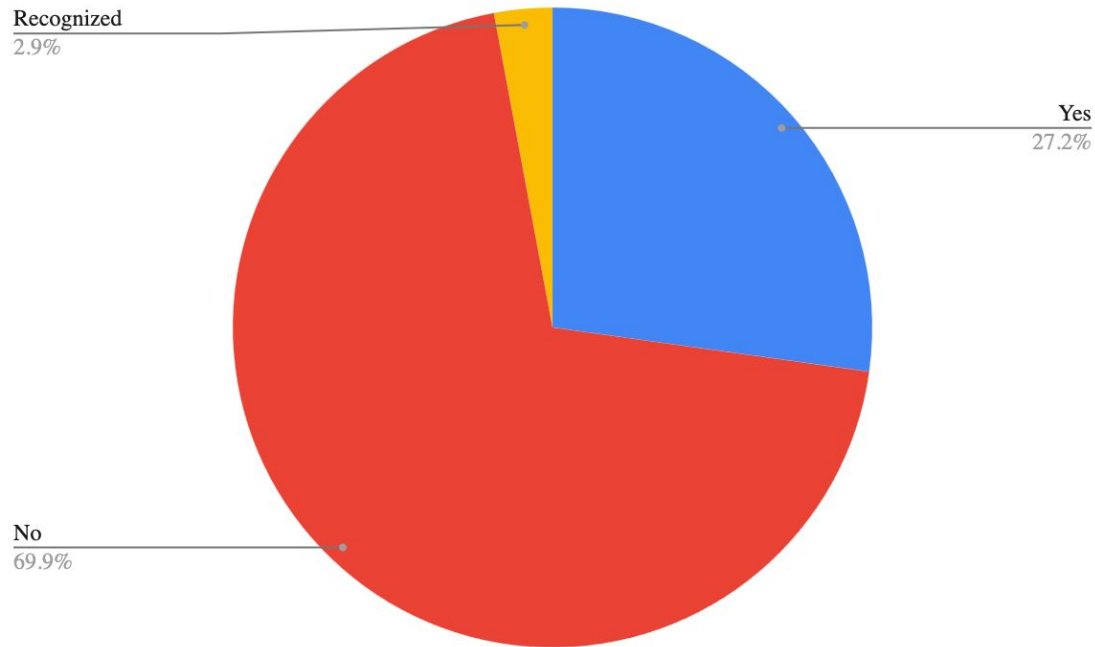
- 8 years old
- 275 Releases
- 1,099 Contributors
- 5,761 Closed Issues
- 8,432 Commits
- 8,500 Forks
- 8,583 Pull Requests
- 19,500 Stars

- Final Releases - [1.15.1](#), [1.14.5](#), [1.13.9](#)
- Kubernetes SRC will remain the CVE Numbering Authority of scope for issuing CVEs
- Continue to work with ingress2gateway project
- Security
 - No patch releases for any vulnerabilities
 - No CVEs detected in dependencies or release artifacts
- Project will be archived after Kubecon EU 26
- Shout out to Steven Jin

Ingress-nginx Feature Set



Ingress2Gateway Ingress-nginx Feature Set



<https://github.com/kubernetes-sigs/ingress2gateway/tree/main/pkg/i2gw/providers/ingressnginx>



KubeCon



CloudNativeCon

Europe 2026

One last thing before I go







KubeCon



CloudNativeCon

Europe 2026

What's Gateway API doing to help?

What's Gateway API doing to help?



KubeCon



CloudNativeCon

Europe 2026

Gateway API has two main ways that we are trying to help users:

- ingress2gateway
- Prioritizing features that help with Ingress migration (like ListenerSet, but there are a lot of others as well)

Tool to programmatically migrate Ingress (or other) resources to Gateway API ones.

Has specific ingress-nginx support, including many annotations, and converts that to equivalent Gateway API config.

Also supports many other sources, both specific Ingress annotations, and other implementation-specific resources.

Version 1.0 has recently been released!

- ingress-nginx specific annotation support
- New emitters for outputting provider-specific Gateway API extensions



kubernetes-sigs/ingress2gateway



KubeCon



CloudNativeCon

Europe 2026

Release 1.5 Update

Release process update



KubeCon



CloudNativeCon

Europe 2026

- Moved to a release-train style model
- Monthly tagged releases for development (**monthly-26.01**)
- Regular semver releases on designated dates
 - These include whatever is ready at the time
- V1.5 is the first release to use this process
 - 5 features moved to Standard (the most ever!)

Features moved to Standard in 1.5



KubeCon



CloudNativeCon

Europe 2026

- ListenerSet support ([GEP-1713](#))
- TLSRoute ([GEP-2643](#))
- Gateway Client Certificate validation ([GEP-91](#), [GEP-3567](#))
- Certificate selection for Gateway TLS origination ([GEP-3155](#))
- HTTPRoute CORS filter ([GEP-1767](#))



KubeCon



CloudNativeCon

Europe 2026

ListenerSet



ListenerSet is a new resource that allows breaking the Listener part out of Gateways, in a separate object.

This helps with some important usecases:

- Large numbers of Listeners on a single Gateway (>64)
- Moving ownership of TLS away from Gateway owners

Already supported by 6 implementations.



