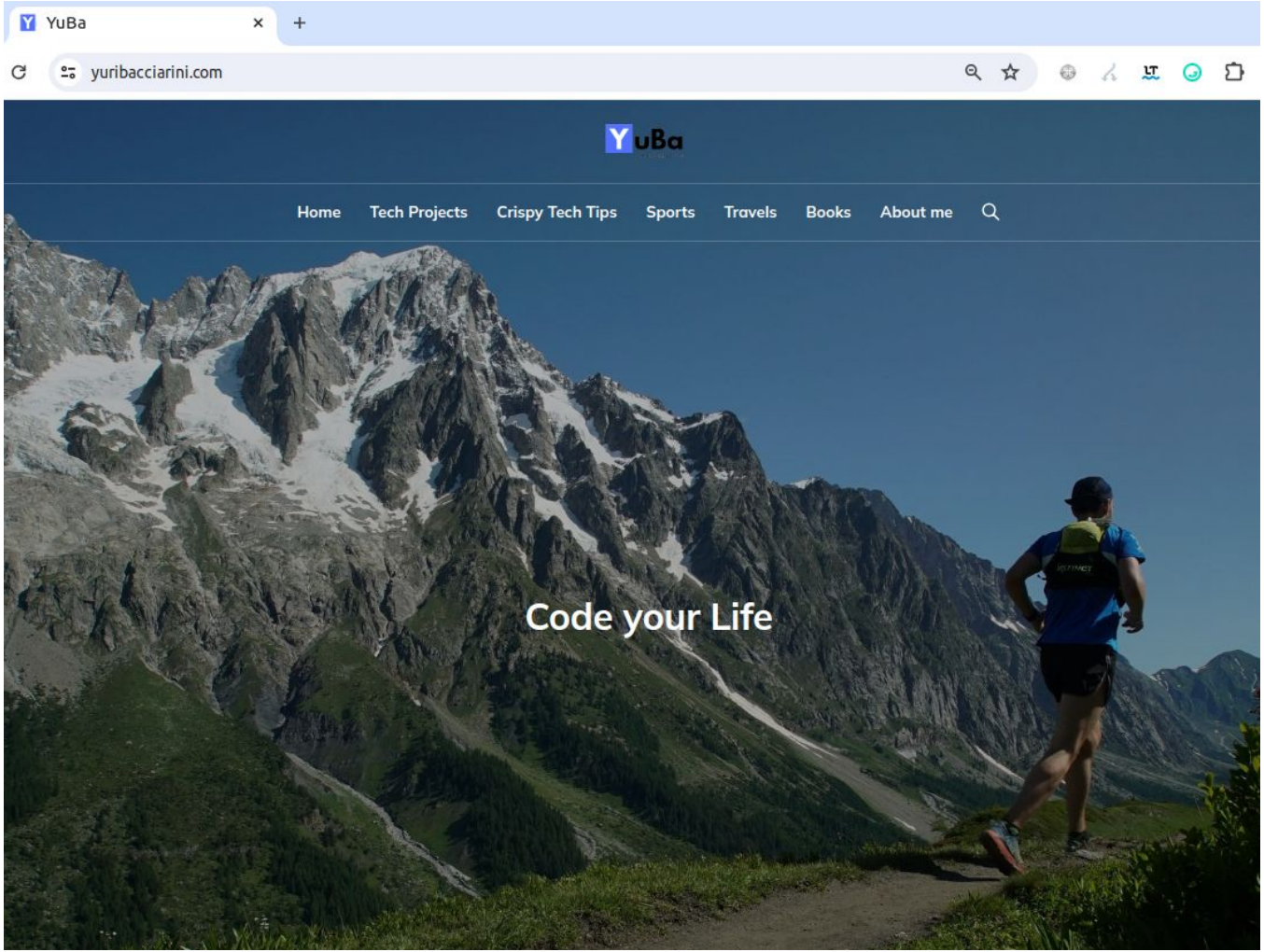
Gateway API

Thanks for all Kubernetes Ingress API
Long life to Gateway API


YuBa
yuribacciarini.com
YuBa
Home
Tech Projects
Crispy Tech Tips
Sports
Travels
Books
About me
Code your Life

Breaking news

Ingress

FEATURE STATE: `Kubernetes v1.19 [stable]`

An API object that manages external access to the services in a cluster, typically HTTP.

Ingress may provide load balancing, SSL termination and name-based virtual hosting.

