

Kubernetes Gateway APIs

John Howard / @howardjohn / Google



#IstioCon

Overview

- Introduction and motivations for Kubernetes Gateway APIs (recap)
- Istio's current implementation status
 - Gateway auto provisioning
 - Mesh support
- Cloud load balancer integrations
- API future



The original Networking API

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



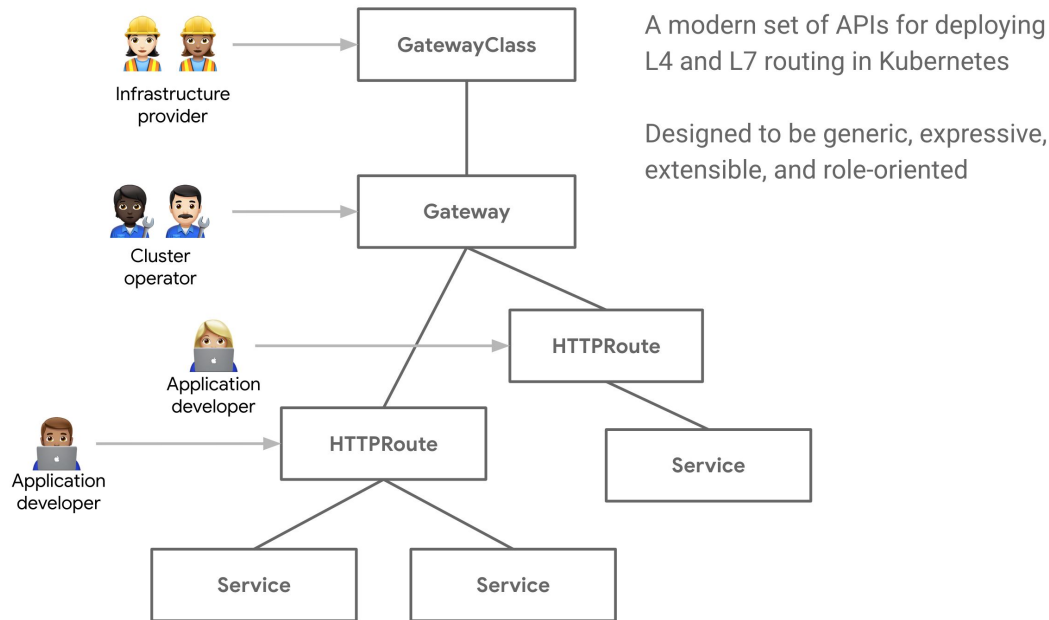
Gateway
HTTPRoute
TCPRoute
BackendPolicy



Extensions



Networking APIs with Gateway



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: gateway.networking.k8s.io/v1alpha2
kind: Gateway
metadata:
  name: gateway
spec:
  gatewayClassName: istio
  listeners:
  - hostname: "*.domain.example"
    port: 80
    protocol: HTTP
```



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: gateway.networking.k8s.io/v1alpha2
kind: HTTPRoute
metadata:
  name: http
spec:
  parentRefs:
  - name: gateway
  hostnames: ["first.domain.example"]
  rules:
  - matches:
    - path:
        type: Prefix
        value: /get
    backendRefs:
    - name: hello-world
      weight: 90
    - serviceName: hello-world-canary
      weight: 10
```

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
```



What is new?

- Full Mesh support
- Conformance testing
- Road to beta
- Automated Gateway Management



Mesh support