KubeCon



CloudNativeCon

Europe 2026

TLS Route



From HTTP-based Routing

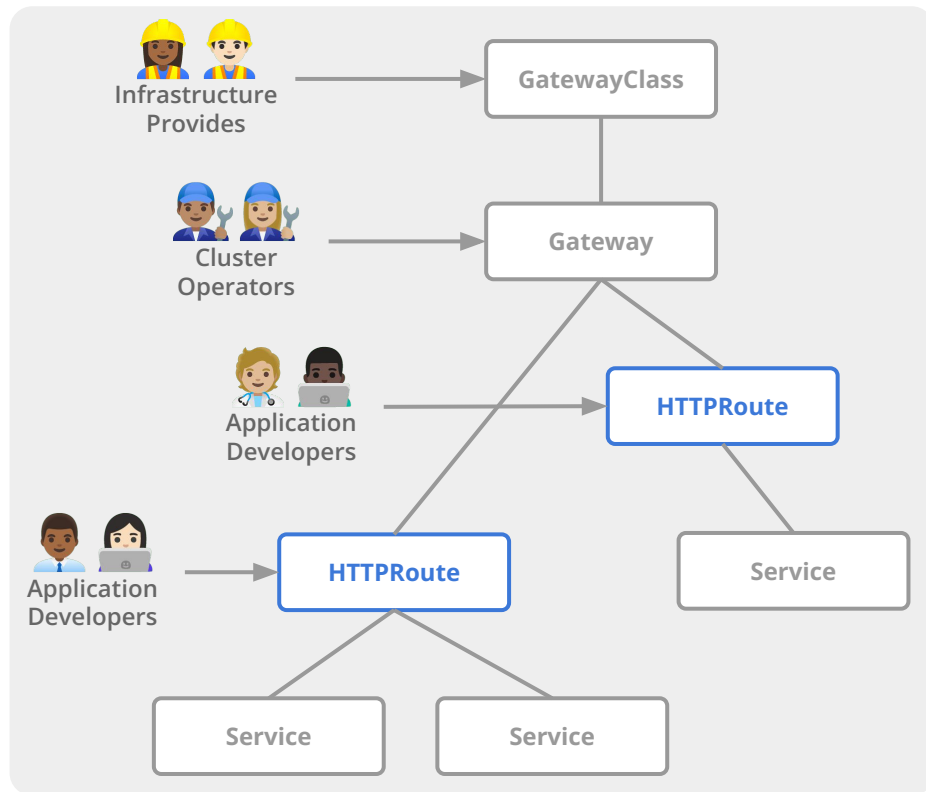


KubeCon



CloudNativeCon

Europe 2026



To SNI-based Routing



KubeCon

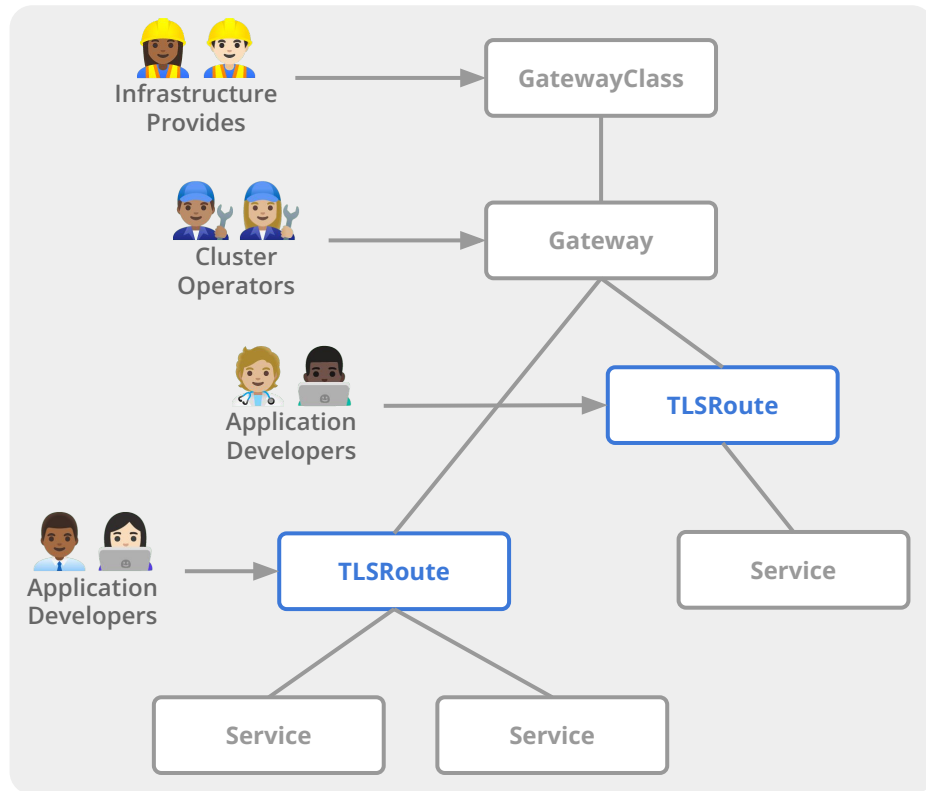


CloudNativeCon

Europe 2026

TLSRoute allows multiplexing TLS connections by matching the Server Name Indication (**SNI**) presented during the TLS handshake.

Already supported by 6 implementations, with more coming later this year.