> **Note:** Ingress is frozen. New features are being added to the Gateway API.

https://kubernetes.io/docs/concepts/services-networking/ingress/

But


DON'T
PANIC


NEW YORK TIMES BESTSELLER
DOUGLAS ADAMS
DON'T PANIC!
THE HITCH-HIKER'S GUIDE TO THE GALAXY
42
OSCAR ABSOLUTE
DOUGLAS ADAMS
GUIDA GALATTICA
per gli
AUTOSTOPPISTI
il ciclo completo
MONDADORI

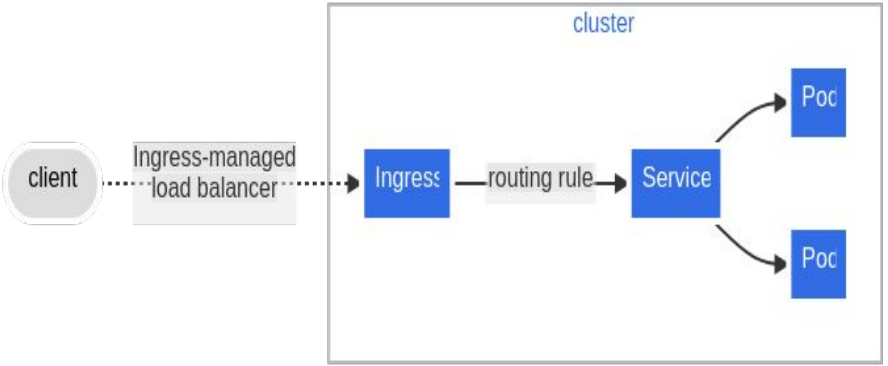
Brief history of Ingress resource

- 📅 2015 – Kubernetes introduces Ingress api
- 🟢 2020 – Ingress api became stable

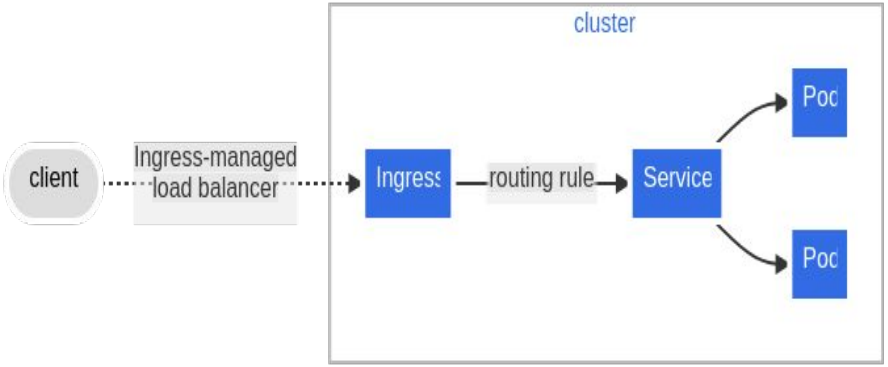
Aim

An API object that manages **external access** to the services in a cluster, typically HTTP

- Kubernetes **layer7** standard
- “**Portability**” (avoid provider lock-in)



How Ingress works




```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: tls-example-ingress
spec:
 tls:
 - hosts:
     - https-example.foo.com
   secretName: testsecret-tls
 rules:
 - host: https-example.foo.com
   http:
     paths:
     - path: /
       pathType: Prefix
       backend:
         service:
           name: service1
           port:
             number: 80
```

