Kubernetes Gateway API : Pourquoi remplacer les Ingress ?

SREFrance Meetup 23/11/2022

Pierre-Yves Aillet

Consultant formateur

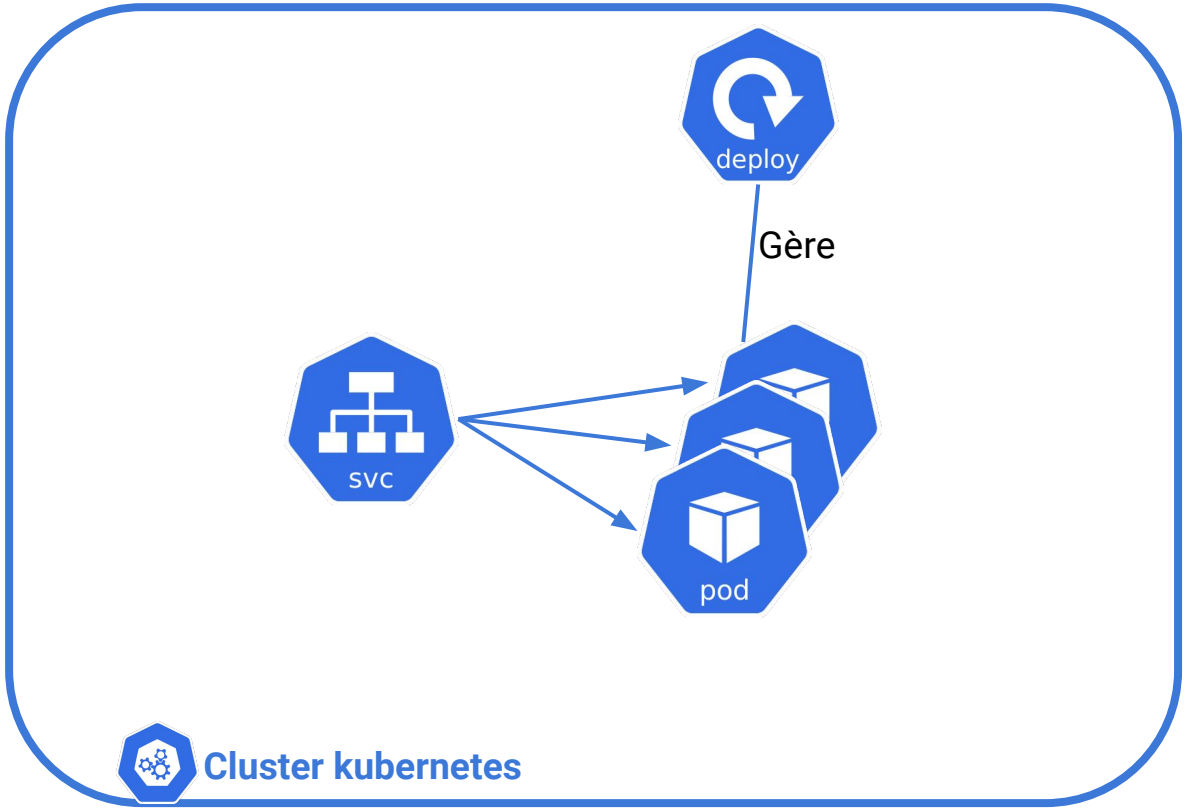
@pyaillet@piaille.fr

@pyaillet

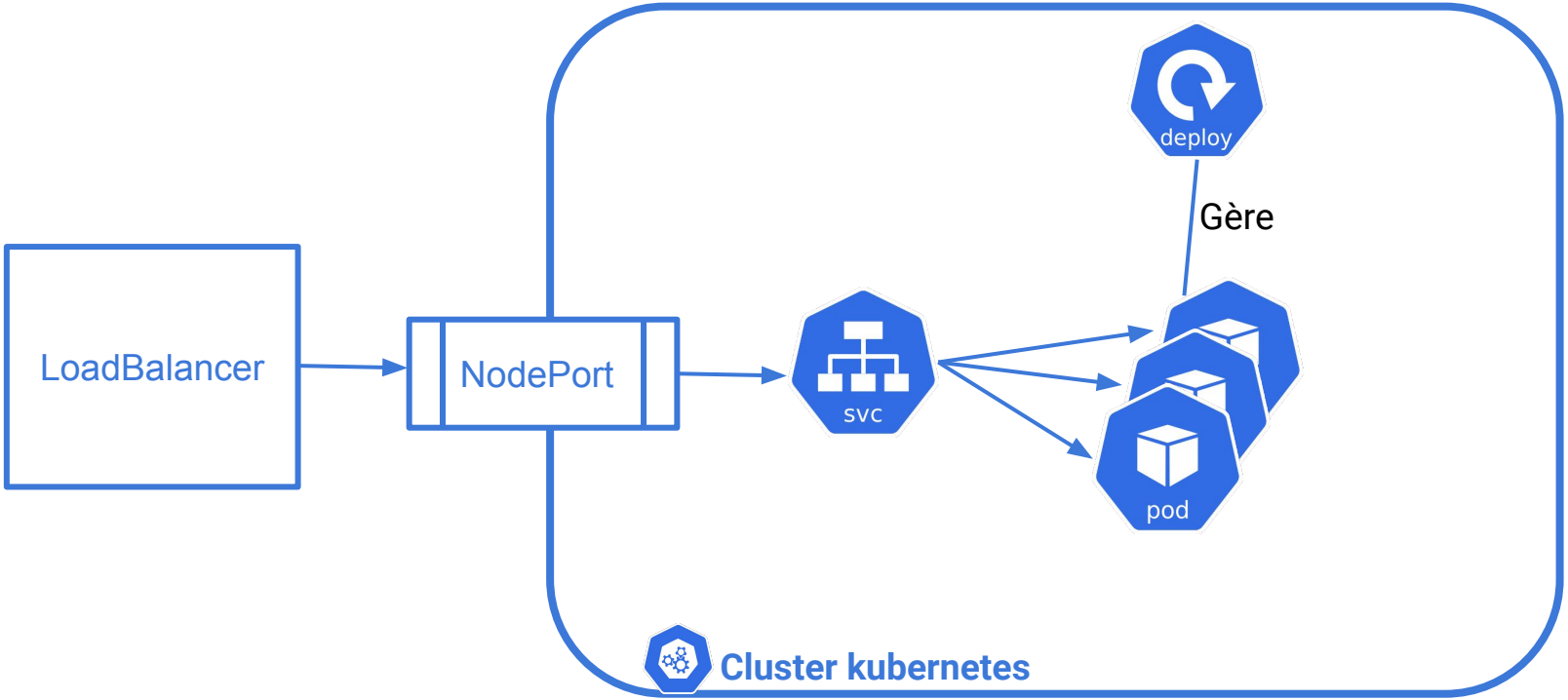
Plan prévisionnel

- Rapide présentation de l’existant
  - Services de type LB - Niveau 4
  - Ingress avec Ingress Controller - Niveau 7
- Problèmes rencontrés
  - Pas d’homogénéïté
  - Configuration du spécifique via annotations
  - Difficultés d’intégration avec les services managés
    - Un (LB,IP) par service (GKE, Kapsule)
    - Un (LB,IP) par Ingress (GKE)
    - Un controller par (LB,IP) (EKS)
- Solution avec la Gateway API
  - Distinction classe de LB (GatewayClass) / instance de LB (Gateway)
  - Controller commun
  - Répartition de la gouvernance
- Et après ?
  - Majoritairement en alpha, mais queques projets en beta et Gateway API Controller en GA sur GKE
  - Initiative GAMMA