To SNI-based Routing



KubeCon



CloudNativeCon

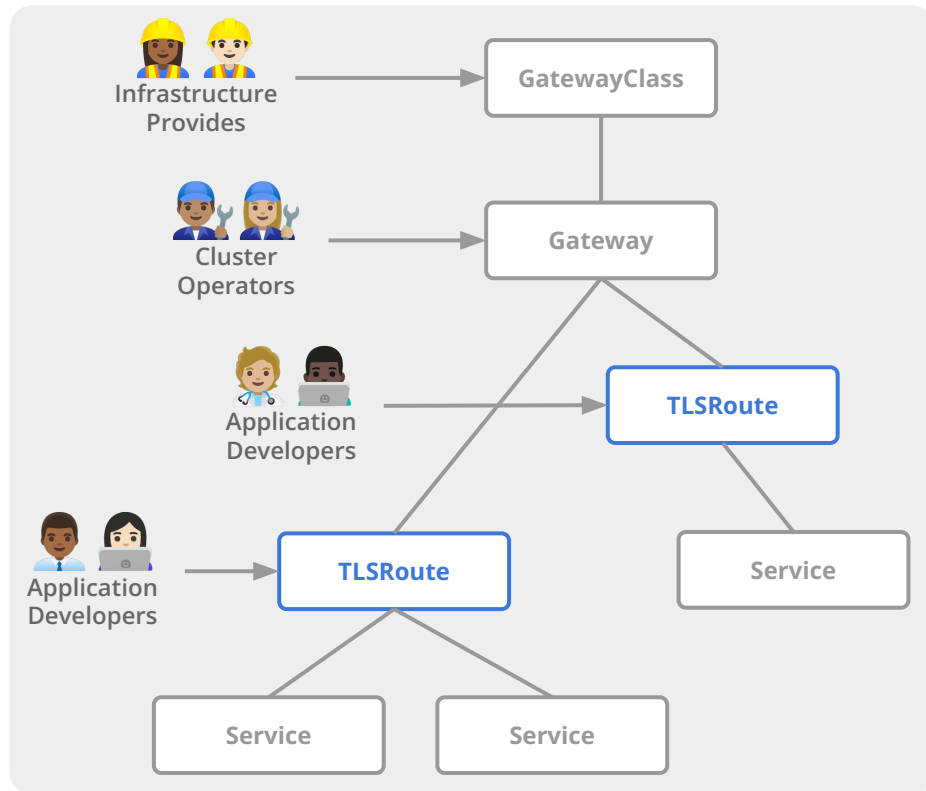
Europe 2026

TLSRoute allows multiplexing TLS connections by matching the Server Name Indication (**SNI**) presented during the TLS handshake.

Gateway TLS Listener Modes:

- Terminate
- Passthrough

Already supported by 6 implementations, with more coming later this year.



Terminate mode



KubeCon



CloudNativeCon

Europe 2026

Terminate mode



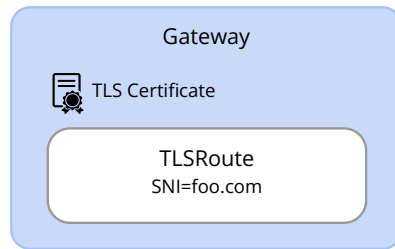
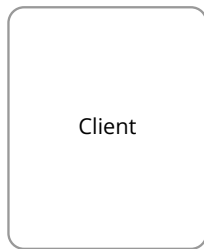
KubeCon



CloudNativeCon

Europe 2026

For the convenience of
**centralized TLS certificate
management** at the Gateway
edge for non-HTTP protocols.



TCP terminated, TLS terminated



Terminate mode



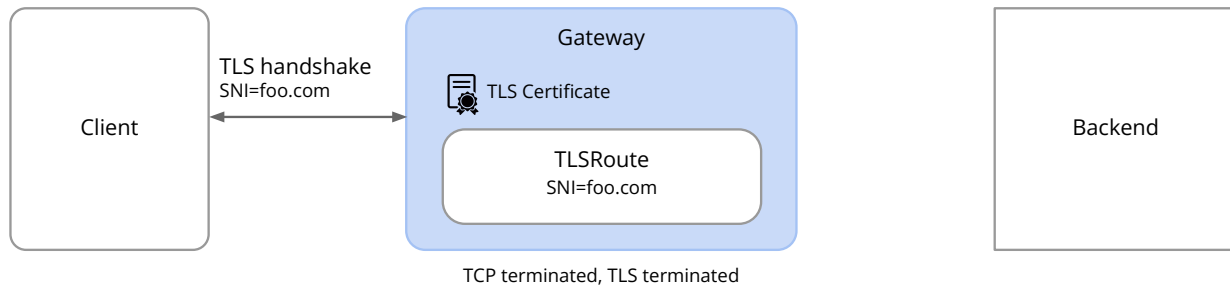
KubeCon



CloudNativeCon

Europe 2026

For the convenience of **centralized TLS certificate management** at the Gateway edge for non-HTTP protocols.



Terminate mode



KubeCon

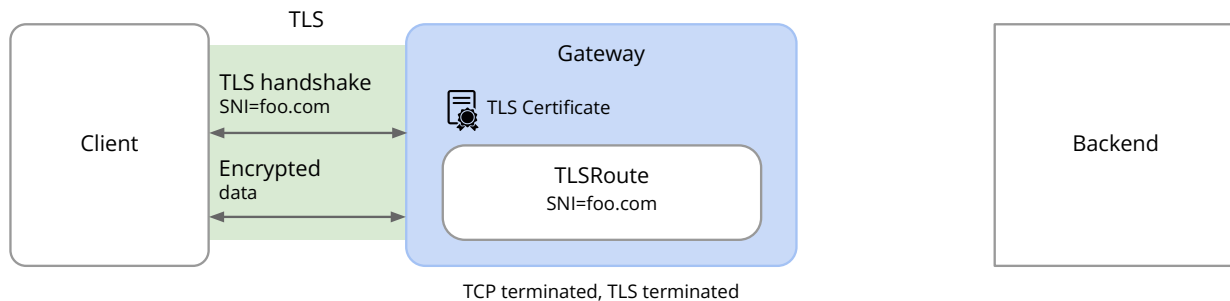


CloudNativeCon

Europe 2026

For the convenience of **centralized TLS certificate management** at the Gateway edge for non-HTTP protocols.