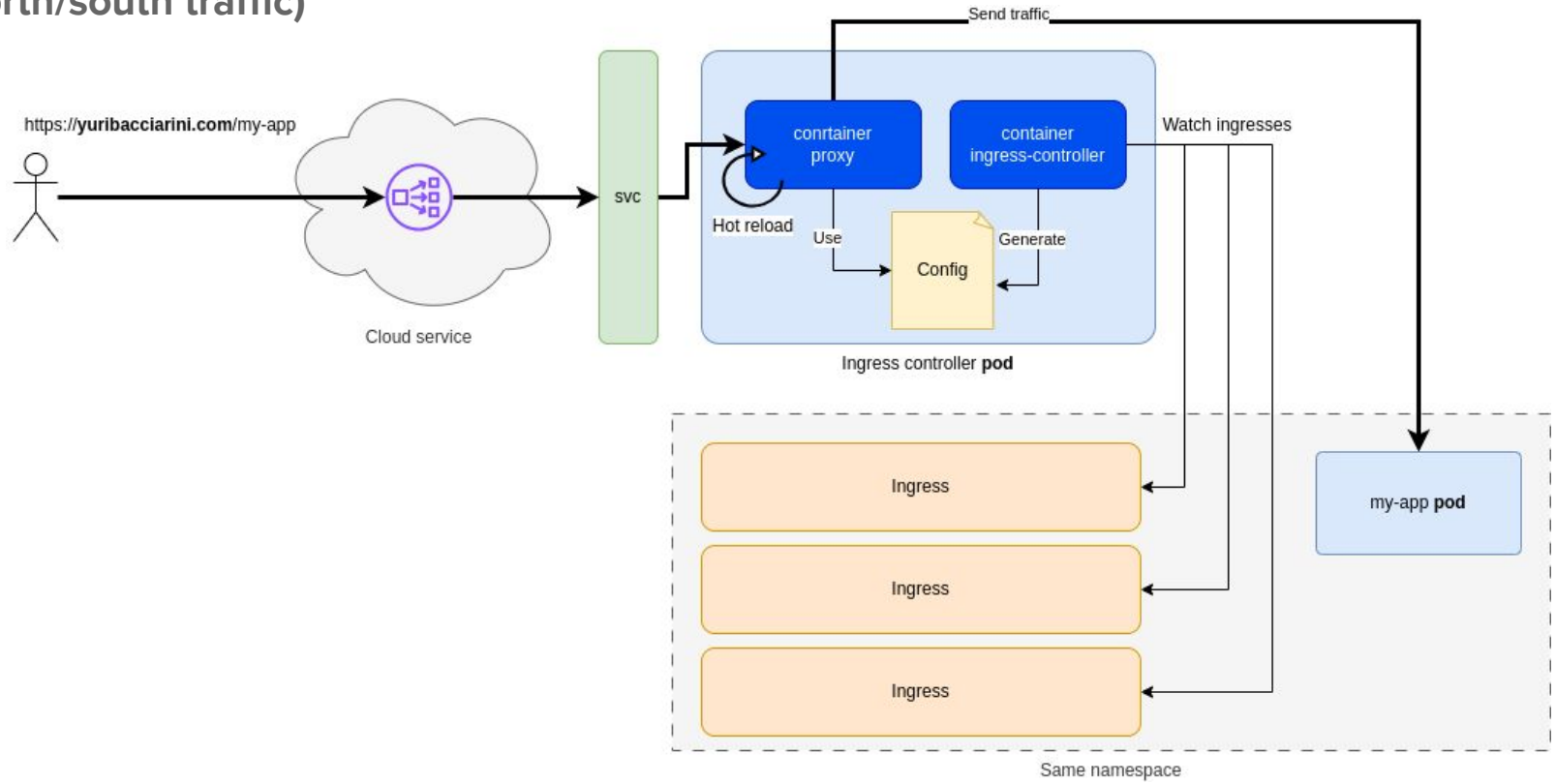
```
~ » k apply -f my-ingress.yaml
```

How Ingress (not) works

Against all odds....
Nothing will happen

How Ingress (really) works

(north/south traffic)



What's wrong with Ingress?

Basically, nothing..

It solved for many years (and it's still doing now) a specific **basic** problem

So what it is missing?

No advanced usage

Not available out-of-the-box

- path rewrite
- ingress config and pods cannot live in different ns
- weighted routing
- header manipulation
- ..

(real) portability

Not role oriented

All configurations in one place (ingress resource):

- tls configuration
- hostname
- one-direction ingressClass configuration
- routing

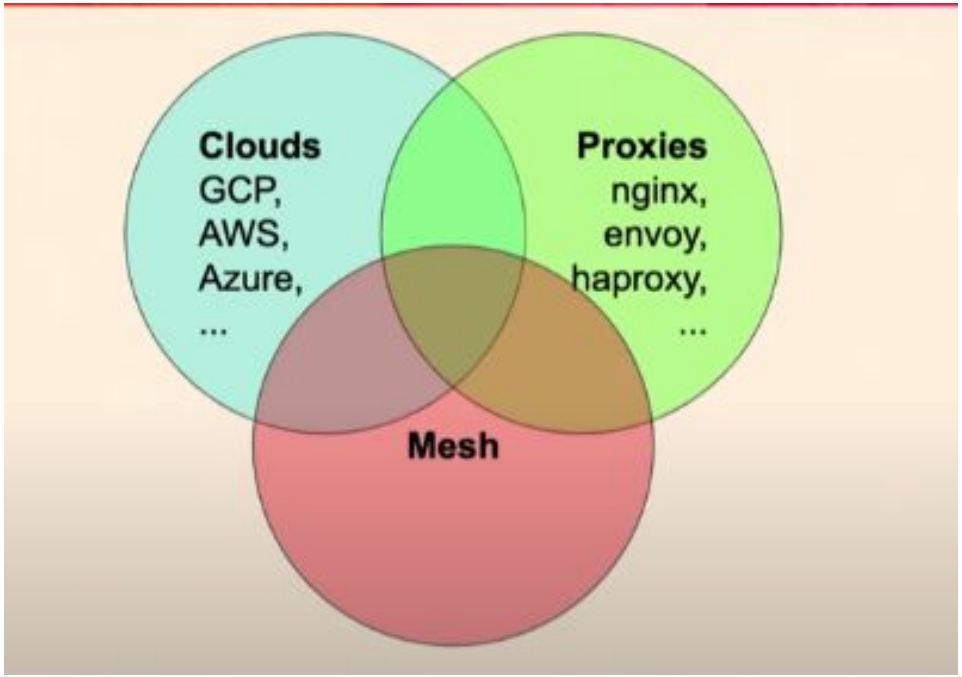
How we survived?

Each Ingress controller has taken its path

Each one with its **CRDs**, custom ingress annotations to mainly add missing Kubernetes ingress features...