Kubernetes

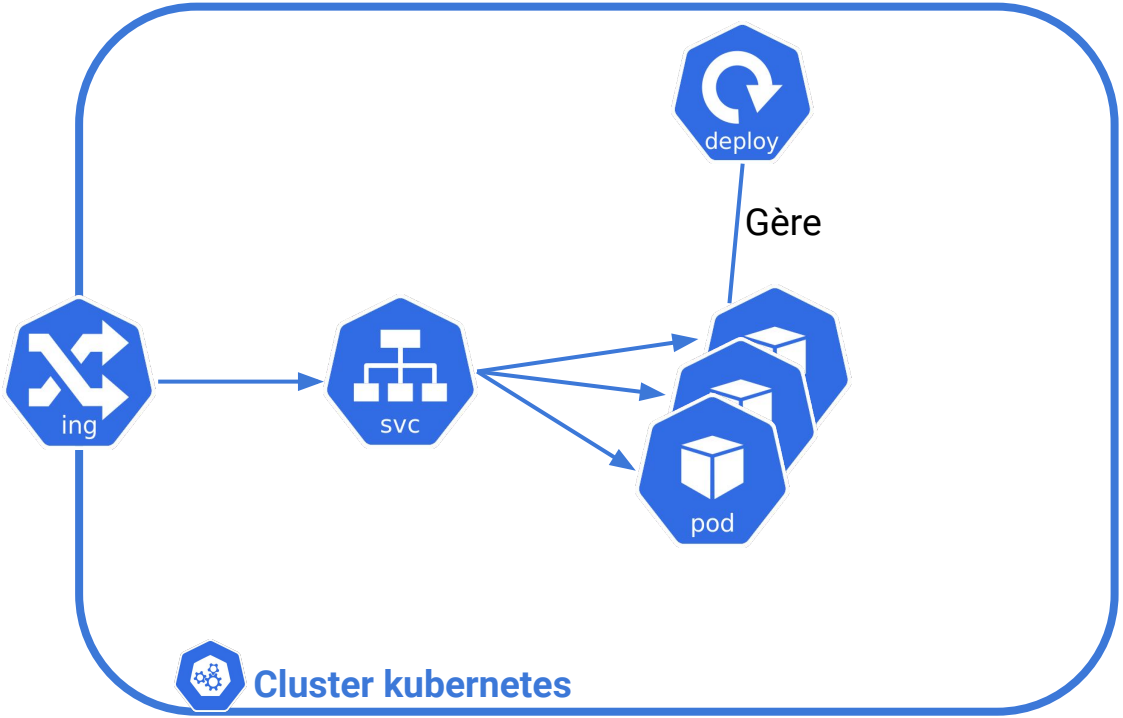


🤔 Comment gère-t-on le traffic entrant ?