1. The TLS session is terminated at the Gateway.



Terminate mode



KubeCon

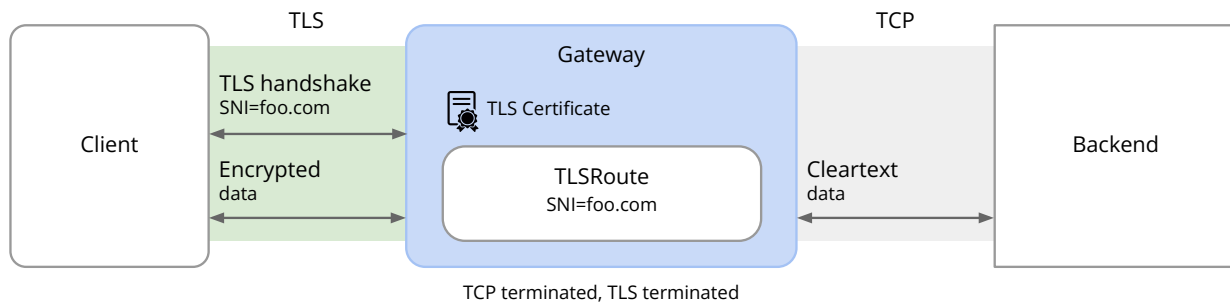


CloudNativeCon

Europe 2026

For the convenience of **centralized TLS certificate management** at the Gateway edge for non-HTTP protocols.

1. The TLS session is terminated at the Gateway.
2. The Gateway routes the decrypted payload as a raw TCP stream.



Passthrough mode



KubeCon



CloudNativeCon

Europe 2026

Passthrough mode



KubeCon



CloudNativeCon

Europe 2026

For strict security compliance,
end-to-end (m)TLS, or when
you cannot store certificates
on the proxy.

Passthrough mode



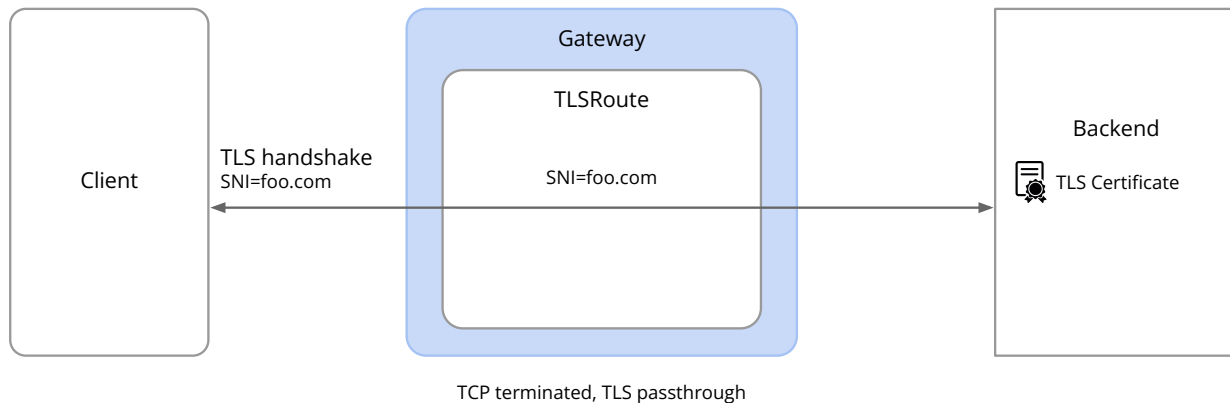
KubeCon



CloudNativeCon

Europe 2026

For strict security compliance,
end-to-end (m)TLS, or when
you cannot store certificates
on the proxy.



Passthrough mode



KubeCon

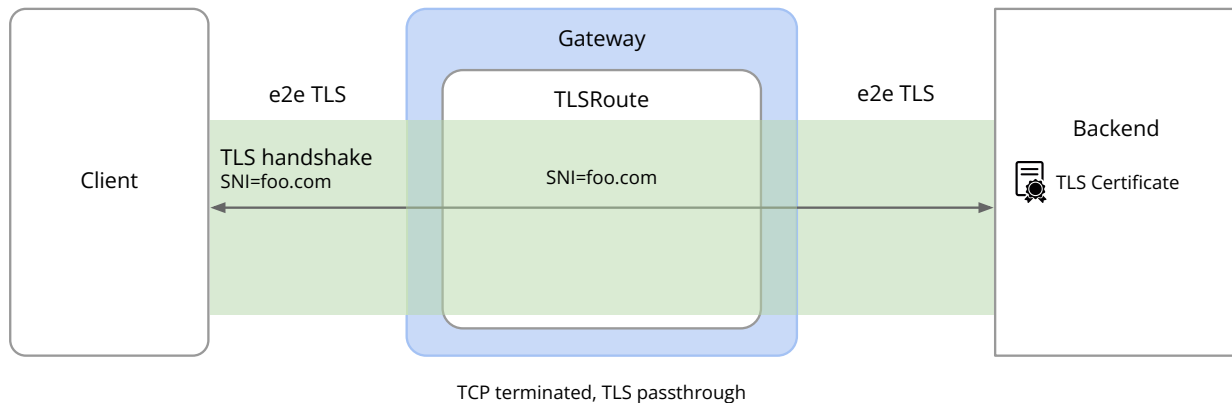


CloudNativeCon

Europe 2026

For strict security compliance,
end-to-end (m)TLS, or when
you cannot store certificates
on the proxy.