```
apiVersion: gateway.networking.k8s.io/v1alpha2
kind: HTTPRoute
metadata:
  name: dual
spec:
  parentRefs:
    # Attach this route to the mesh
    - kind: Mesh
      name: istio
    # And also this Gateway
    - kind: Gateway
      name: gateway
      namespace: istio-system
  hostnames: ["istiocon.example.com"]
  rules:
    - backendRefs:
        - name: example
          port: 80
```



Conformance Testing

projob_name: integ-pilot_istio_postsubmit prowjob_config_url: https://github.com/istio/test-infra/tree/master/prow/cluster/jobs/istio/istio/istio.istio.master.gen.yaml

04-25 15:53 PDT @1518724824489267200 -- 04-13 16:07 PDT @; Served from cache in 0.63 seconds

About ▾ Size ▾ Options ▾ Graph ▾ Local Time: ON Display Clustered Failures List

04-25

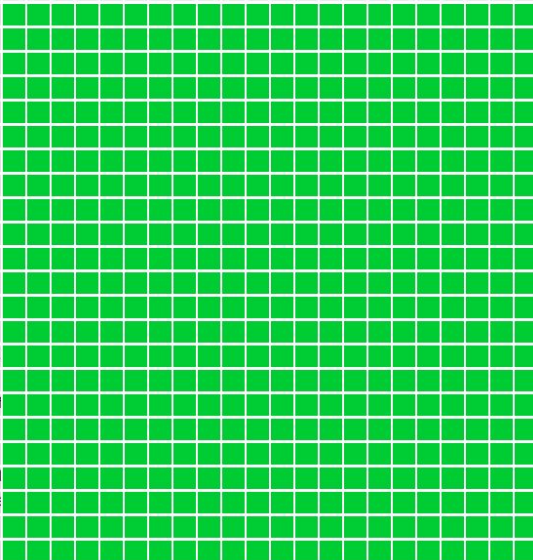
04-23

04-22

missing

[Show 12657 stale tests](#) (no results in last 10+ runs)

istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteCrossNamespace
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteCrossNamespace/Simple_HTTP_request_should_reach_web-backend
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteInvalidCrossNamespace
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteInvalidCrossNamespace/Gateway_should_have_0_Routes_attached
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteInvalidCrossNamespace/Route_should_not_have_Parents_set_in_status
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/0_request_to_/should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/1_request_to_/example_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/2_request_to_/with_headers_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/3_request_to_/v2_should_go_to_infra-backend-v2
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/4_request_to_/v2/example_should_go_to_infra-backend-v2
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatching/5_request_to_/with_headers_should_go_to_infra-backend-v2
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/0_request_to_example.com/should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/1_request_to_example.com/example_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/2_request_to_example.net/example_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/3_request_to_example.com/example_with_headers_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/4_request_to_example.com/v2_should_go_to_infra-backend-v2
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/5_request_to_example.net/v2_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/6_request_to_example.com/v2/example_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteMatchingAcrossRoutes/7_request_to_example.com/with_headers_should_go_to_infra-backend-v1
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteSimpleSameNamespace
istio.io/istio/tests/integration/pilot.TestGatewayConformance/HTTPRouteSimpleSameNamespace/Simple_HTTP_request_should_reach_infra-backend

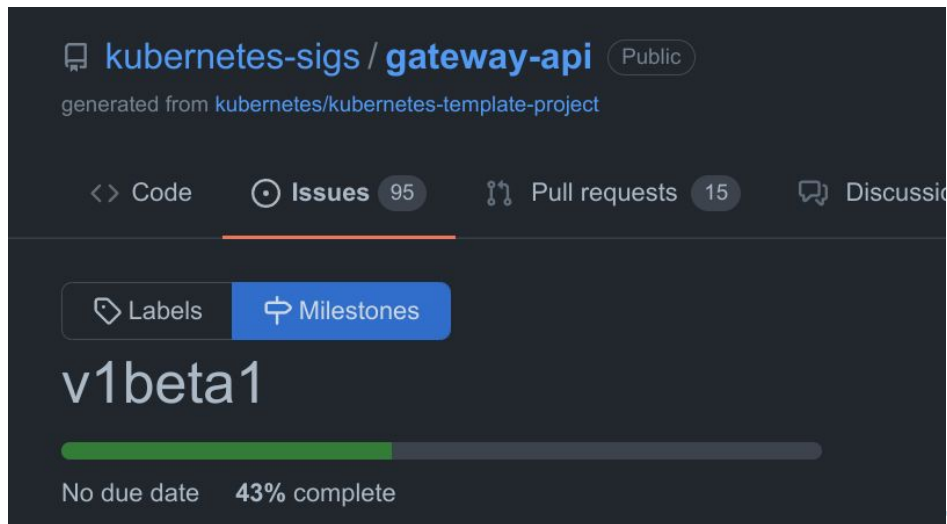


#IstioCon



Path to Beta

- Rapidly approaching beta stability
 - Istio support is expected to go beta when the API does
- Now is the optimal time to give feedback!



Improved Gateway Management

#IstioCon



Gateway Management: Before

