Kubernetes Gateway APIs

John Howard / @howardjohn / Google

Overview

- Introduction and motivations for Kubernetes Gateway APIs
- Istio's plans for Kubernetes Gateway APIs

Current state of Networking APIs

Ingress

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: hello-world
spec:
  rules:
  - http:
      paths:
      - path: /hello
        pathType: Prefix
        backend:
          service:
            name: hello-word
```


Current state of Networking APIs

Ingress

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: hello-world
spec:
  rules:
  - http:
      paths:
      - path: /hello
        pathType: Prefix
        backend:
          service:
            name: hello-word
```

Ingress Extensions

```
annotations:
  nginx.ingress.kubernetes.io/rewrite-target
  kubernetes.io/ingress.allow-http
```

Ingress Replacements

Gateway
VirtualService

Networking APIs with Gateway

Ingress

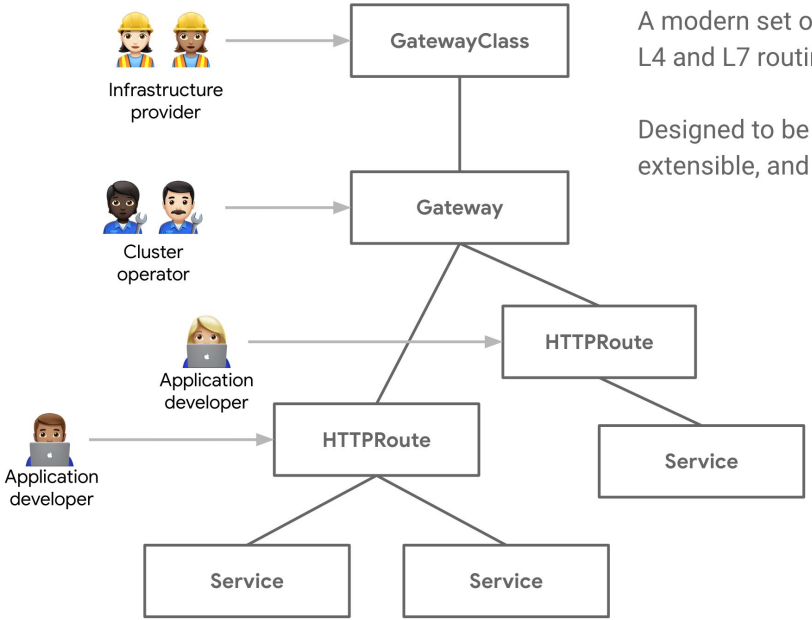
```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: hello-world
spec:
  rules:
  - http:
      paths:
      - path: /hello
        pathType: Prefix
        backend:
          service:
            name: hello-word
```

Gateway APIs

Gateway
HTTPRoute
TCPRoute
BackendPolicy

Extensions

Networking APIs with Gateway



A modern set of APIs for deploying L4 and L7 routing in Kubernetes

Designed to be generic, expressive, extensible, and role-oriented

Ecosystem

- Unlock vast Kubernetes ecosystem
  - Enable more seamless integrations with existing Istio-aware projects like Knative and external-dns
  - Enable integrations with new projects, such as cert-manager or off-the-shelf Helm charts
- Seamless migrations
  - As simple as changing "gateway class" field to migrate between gateway implementations

Networking APIs with Gateway

Kubernetes Gateway

```
apiVersion: networking.x-k8s.io/v1alpha1
kind: Gateway
metadata:
 name: gateway
spec:
 gatewayClassName: istio
 listeners:
 - hostname: "*.domain.example"
   port: 80
   protocol: HTTP
   routes:
     kind: HTTPRoute
```


Istio Gateway

```
apiVersion: networking.istio.io/v1beta1
kind: Gateway
metadata:
  name: gateway
spec:
  selector:
    istio: ingressgateway
  servers:
  - hosts:
    - '*.domain.example'
    port:
      name: http
      number: 80
      protocol: HTTP
```


Networking APIs with Gateway

Kubernetes Route

```
apiVersion: networking.x-k8s.io/v1alpha1
kind: HTTPRoute
metadata:
  name: http
spec:
  hostnames: ["first.domain.example"]
  rules:
  - matches:
    - path:
        type: Prefix
        value: /get
    forwardTo:
      - serviceName: hello-world
        weight: 9
      - serviceName: hello-world-canary
        weight: 1
```