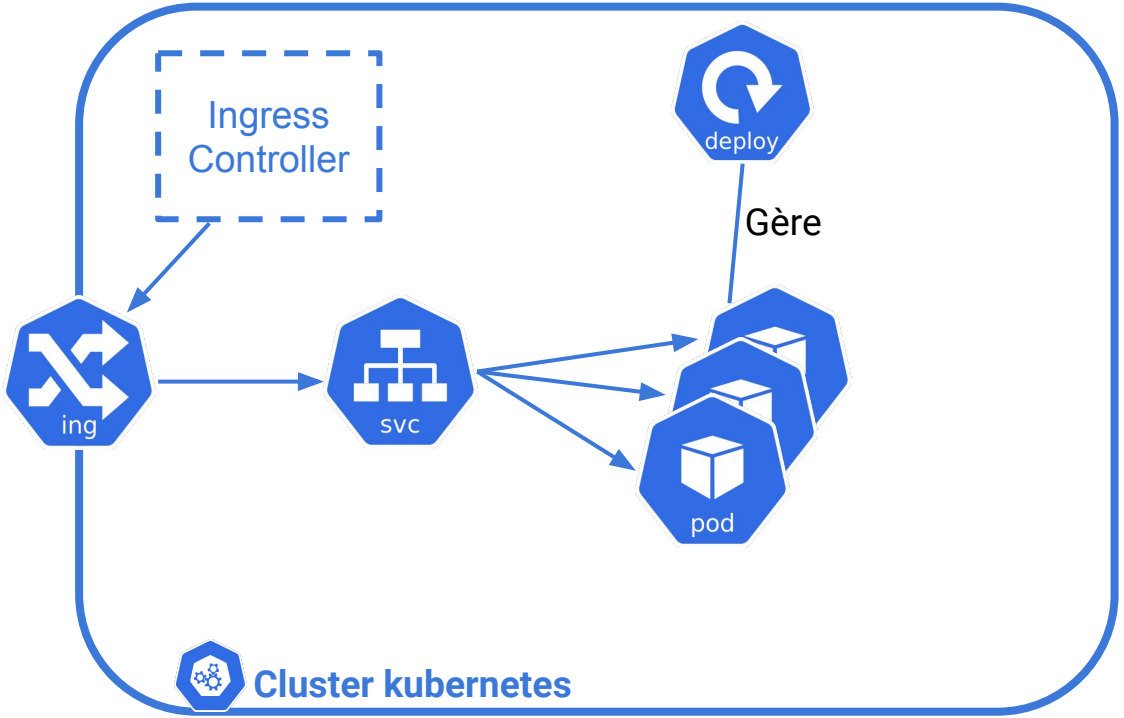
NodePort ou LoadBalancer

🤔 Comment gère-t-on le traffic entrant ?



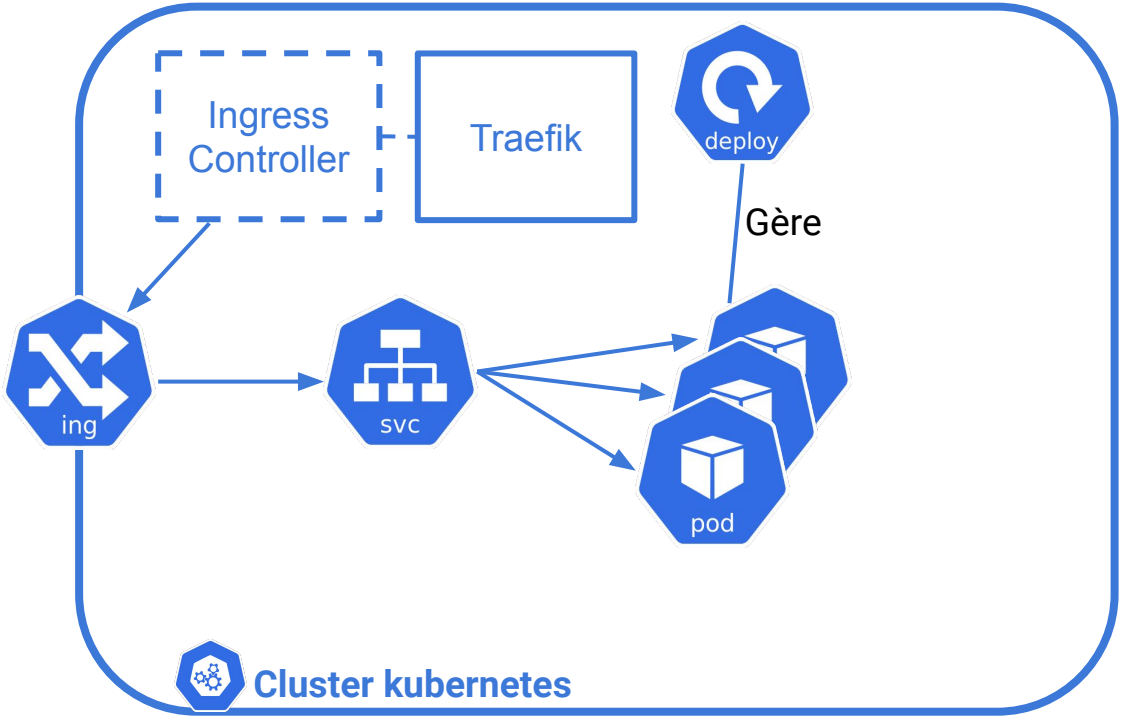
Ingress avec Ingress Controller

🤔 Comment gère-t-on le traffic entrant ?