```
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
```

User synchronized

```
apiVersion: v1
kind: Service
metadata:
  name: gateway
spec:
  ports:
  - name: http
    port: 80
  selector:
    istio: ingressgateway
  type: LoadBalancer
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gateway
spec:
  selector:
    matchLabels:
      istio: ingressgateway
  template:
    metadata:
      labels:
        istio: ingressgateway
    spec: ...
```



Gateway Management: Before

```
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
  - hosts:
    - '*.database.example'
    port:
      name: mysql
      number: 3306
      protocol: TCP
```

User synchronized

```
apiVersion: v1
kind: Service
metadata:
  name: gateway
spec:
  ports:
  - name: http
    port: 80
  selector:
    istio: ingressgateway
  type: LoadBalancer
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gateway
spec:
  selector:
    matchLabels:
      istio: ingressgateway
  template:
    metadata:
      labels:
        istio: ingressgateway
    spec: ...
```



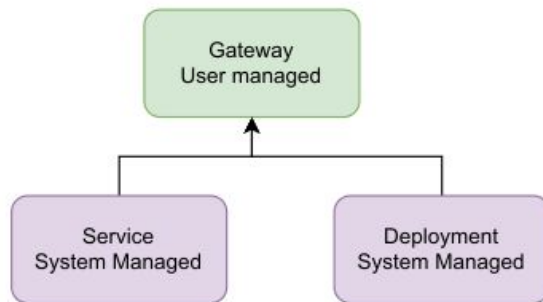
Gateway Management: After

```
apiVersion:  
gateway.networking.k8s.io/v1alpha2  
kind: Gateway  
metadata:  
  name: gateway  
spec:  
  gatewayClassName: istio  
  listeners:  
  - name: default  
    port: 80  
    protocol: HTTP
```



Gateway Management: After


```
apiVersion:
gateway.networking.k8s.io/v1alpha2
kind: Gateway
metadata:
  name: gateway
spec:
  gatewayClassName: istio
  listeners:
  - name: default
    port: 80
    protocol: HTTP
```



Gateway Management: After

```
apiVersion:
gateway.networking.k8s.io/v1alpha2
kind: Gateway
metadata:
  name: gateway
spec:
  gatewayClassName: istio
  listeners:
  - name: default
    port: 80
    protocol: HTTP
```

System
synchronized



```
apiVersion: v1
kind: Service
metadata:
  name: gateway
spec:
  ports:
  - name: http
    port: 80
  selector:
    istio: ingressgateway
  type: LoadBalancer
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gateway
spec:
  selector:
    matchLabels:
      istio: ingressgateway
  template:
    metadata:
      labels:
        istio: ingressgateway
    spec: ...
```



Gateway Management: Interoperability

```
apiVersion:  
gateway.networking.k8s.io/v1alpha2  
kind: Gateway  
metadata:  
  name: gateway  
spec:  
  gatewayClassName: istio  
  listeners:  
  - name: default  
    port: 80  
    protocol: HTTP
```



Gateway Management: Interoperability