1. The encrypted byte stream is proxied directly to the destination backend.



Passthrough mode



KubeCon

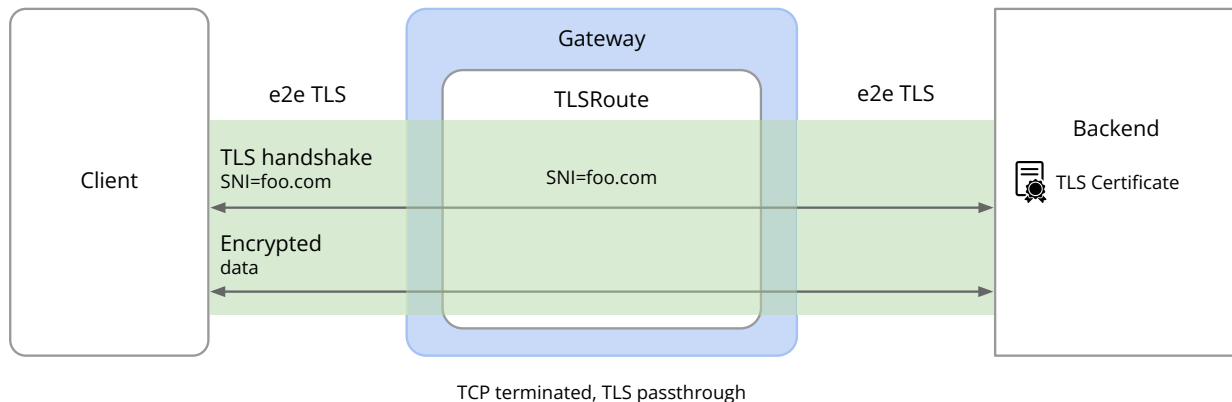


CloudNativeCon

Europe 2026

For strict security compliance,
end-to-end (m)TLS, or when
you cannot store certificates
on the proxy.

1. The encrypted byte stream is proxied directly to the destination backend.
2. The Gateway has zero access to private keys or unencrypted data



Passthrough mode



KubeCon

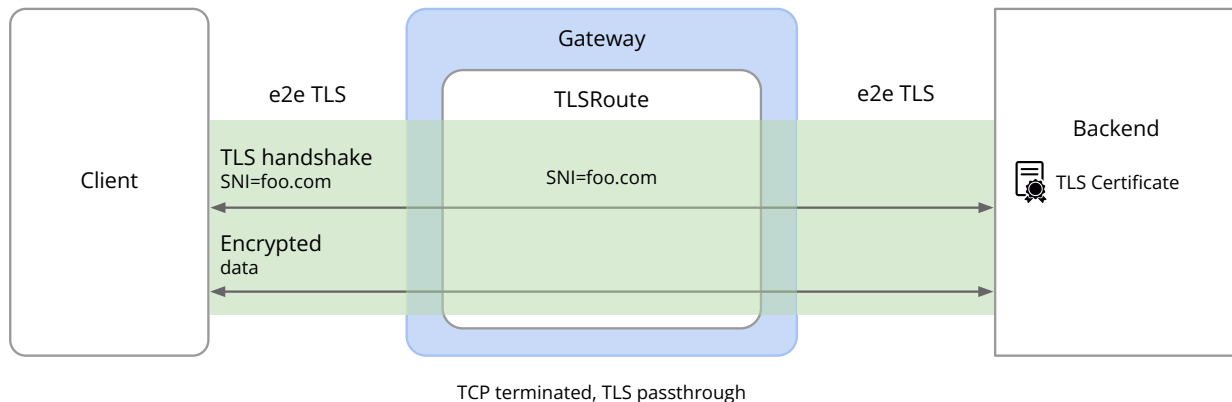


CloudNativeCon

Europe 2026

For strict security compliance, **end-to-end (m)TLS**, or when you cannot store certificates on the proxy.

1. The encrypted byte stream is proxied directly to the destination backend.
2. The Gateway has zero access to private keys or unencrypted data



With HTTPS, wildcard TLS certificates can cause misrouting due to HTTP connection coalescing.



KubeCon



CloudNativeCon

Europe 2026

New mTLS features



Terminated TLS



KubeCon



CloudNativeCon

Europe 2026

Frontend (m)TLS



Customer

CA cert

Client cert

1. Initiate TLS handshake

2. Present **Server** certificate

3. Validate **Server** certificate with CA

4. Present **Client** certificate

5. Validate **Client** certificate with CA

6. Established the connection

Terminate the connection



Gateway

Server cert

Backend CA cert

Frontend CA cert

Client cert



1. Initiate TLS handshake

2. Present **Server** certificate

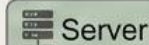
3. Validate **Server** certificate with CA

4. Present **Client** certificate

5. Validate **Client** certificate with CA

6. Established the connection

Backend (m)TLS



Backend/Pod

Server cert

CA cert

Client certificate validation



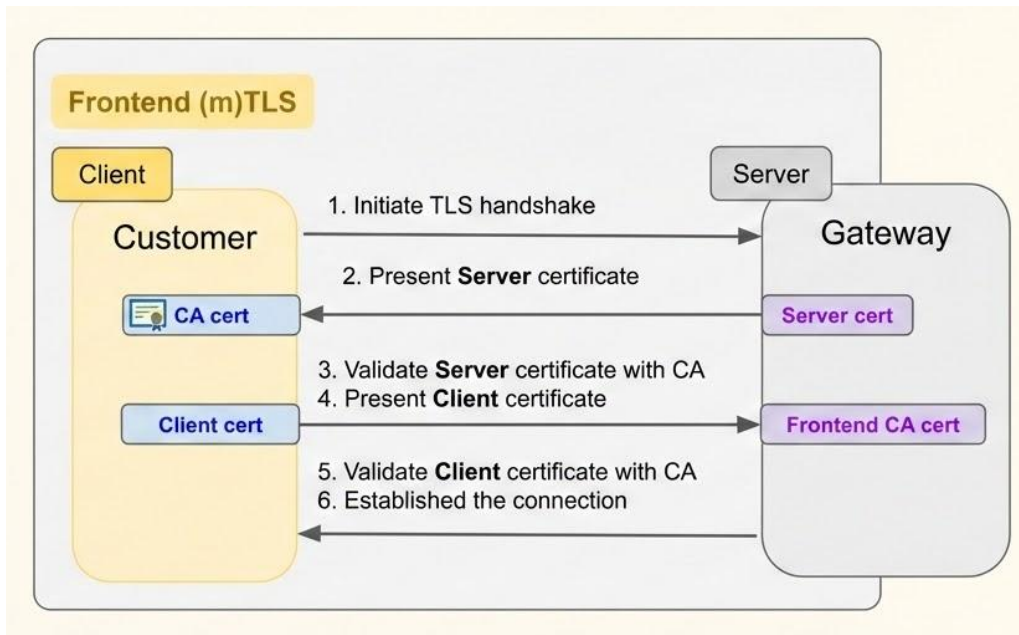
KubeCon



CloudNativeCon

Europe 2026

1. Validate Client certificate against a set of trusted Certificate Authorities (CAs)
2. Flexible configuration as default for Gateway or per port overrides
3. Support mode for Strict or Insecure validation
4. In Standard from v1.5 release