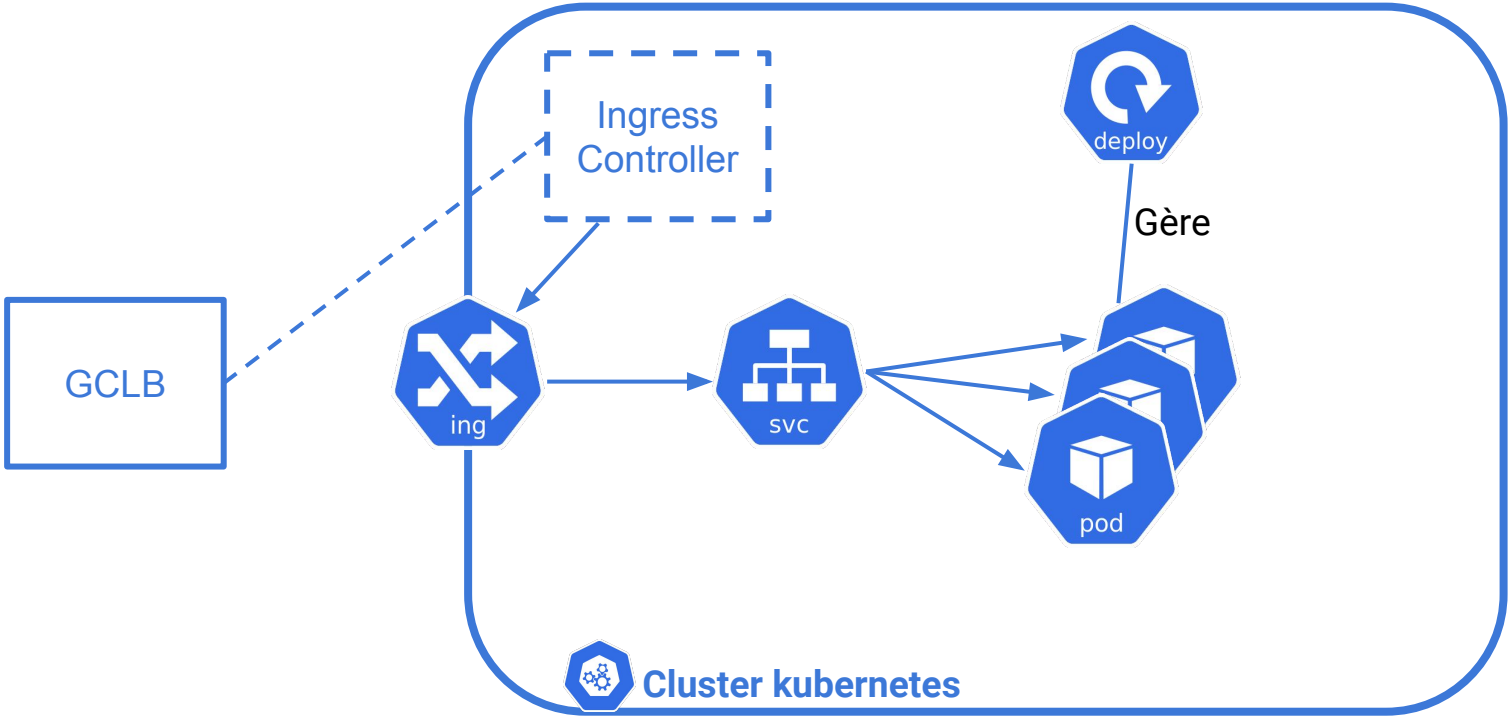
Ingress avec Ingress Controller

🤔 Comment gère-t-on le traffic entrant ?