= 🤯

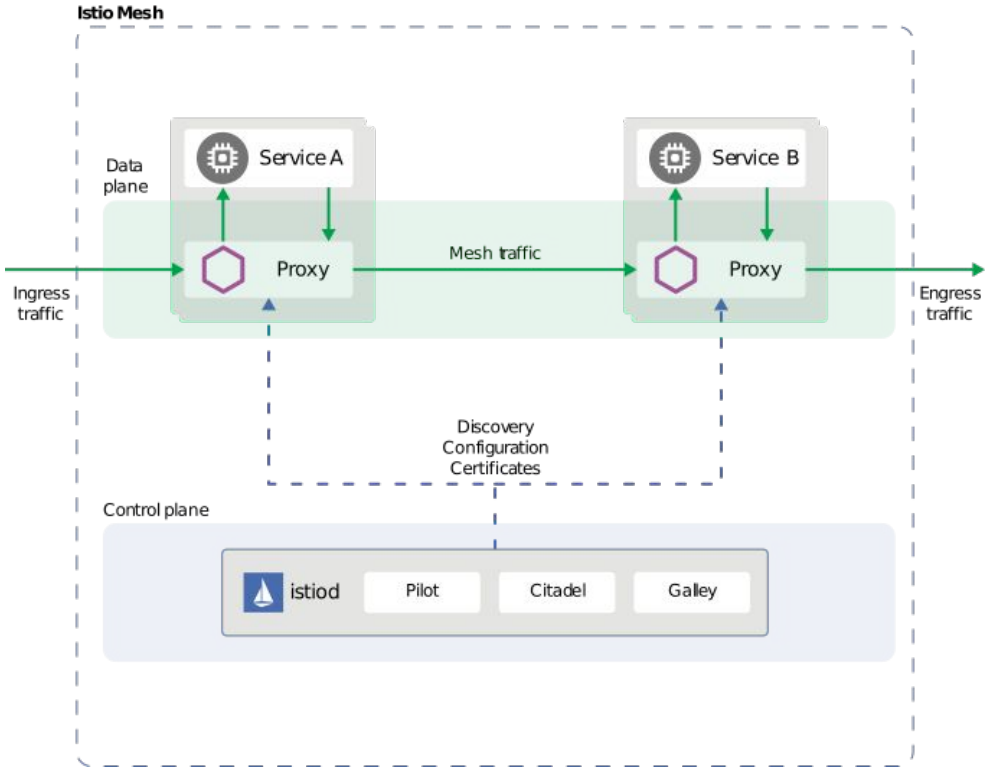
Big picture



Ingress V2 and Multicluster Services - Rohit Ramkumar & Bowei Du, Google
https://www.youtube.com/watch?v=Ne9UJL6irXY

Mesh - Istio

(east/west traffic)



An huge welcome to Gateway API

Who is that?
Gateway API is an open source project managed by the SIG-NETWORK community

Who compose SIG-NETWORK?
“Network Special Interest Group” officially recognized by Kubernetes as a contributor to Kubernetes itself.
More info → https://github.com/kubernetes/community/blob/master/sig-network/README.md

Stability
31 October 2023 → Gateway API v1.0 GA Release

Documentation
https://gateway-api.sigs.k8s.io

..trick or treat?

Areas of Responsibility

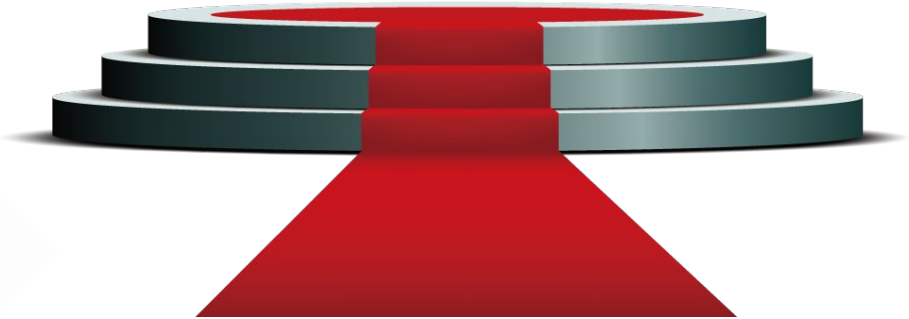
SIG Network is responsible for the following Kubernetes subsystems:

- DNS
- Ingress
- Network plugins / CNI
- Network Policy
- Services / kube-proxy

An huge welcome to Gateway API

Ingress API
Istio

Kubernetes Gateway API



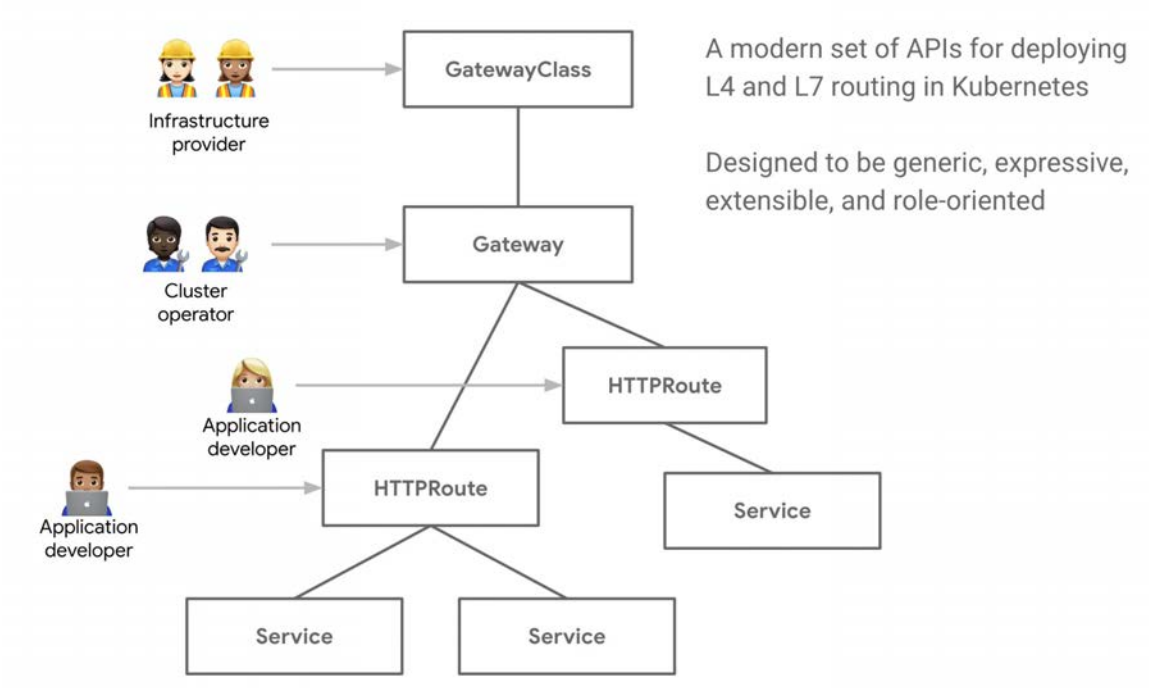
Jul 2022

https://open.spotify.com/episode/6oh4yqPaNv8huSpulwH38Z
Guest: Rob Scott
(software engineer at Google and a lead on the SIG Network Gateway API project)

Intro to Gateway API

It is an API (collection of resources) that model service networking in Kubernetes

- `GatewayClass`
- `Gateway`
- `HTTPRoute`
- `TCPRoute`
- etc..



https://gateway-api.sigs.k8s.io/

Intro to Gateway API

If you're familiar with the older **Ingress API**, you can think of the Gateway API as analogous to a **more-expressive next-generation version of that API.**

Inspired by Istio.

Gateway API concepts

- **Role-oriented** - Gateway is composed of API resources which model organizational roles that use and configure Kubernetes service networking.

- **Portable** - This isn't an improvement but rather something that should stay the same. Just as Ingress is a universal specification with numerous implementations, Gateway API is designed to be a portable specification supported by many implementations.

- **Expressive** - Gateway API resources support core functionality for things like header-based matching, traffic weighting, and other capabilities that were only possible in Ingress through custom annotations.

- **Extensible** - Gateway API allows for custom resources to be linked at various layers of the API. This makes granular customization possible at the appropriate places within the API structure.

Gateway API != API Gateway ⚠️

API Gateway

An API Gateway is a general concept that describes anything that exposes capabilities of a backend service.
A lot of solution in the markets (ex. AWS Api Gateway)

Gateway API

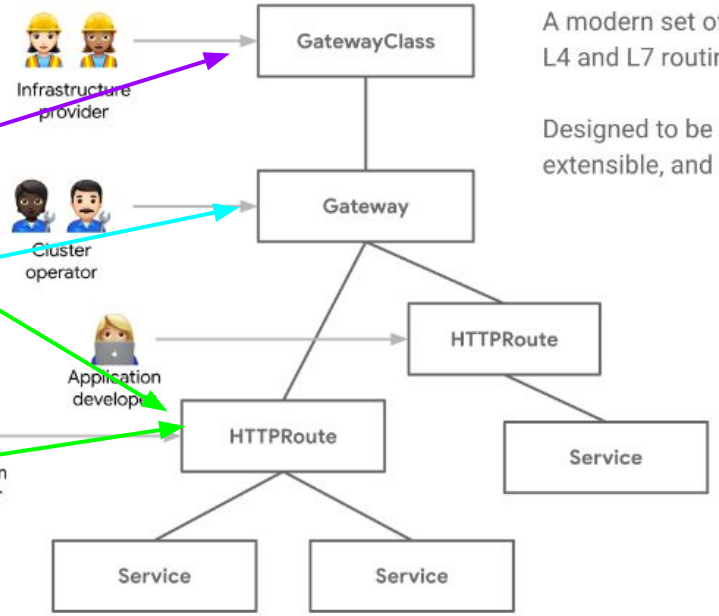
The Gateway API is an interface, or set of resources, that model service networking in Kubernetes

A lot of API Gateways are already implementing Gateway API
Some GA examples: Easegress (GA), Google Kubernetes Engine (GA), Kong (GA), WSO2 APK (GA), NGINX Gateway Fabric (GA)

