K8s-based API Gateway

Steve Flanders
Open Source Summit, North America 2019
About Me

Steve Flanders

Head of Product and Experience at Omnition

@smflanders | https://sflanders.net
Agenda

● Background
  ○ The move from monolith to microservice-based architecture
  ○ An introduction to API gateways
  ○ Kubernetes and what it provides

● Ambassador
  ○ What is it?
  ○ How does it work?
  ○ What does it provide?

● Scenarios
  ○ Edge routing
  ○ Service mesh
  ○ Tips
Background
Monolith to microservices

API Gateway
Kubernetes and what it provides

From [https://kubernetes.io/](https://kubernetes.io/): Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.

---

```yaml
apiVersion: v1
class: Service
metadata:
  name: ambassador
class: Service
spec:
  type: LoadBalancer
  externalTrafficPolicy: LoadBalancer
  ports:
  - port: 80
    targetPort: 8080
  selector:
    service: ambassador
```
Kubernetes and what it provides

- Deployment
- Service
- Deployment
- Service
Kubernetes and what it provides

Deployment

Service
Type = LoadBalancer

Deployment

Service
Type = LoadBalancer
Kubernetes and what it provides

Deployment
Service
Deployment
Service
Ingress makes HTTP/HTTPS services public with standardization on defining:

- Load balancing policies
- SSL / TLS termination
- Name based virtual hosting

Diagram:
- Ingress
- Ingress Controller
- Service
- Deployment
Kubernetes: Ingress versus Service

Service (typically fronts a deployment/statefulset/etc)

- ClusterIP: internal-only virtual IP that other pods can access
- NodePort: exposes an external port that maps to an internal port
- LoadBalancer: exposes an external IP that maps to an internal IP
- ExternalName: CNAME to an outside resource
Ambassador
What is it?

- **Ambassador**
  - An open source, Kubernetes-native microservices API gateway built on the Envoy Proxy.

- **Envoy**
  - An open source edge and service proxy, designed for cloud-native applications.

Put another way: Envoy is a “data plane” and Ambassador is a “control plane”
How does it work?

1. Service owner defines configuration in Kubernetes manifests.
2. Kubernetes API notifies Ambassador of manifest changes.
3. Ambassador parses the change and transforms the configuration to a semantic intermediate representation. Envoy configuration is generated from this IR.
4. The new configuration is passed to Envoy via the gRPC-based Aggregated Discovery Service (ADS) API.
5. Traffic flows through the reconfigured Envoy, without dropping any connections.
What does it provide?

Support for:
- Self-Service via Kubernetes constructs
- Kubernetes-Native Architecture
- Istio Integration
- Flexible Canary Deployments

Features:
- gRPC and HTTP/2 Support
- Authentication
- Rate Limiting
- Integrated Diagnostics
How is it different?

Alternatives fall in three basic categories:

1. Hosted API gateways
   (e.g. Amazon API gateway)
2. Traditional API gateways
   (e.g. Kong)
3. L7 proxies
   (e.g. Traefik, NGINX, HAProxy, or Envoy, or Ingress controllers built on these proxies)

Ambassador differences:

- No vendor lock-in (e.g. hosted gateways)
- No dependency on external database (e.g. Kong)
- Self service and Kubernetes native
- Leverages Envoy
- Offers developer features (e.g. traffic shadowing)
Configuration options by version

- **< 0.50.0**
  - Configmaps
  - Annotations

- **>= 0.50.0**
  - Annotations

- **>= 0.70.0**
  - Annotations
  - CRDs
Encryption

apiVersion: v1
kind: Service
metadata:
  annotations:
    getambassador.io/config: |
      ---
      apiVersion: ambassador/v0
      kind: Module
      name: tls
      ambassador_id: myID
      config:
        server:
          enabled: True

...
Authentication

```yaml
apiVersion: v1
kind: Service
metadata:
  annotations:
    getambassador.io/config: |
      ---
    apiVersion: ambassador/v0
    ambassador_id: myID
    kind: AuthService
    name: authentication
    auth_service: "auth:3000"
    path_prefix: "/auth/api/check"
---
```

```yaml
---
apiVersion: ambassador/v0
kind: Mapping
metadata:
  name: auth-mapping
  namespace: default
spec:
  ambassador_id: myID
  auth_service: "auth:3000"
  path_prefix: "/auth/api/check"
```

Annotations (0.70.0 and older)

CRDs (0.70.0 and NEWER)
Mapping

apiVersion: v1
kind: Service
metadata:
  annotations:
    getambassador.io/config: |
      ---
      apiVersion: ambassador/v0
      kind: Mapping
      name: omnition-mapping
      ambassador_id: myID
      prefix: /
      host: omnition.io
      service: web.default

---

apiVersion: getambassador.io/v1
kind: Mapping
metadata:
  name: omnition-mapping
  namespace: default
spec:
  ambassador_id: myID
  prefix: /
  host: omnition.io
  service: web.default.svc

...
Scenarios
Edge (North/South) Routing

- Ability to control/route ingress traffic
- Offload requirements such as
  - Authentication (e.g. require all ingress traffic to be authenticated)
  - Encryption (e.g. TLS termination and pass-through)
  - Retries and timeouts
Edge Routing: Monolith to microservices
Edge Routing: Monolith to microservices
Edge Routing: Microservices
Service Mesh: Istio

- Automatic load balancing for HTTP, gRPC, WebSocket, and TCP traffic.
- Fine-grained control of traffic behavior with rich routing rules, retries, failovers, and fault injection.
- A pluggable policy layer and configuration API supporting access controls, rate limits and quotas.
- Automatic metrics, logs, and traces for all traffic within a cluster, including cluster ingress and egress.
- Secure service-to-service communication in a cluster with strong identity-based authentication and authorization.
Service Mesh: Istio + Ambassador

Istio Gateway

Query Service

Billing Service

Auth Service

Istio

Envoy

Envoy

Envoy

SaaS service
Tips

- Understand your business requirements
- Do your own research
- Follow first principles
- Follow the open source community
- Don’t write your own
References

Kubernetes

- https://kubernetes.io/docs/concepts/services-networking/ingress/
- https://kubernetes.io/docs/concepts/services-networking/ingress-controllers/
- https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/

Ambassador

- https://www.getambassador.io/about/why-ambassador/
- https://blog.getambassador.io/kubernetes-ingress-nodeport-load-balancers-and-ingress-controllers-6e29f1c44f2d
- https://blog.getambassador.io/api-gateway-vs-service-mesh-104c01fa4784?gi=292eb9fa321a
- https://www.getambassador.io/user-guide/with-istio/

Other

- https://docs.google.com/spreadsheets/d/16bxRgpO1H_Bn-5xVZ1WrRI-0A-GO16egmhvqqLMOmg/edit
- https://medium.com/google-cloud/kubernetes-nodeport-vs-loadbalancer-vs-ingress-when-should-i-use-what-922f010849e0
Questions?