Ingress avec Ingress Controller

🤔 Comment gère-t-on le traffic entrant ?



Ingress avec Ingress Controller

😕 Problèmes…

- Configuration du spécifique via annotations

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
  annotations:
    service.beta.kubernetes.io/scw-loadbalancer-forward-port-algorithm: "roundrobin"
    service.beta.kubernetes.io/scw-loadbalancer-health-check-delay: "10s"
spec:
  type: LoadBalancer
  selector:
    app.kubernetes.io/name: my-app
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 80
```

svc-lb-kapsule-scaleway.yaml

😕 Problèmes…

- Configuration du spécifique via annotations

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
  annotations:
    networking.gke.io/load-balancer-type: "Internal"
spec:
  type: LoadBalancer
  selector:
    app.kubernetes.io/name: my-app
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 8080
```

svc-google-lb-gke.yaml

😕 Problèmes…

- Configuration du spécifique via annotations

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: my-ingress
  rules:
  - http:
      paths:
      - path: /my-path
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

ingress-nginx.yaml

😕 Problèmes…

- Configuration du spécifique via annotations

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/scheme: internet-facing
spec:
  rules:
  - http:
      paths:
      - path: /my-path
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

ingress-aws-alb.yaml

😕 Problèmes…

- Configuration du spécifique via annotations … ou CRDs

```yaml
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: my-ingress
spec:
  entryPoints:
    - my-entrypoint
  routes:
  - kind: Rule
    match: Host(`my-app.example.com`)
    priority: 10
    middlewares:
    - name: my-middleware
      namespace: default
    services:
    - kind: Service
      name: my-service
      port: 80
      strategy: RoundRobin
      weight: 10
```

ingress-traefik.yaml

```yaml
# Strip prefix /foobar and /fiibar
apiVersion: traefik.containo.us/v1alpha1
kind: Middleware
metadata:
  name: my-middleware
spec:
  stripPrefix:
    prefixes:
      - /foobar
      - /fiibar
```

middleware-traefik.yaml

😕 Problèmes…

- Deux types d’objets différents

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: LoadBalancer
  selector:
    app.kubernetes.io/name: my-app
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 80
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  ingressClassName: my-ingress
  rules:
  - http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