```
apiVersion:  
gateway.networking.k8s.io/v1alpha2  
kind: Gateway  
metadata:  
  name: gateway  
spec:  
  gatewayClassName: nginx  
  listeners:  
  - name: default  
    port: 80  
    protocol: HTTP
```

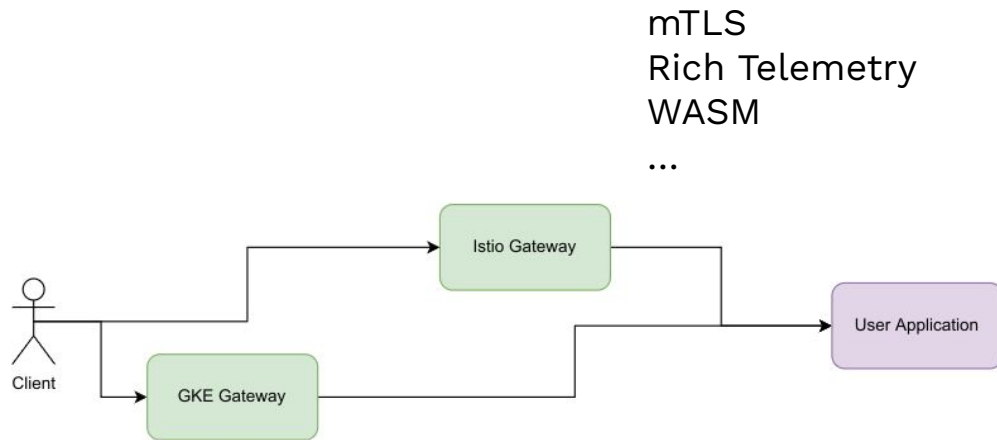


Gateway Management: Interoperability