TLS Overview



KubeCon

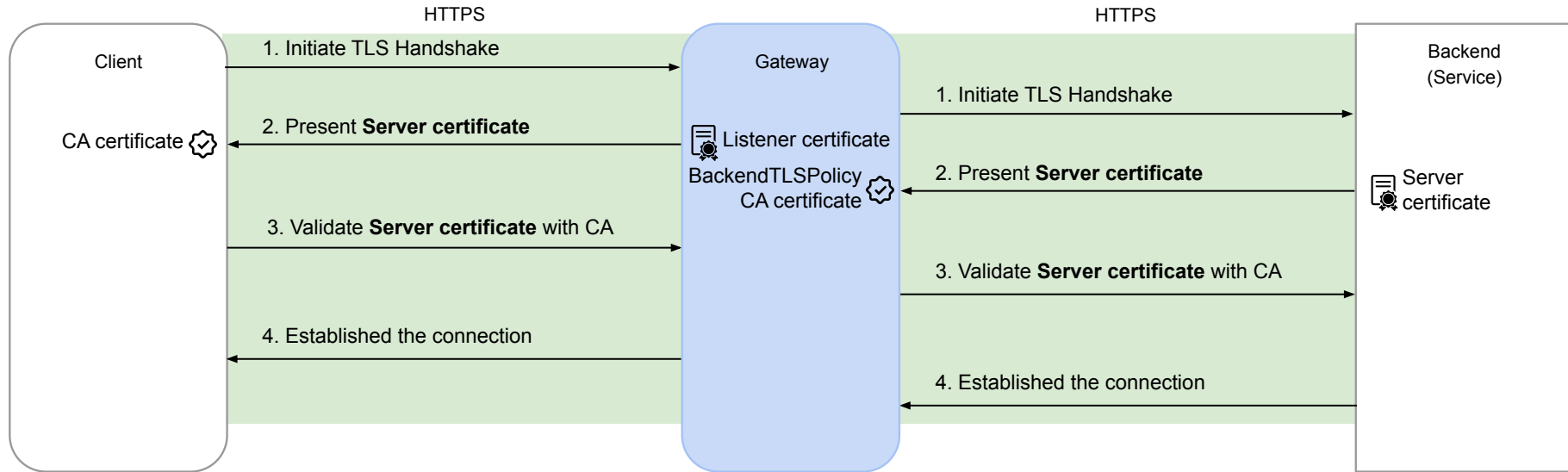


CloudNativeCon

Europe 2026

Frontend TLS

Backend TLS



Mutual TLS (mTLS) Overview



KubeCon

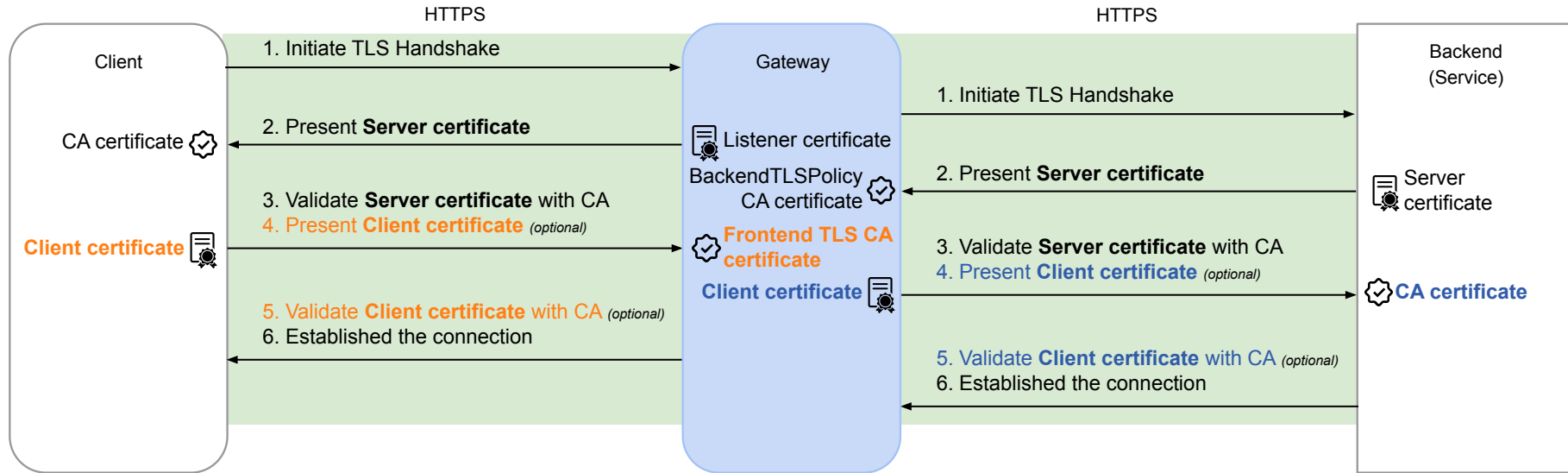


CloudNativeCon

Europe 2026

Frontend mTLS

Backend mTLS



Client certificate validation



KubeCon

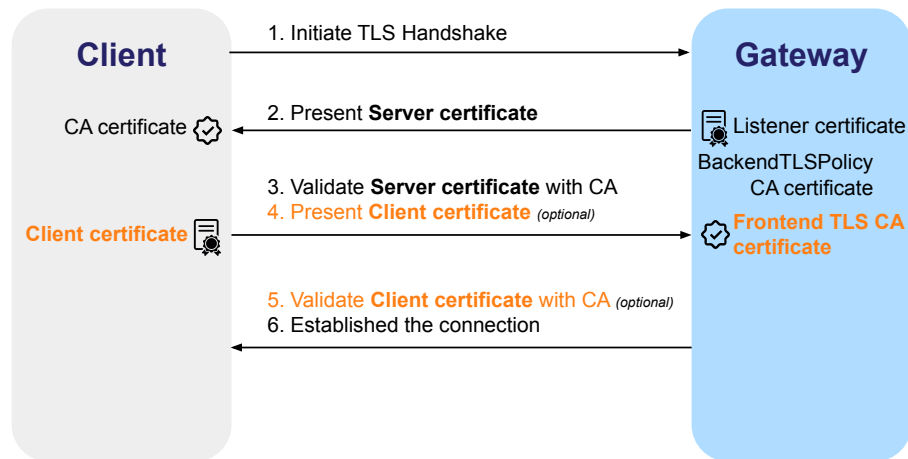


CloudNativeCon

Europe 2026

- Validate Client certificate against a set of trusted **Certificate Authorities (CAs)**
- Flexible configuration as default for Gateway or per port overrides
- Support mode for Strict or Insecure validation
- In Standard from v1.5 release

Frontend mTLS



HTTP2 Connection coalescing



KubeCon



CloudNativeCon

Europe 2026

- **Connection coalescing** is a performance feature introduced in HTTP2
- Clients are free to reuse its TLS connection for requests sent to any domains in the Gateway that are using the same IP and Port when they have matching certificates
- This might lead to bypassing TLS configuration if it is bound to the listener

```
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: wildcard-tls-gateway
spec:
  gatewayClassName: example
  listeners:
  - name: foo-https
    protocol: HTTPS
    port: 443
    hostname: foo.example.com
    tls:
      certificateRefs:
      - kind: Secret
        group: ""
        name: foo-example-com-cert
  - name: wildcard-https
    protocol: HTTPS
    port: 443
    hostname: "*.example.com"
    tls:
      certificateRefs:
      - kind: Secret
        group: ""
        name: wildcard-example-com-cert
```

Client certificate validation API



KubeCon



CloudNativeCon

Europe 2026

- By default Client certificate configuration is defined per Gateway and applies for all HTTPS listeners (`tls.frontend.default`)