😕 Problèmes…

- Partage des responsabilités

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: no-svc-lb
spec:
  hard:
    services.loadbalancers: "0"
```

😕 Problèmes…

- Intégration avec les services managés
  - Sur GKE : 1 Ingress = 1 LB HTTP(S)
  - Sur Kapsule, OVH Managed Kubernetes : Pas d'Ingress intégré
  - Sur EKS et AKS : Un LB HTTP(S) peut-être partagé par plusieurs Ingress

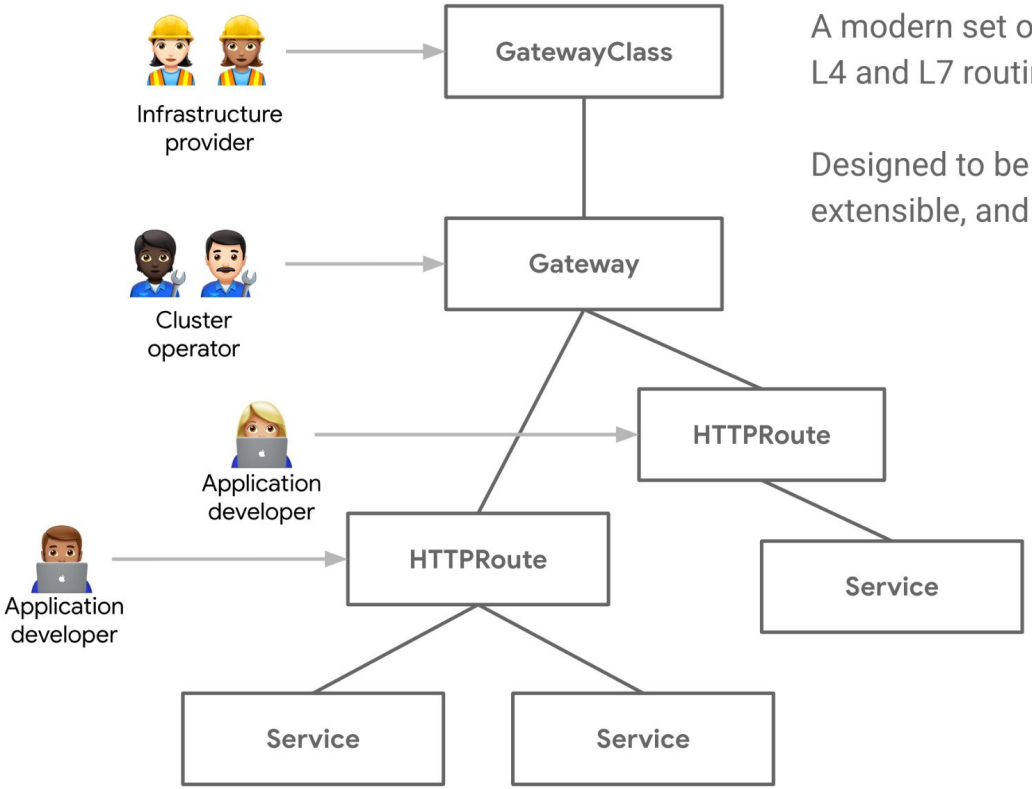
😕 Problèmes…

- Intégration avec les services managés
  - hétérogène
  - configurée par annotation
  - empilement de LoadBalancer

💡 Une solution : Gateway API

- Initiative lancée en avril 2022 par le SIG-NETWORK
- Code et ressources disponible sur github
- Centrée sur les rôles

💡 Une solution : Gateway API



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

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

📝 GatewayClass

Fourniture par :

- Le Cloud Provider (correspondant à un type de LoadBalancer)
- Le fournisseur de l’implémentation de Gateway

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: GatewayClass
metadata:
  name: gke-l7-gxlb
spec:
  controllerName: networking.gke.io/gateway
```

🚪 Gateway

- Instance d’un LoadBalancer pour une classe de Gateway
- Récupère sa configuration via le sélecteur

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

*Routes

`HTTPRoute`

- Équivalent des Ingress mais plus puissants
- `rules` : Possibilité d'utiliser Path, Header, QueryParam, HTTPMethod
- `backendRefs` : Possibilité de définir un poids

Actuellement experimentaux

- `TLSRoute`
- `TCPRoute`

```yaml
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: my-route
spec:
  parentRefs:
  - name: my-gateway
  hostnames:
  - "my-app.example.com"
  rules:
  - matches:
    - path:
        type: PathPrefix
        value: /login
    backendRefs:
    - name: my-service
      port: 80
```

🔀 Filters

- Objectif de standardiser les fonctionnalités offertes par les Gateway
- Trois types de filtres :
  - Core : doivent être gérés
  - Extended : leur implémentation est recommandée
  - Implementation-Specific : spécifiques au fournisseur