Implementation status → https://gateway-api.sigs.k8s.io/implementations/

Analogy with Ingress

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
name: tls-example-ingress
annotations:
  nginx.org/proxy-connect-timeout: "30s"
  nginx.org/proxy-read-timeout: "20s"
  nginx.org/client-max-body-size: "4m"
  ...
spec:
ingressClassName: nginx-example
tls:
- hosts:
    - https-example.foo.com
  secretName: testsecret-tls
rules:
- host: https-example.foo.com
  http:
    paths:
    - path: /
      pathType: Prefix
      backend:
        service:
          name: service1
          port:
            number: 80
```



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

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

Analogy with Ingress – controller

A *gateway controller* is software that manages the infrastructure associated with routing traffic across contexts using the Gateway API, analogous to the earlier ***ingress controller*** concept.
Gateway controllers often, but not always, run in the cluster where they're managing infrastructure.

So basically, gateway controller is an evolution of ingress controller.

Let's see with more focus the Gateway APIs

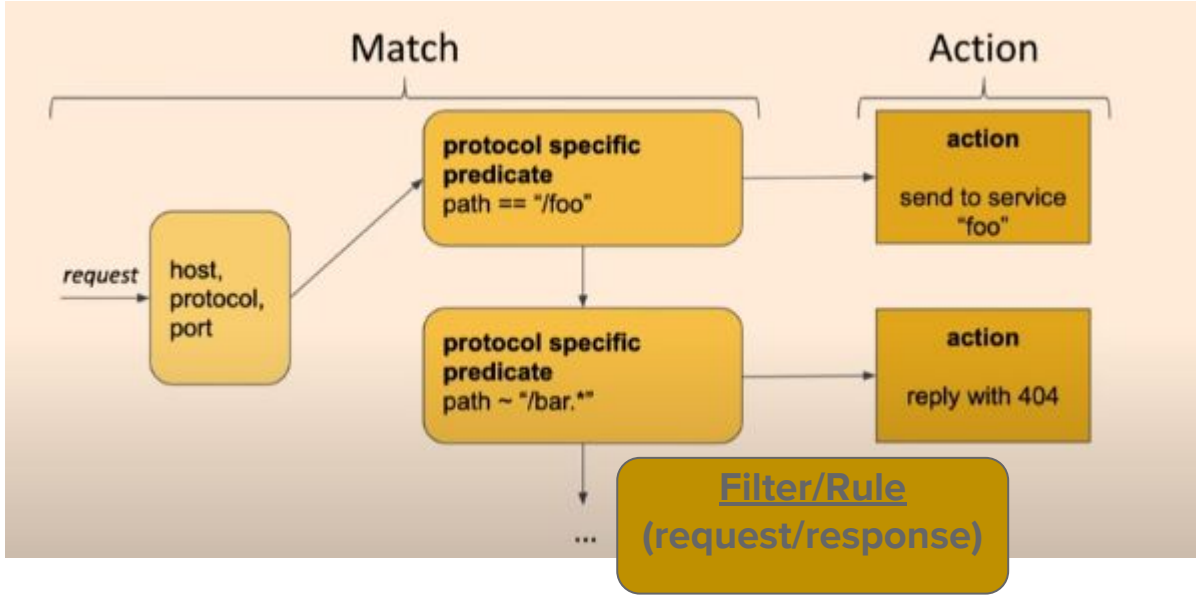
API Types

- Gateway
- GatewayClass
- GRPCRoute
- HTTPRoute
- Policy
- ReferenceGrant

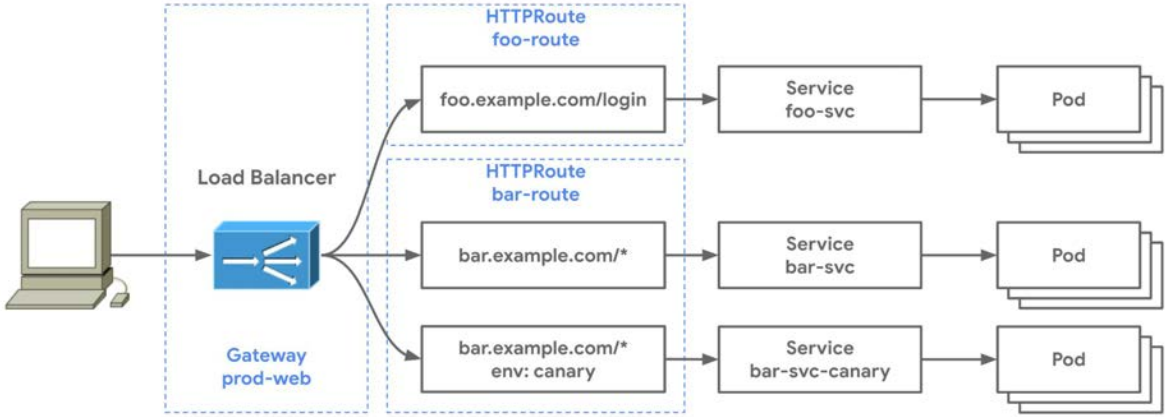
HTTPRoute

HTTPRoute is a Gateway API type for specifying routing behavior of HTTP requests from a Gateway listener to an API object, i.e. Service.

Same concept of Istio's **VirtualService**



HTTPRoute - routing