```
apiVersion:  
gateway.networking.k8s.io/v1alpha2  
kind: Gateway  
metadata:  
  name: gateway  
spec:  
  gatewayClassName: gke  
  listeners:  
    - name: default  
      port: 80  
      protocol: HTTP
```



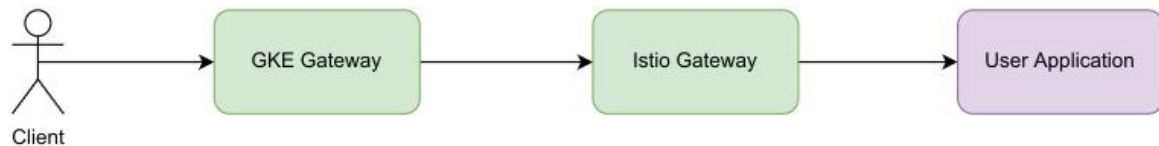
Gateway Management: Interoperability



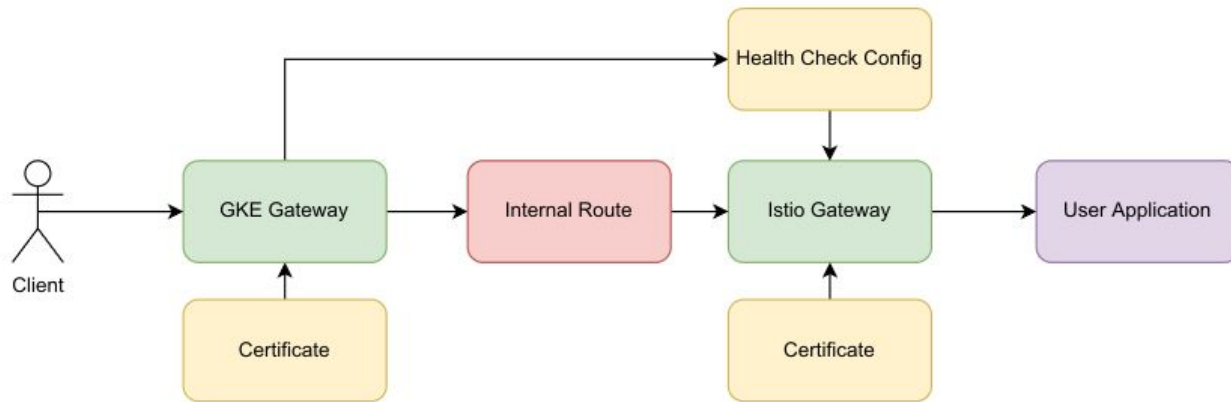
Anycast Global LB
Managed TLS certificates
Cloud Armor policies
...



Best of both worlds: Cloud LB integration



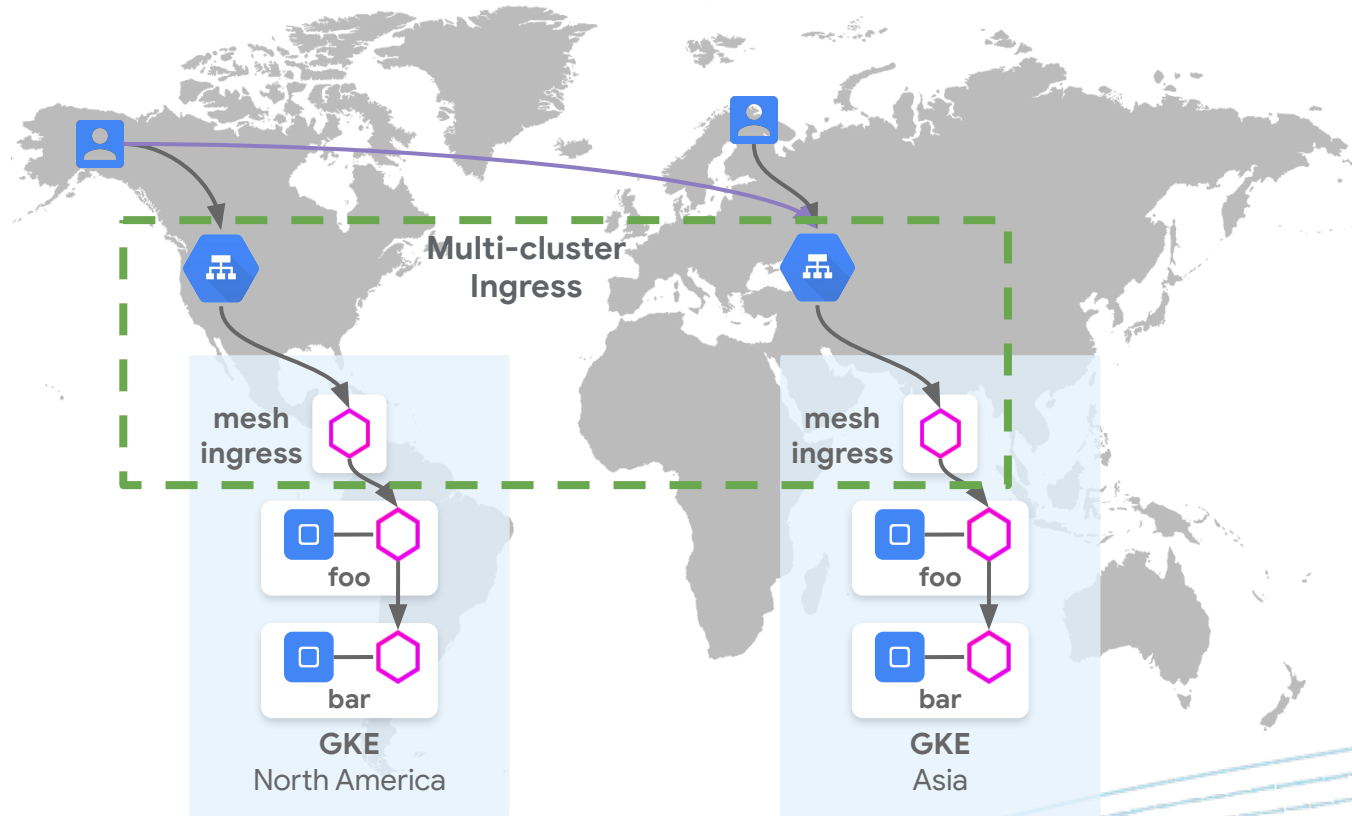
Current state of Cloud LB integration



Future of Cloud LB integration



Future of Cloud LB integration: multi-cluster

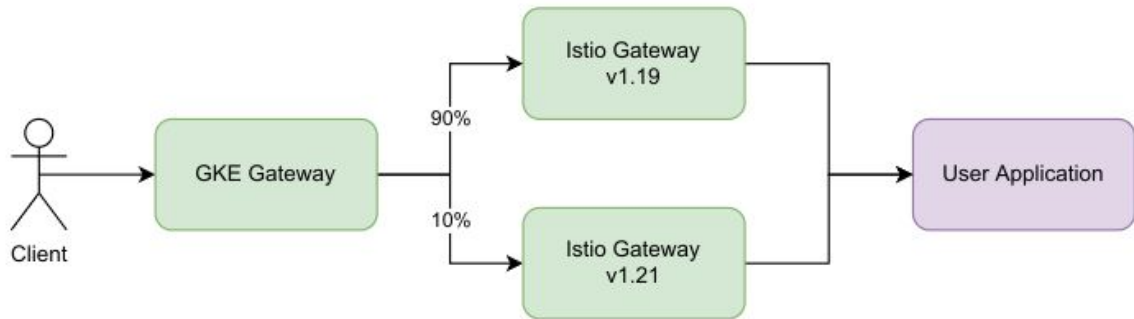


Future of Cloud LB integration: Safe upgrades

User config:



Behind the scenes:



Future of Cloud LB integration