- To introduce finer granularity per-port override can be configured (`tls.frontend.perPort`)
- CA Certificates are configured with a list of ConfigMaps containing PEM encoded CA certificates
- TLS Configuration supports `AllowValidOnly` and `AllowInsecureFallback` modes

```
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: client-validation-basic
spec:
  gatewayClassName: acme-lb
  tls:
    frontend:
      default:
        validation:
          caCertificateRefs:
            - kind: ConfigMap
              group: ""
              name: foo-example-com-ca-cert
      perPort:
        - port: 8443
          tls:
            validation:
              caCertificateRefs:
                - kind: ConfigMap
                  group: ""
                  name: bar-example-com-ca-cert
    listeners:
      - name: foo-https
        protocol: HTTPS
        port: 443
        hostname: foo.example.com
        tls:
          certificateRefs:
            - kind: Secret
              group: ""
              name: foo-example-com-cert
      - name: bar-https
        protocol: HTTPS
        port: 8443
        hostname: bar.example.com
        tls:
          certificateRefs:
            - kind: Secret
              group: ""
              name: bar-example-com-cert
```

Gateway's client certificate selection



KubeCon

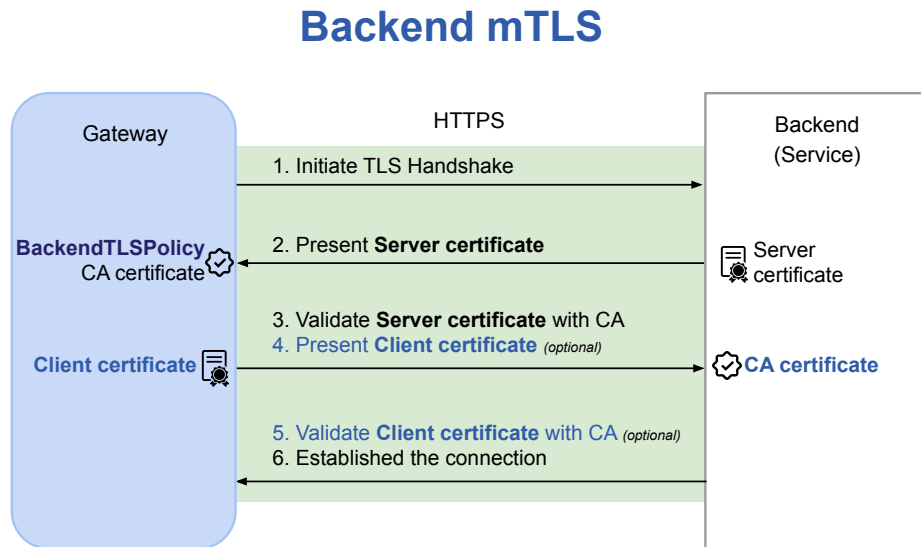


CloudNativeCon

Europe 2026

To enable TLS to a backend, a **BackendTLSPolicy** attached to a backend Service via a target reference must be configured.

Mutual TLS (mTLS) for backend connections requires the Gateway to present a **client certificate** to the backend.