```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
 name: header-http-echo
spec:
 parentRefs:
   - name: acme-gw
 rules:
   - matches:
       - path:
           type: PathPrefix
           value: /add-a-request-header
     filters:
       - type: RequestHeaderModifier
         requestHeaderModifier:
           add:
             - name: my-header-name
               value: my-header-value
     backendRefs:
       - name: echo
         port: 8080
```

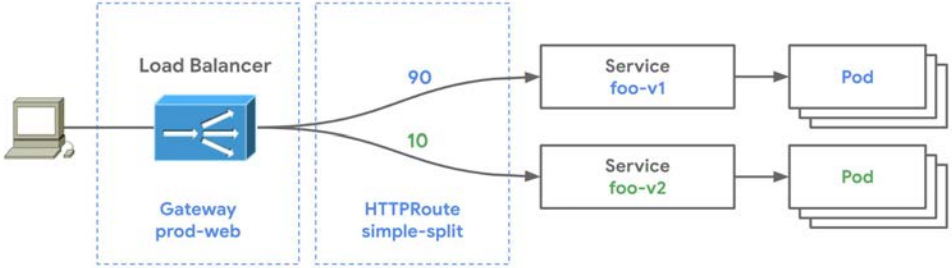
HTTPRoute - rewrite url

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
 name: http-filter-rewrite
spec:
 hostnames:
   - rewrite.example
 rules:
   - filters:
       - type: URLRewrite
         urlRewrite:
           hostname: elsewhere.example
           path:
             type: ReplacePrefixMatch
             replacePrefixMatch: /fennel
     backendRefs:
       - name: example-svc
         weight: 1
         port: 80
```

HTTPRoute - header modifier

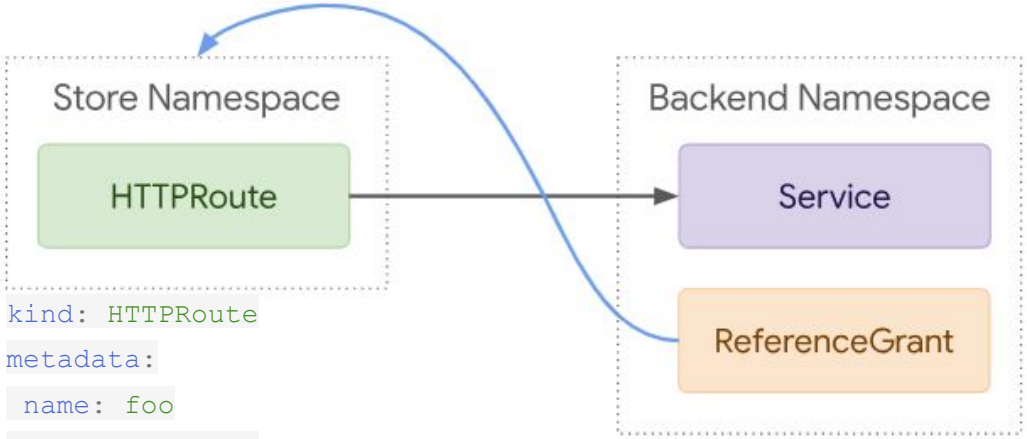
```yaml
  filters:
  - type: ResponseHeaderModifier
    responseHeaderModifier:
      add:
      - name: X-Header-Add-1
        value: header-add-1
      - name: X-Header-Add-2
        value: header-add-2
      - name: X-Header-Add-3
        value: header-add-3
```

HTTPRoute - HTTP traffic splitting




```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
 name: simple-split
spec:
 rules:
 - backendRefs:
   - name: foo-v1
     port: 8080
     weight: 90
   - name: foo-v2
     port: 8080
     weight: 10
```

ReferenceGrant – backend on different ns




```yaml
kind: HTTPRoute
metadata:
 name: foo
 namespace: foo
spec:
 rules:
 - matches:
   - path: /bar
   backendRefs:
     - name: bar
       namespace: bar
```

```yaml
kind: ReferenceGrant
metadata:
 name: bar
 namespace: bar
spec:
 from:
 - group: gateway.networking.k8s.io
   kind: HTTPRoute
   namespace: foo
 to:
 - group: ""
   kind: Service
```

GatewayClass

- cluster-scoped
- defined by the infrastructure provider
- represents a class of Gateways that can be instantiated (ex. public/private gateways)
- same function as the `networking.IngressClass` resource

```yaml
kind: GatewayClass
metadata:
 name: internet
 ...
---
kind: GatewayClass
metadata:
 name: private
 ...
```

Gateway

- namespaced
- triggers load balancing infrastructure provisioning of the GatewayClass type
- Define the hostnames, ports, protocol, termination, TLS settings
- which routes can be attached to this gateway

```yaml
apiVersion:
gateway.networking.k8s.io/v1beta1
kind: Gateway
metadata:
 name: prod-web
spec:
 gatewayClassName: internet
 listeners:
 - protocol: HTTP
   port: 80
   name: prod-web-gw
   allowedRoutes:
     namespaces:
       from: Same
```

Hands-on time!