```
apiVersion:
gateway.networking.k8s.io/v1alpha2
kind: Gateway
metadata:
  name: gateway
spec:
  gatewayClassName: istio+gke
  listeners:
  - protocol: HTTPS
    hostname: example.com
```

This will:

- Provision **global** load balancer and IP
- Provision Istio gateways in required clusters/regions
- Configure DNS to point example.com to the IP
- Generate a **managed** certificate for example.com
- Use the **same APIs** as other implementations

Not required by the user:

- Deploying and managing any istio-ingressgateway Deployment/Service
- Manually configuring certs or DNS
- Keeping multiple resources in sync
- Intimate knowledge of implementation details

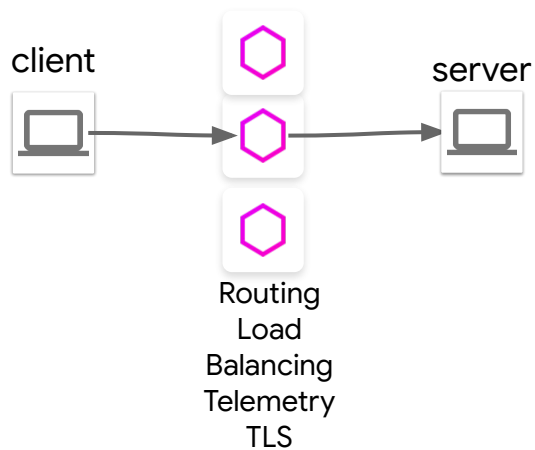


Istio API Future

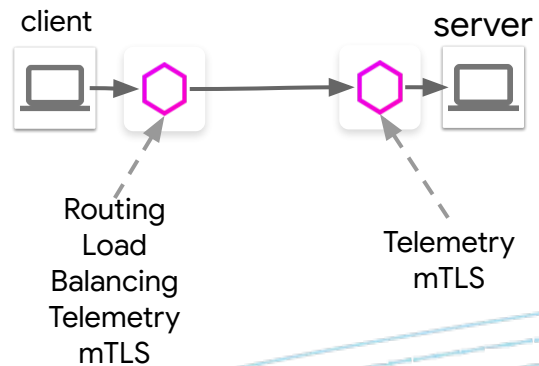
- All plans tentative so far
- Optimistically, Gateway APIs become the "stable" networking APIs for Istio
- Existing APIs (VirtualService, Gateway, DestinationRule) will stick around for a long time, even after Gateway APIs are promoted to stable



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:

- <http://gateway-api.org/>
- <https://istio.io/latest/docs/tasks/traffic-management/ingress/gateway-api/>
- <https://istio.io/latest/about/community/join/>