Istio VirtualService

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: route
spec:
  gateways: ["gateway"]
  hosts: ["first.domain.example.com"]
  http:
  - match:
    - uri:
        prefix: /get
    route:
    - destination:
        host: hello-world
      weight: 90
    - destination:
        host: hello-world
        subset: canary
      weight: 10
```


Istio API Equivalence

- Goal: all Istio configuration can be expressed with Kubernetes Gateway.
  - Anything not present in the core API is supported via extensions.
  - API fields are "core", "extended", or "custom", allowing flexibility to add Istio functionality seamless to the APIs.
  - Existing Istio resources will interoperate.
- Tooling will be created to help migrate resources, with support for full Istio API surface.

Extensibility: explicit composition

```
apiVersion: networking.x-k8s.io/v1alpha1
kind: HTTPRoute
metadata:
  name: route
spec:
  rules:
  - filters:
    - type: ExtensionRef
      extensionRef:
        group: security.istio.io
        kind: AuthorizationPolicy
        name: authz-policy
    forwardTo:
      - serviceName: hello-world
```

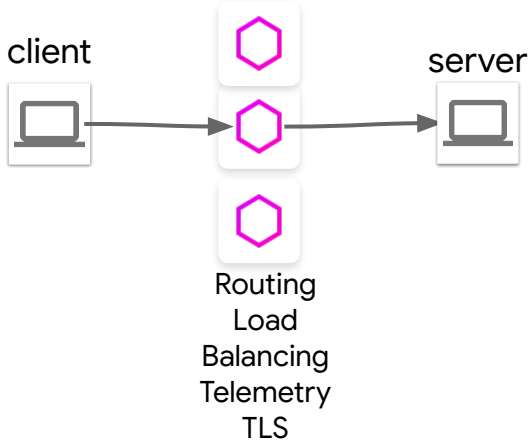

Extensibility: implicit composition

```
apiVersion: networking.x-k8s.io/v1alpha1
kind: HTTPRoute
metadata:
  name: route
spec:
  rules:
  - forwardTo:
      - serviceName: hello-world
---
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: lb-policy
spec:
  host: hello-world
  trafficPolicy:
    loadBalancer:
      simple: LEAST_CONN
```

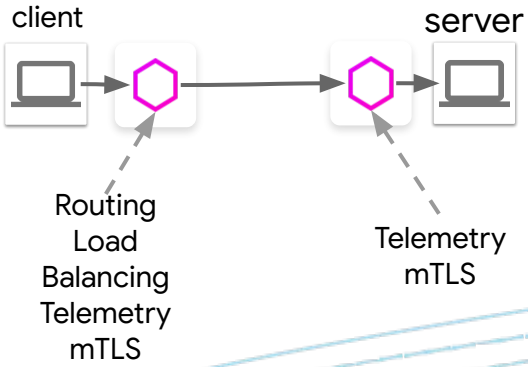

Istio API Future

- All plans tentative so far
- Optimistically, Gateway APIs become the "stable" networking APIs for Istio
- Existing APIs (VirtualService, Gateway, DestinationRule) will stick around for a long time, even after Gateway APIs are promoted to stable

Ingress



Mesh



Istio Implementation

Ingress

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: ingress-route
spec:
  gateways: ["ingressgateway"]
  hosts: ["hello-world.example.com"]
  http: ...
```

Mesh

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: mesh-route
spec:
  gateways: ["mesh"]
  hosts: ["hello-world.default.svc.cluster.local"]
  http: ...
```

Mesh Challenges

- Orders of magnitude more Gateways/proxies to manage
- Consumer vs producer becomes important
  - Generally ingress is almost entirely producer managed. For mesh, its common to have the client control settings. For example, a producer of "foo" may set the LbPolicy to round robin, but a consumer overrides it to least connections.
- Implicit behavior
  - Ingress is opt-in; empty configuration results in no routes. For mesh, we typically have all Services available automatically to allow dropping in place to an existing cluster.

Thank you!

For more information:

- https://kubernetes-sigs.github.io/gateway-api/
- https://istio.io/latest/docs/tasks/traffic-management/ingress/service-apis/
- https://istio.io/latest/about/community/join/

#IstioCon