Summary

Ingress API

- single file configuration
- redundancy (ex. tls configuration)
- no advanced usage by design

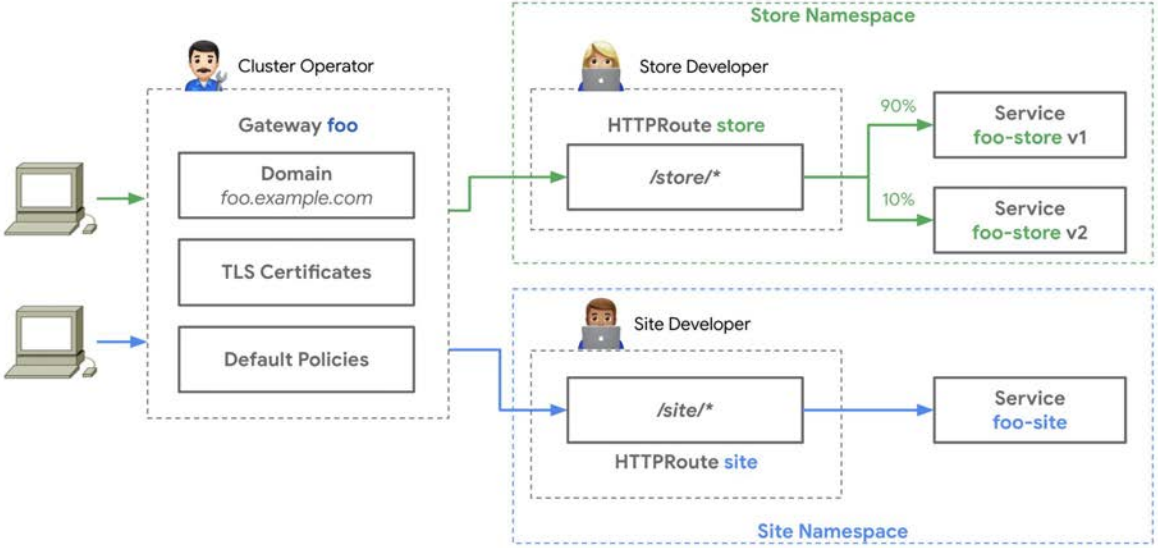
Gateway API

- split configurations in multiple levels
- Role oriented
- cross namespace support
- Different teams can be responsible for a specific configuration part (RBAC)

Empower shared infrastructure governance

Example of shared infrastructure with 2 user personas

- Cluster Operator
- Developers



Just last 2 things looking to the future…

1) Will Gateway API replace the Ingress API?

Q: Will Gateway API replace the Ingress API?

Official Answer: No. The Ingress API is GA since Kubernetes 1.19. There are no plans to deprecate this API and we expect most Ingress controllers to support it indefinitely.

https://gateway-api.sigs.k8s.io/faq/?h=replace

IMO: Yes

2) Gateway API for Service Mesh

gateway-api.sigs.k8s.io/mesh/

Conf42 | Cloud | To read | Tools | 16t hub | magic | Bold and Italic... | ENVision | FinOps | All Bookmarks

gateway api

Search

kuberne
v1.0.

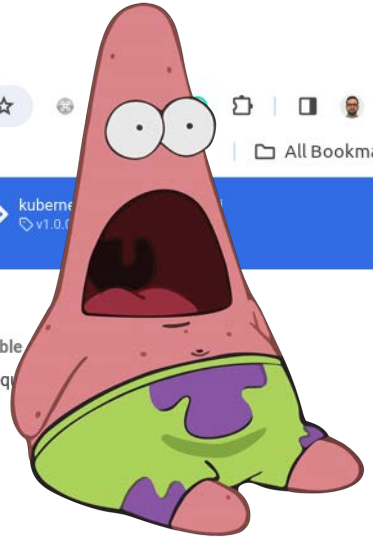
Gateway API for Service Mesh

Table
Requ

Experimental in v0.8.0+

The "GAMMA initiative" refers to the group that is defining how Gateway API can be used for Service Mesh. To date, this group has been able to define service mesh support in the Gateway API with relatively small changes. The most significant change that GAMMA has introduced to date is that, when configuring a service mesh, individual route resources (such as HTTPRoute) are associated directly with Service resources.

This is primarily because there will typically only be one mesh active in the cluster, so the Gateway and GatewayClass resources are not used when working with a mesh. In turn, this leaves the Service resource as the most universal binding point for routing information.



2) Gateway API for Service Mesh

Which Routes attach to a given Service is controlled by the Routes themselves (working with Kubernetes RBAC): the Route simply specifies a `parentRef` that is a Service, rather than a Gateway.

```
kind: HTTPRoute
metadata:
  name: smiley-route
  namespace: faces
spec:
  parentRefs:
    - name: smiley
      kind: Service
      group: core
      port: 80
  rules:
    ...
```

Thanks and see you

👍 **Reach me** for any doubts, ideas, projects, open source stuff, virtual coffee for any tech matter

🔗 **My cave** and **contacts** → https://www.yuribacciarini.com

🔗 All the **hands-on** stuff on **GitHub** → https://github.com/texano00/k8s-gateway-api-playground