Gateway's client certificate selection API



KubeCon



CloudNativeCon

Europe 2026

During the TLS handshake, the Gateway uses the TLS client certificate specified in:

`tls.backend.clientCertificateRef`

Note: This configuration defines the *Gateway's identity* and is applied to *all* TLS connections originating from that Gateway.

```
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: backend-tls
spec:
  gatewayClassName: acme-lb
  tls:
    backend:
      clientCertificateRef:
        kind: Secret
        group: ""
        name: foo-example-cert
  listeners:
  - name: foo-http
    protocol: HTTP
    port: 80
    hostname: foo.example.com
```



KubeCon



CloudNativeCon

Europe 2026

HTTPRoute CORS filter



HTTPRoute CORS filter API



KubeCon



CloudNativeCon

Europe 2026

Cross-Origin Resource Sharing (CORS) is an HTTP header based security mechanism that allows a server to indicate any origins other than its own from which a **browser** should permit loading resources.

CORS also relies on a mechanism by which browsers make a **preflight** request to the server to check that the server will permit the actual request.

The **CORS** filter in an HTTPRouteRule (`rules[].filters`) can be used to specify the CORS policy.

Policy ≠ HTTP Headers

Policies may include wildcards (*), which might be dynamically translated into the appropriate HTTP response header.

```
apiVersion: gateway.networking.k8s.io/v1
kind: HTTPRoute
metadata:
  name: example-cors-route
spec:
  parentRefs:
  - name: example-gateway
  rules:
  - backendRefs:
    - name: example-svc
      port: 8080
    filters:
    - cors:
        allowOrigins:
        - "https://www.foo.com"
        - "https://*.bar.com"
        allowMethods:
        - PUT
        - POST
        allowHeaders:
        - "x-header-1"
        - "x-header-2"
        exposeHeaders:
        - "x-header-3"
        allowCredentials: false
        maxAge: 3600
      type: CORS
```



KubeCon



CloudNativeCon

Europe 2026

Controller Matching Wizard

<https://gateway-api.sigs.k8s.io/implementations/wizard/>



Overview

Introduction

Concepts

API Overview

Troubleshooting and Status

Conformance

Roles and Personas

Security

Tools

Use Cases

Versioning

Traffic Matching

Hostnames

Service Mesh

Overview

GAMMA Initiative

Service Facets

Implementations

List

Matching Wizard

Comparisons

v1.5

v1.4

Introduction

Gateway API is an official Kubernetes project focused on L4 and L7 routing in Kubernetes. This project represents the next generation of Kubernetes Ingress, Load Balancing, and Service Mesh APIs. From the outset, it has been designed to be generic, expressive, and role-oriented.

The overall resource model focuses on 3 separate [personas](#) and corresponding resources that they are expected to manage:

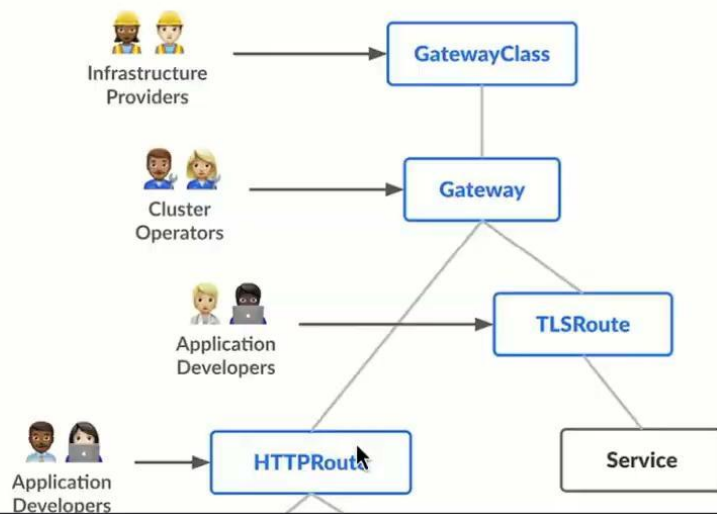


Table of contents

Gateway API for Ingress

Gateway API for Service Mesh (the GAMMA initiative)

Getting started

Gateway API concepts

Why does a role-oriented API matter?

Personas

Use Cases

What's the difference between Gateway API and an API Gateway?

Who is working on Gateway API?



Overview

- API Overview
- Troubleshooting and Status
- Conformance
- Roles and Personas
- Security
- Tools
- Use Cases
- Versioning
- Traffic Matching
- Hostnames
- Service Mesh
 - Overview
- GAMMA Initiative
- Service Facets
- Implementations
 - List
 - [Matching Wizard](#)
- Comparisons
 - v1.5
 - v1.4
 - v1.3

Matching controllers

Organization	Project	Conformance	Must have	Nice to have	Missing
agentgateway	agentgateway v1.0.0	Core, GRPCRoute, TLSRoute	2/2	5/5	—
kgateway-dev	kgateway v2.3.0-beta.3	Core, GRPCRoute, TLSRoute	2/2	4/5	Gateway Backend Client Certificate
airlock	microgateway 5.0.0	Core	2/2	3/5	ListenerSet, [TLSRoute] Core Features
nginx	nginx-gateway-fabric v2.5.0-pre-release	Core, GRPCRoute, TLSRoute	2/2	3/5	Gateway Frontend Client Certificate Validation, ListenerSet
traefik	traefik v3.7	Core, GRPCRoute, TLSRoute	2/2	1/5	Gateway Backend Client Certificate, Gateway Frontend Client Certificate Validation, ListenerSet, HTTPRoute CORS
HAProxy Ingress	haproxy-ingress v0.17.0-alpha.1	Core	2/2	1/5	Gateway Backend Client Certificate, Gateway Frontend Client Certificate Validation, ListenerSet, [TLSRoute] Core Features

Get Involved



KubeCon



CloudNativeCon

Europe 2026

- Tons of opportunities to get involved
 - ingress2gateway
 - gwctl
 - docs
 - choose your own

gateway-api.sigs.k8s.io



#sig-network-gateway-api



kubernetes-sigs/gateway-api



KubeCon



CloudNativeCon

Europe 2026

Thanks!