🔀 Filters

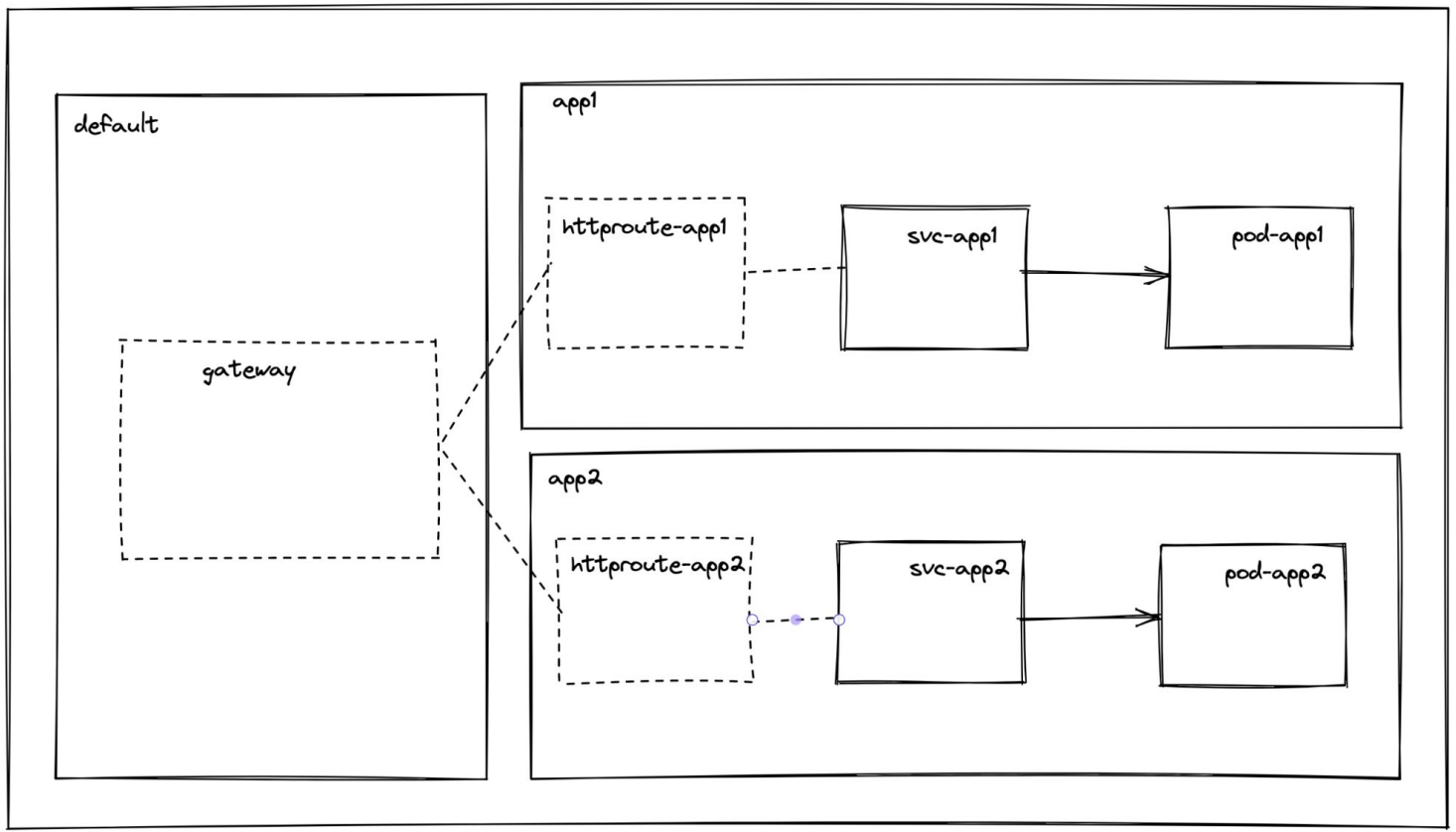
Core :

- requestHeaderModifier
- responseHeaderModifier
- requestRedirect

Extended :

- requestMirror
- urlRewrite

Une démo ?


default
gateway
app1
httproute-app1
svc-app1
pod-app1
app2
httproute-app2
svc-app2
pod-app2

🗺️ On en est où ?

- Passage en beta en juillet
- Liste des implémentations et de leur statut : https://gateway-api.sigs.k8s.io/implementations/
- Passage en GA du contrôleur sur GKE le 11 novembre dernier
- Version beta supportée par Contour, Istio, Kong
- Version alpha ou en cours sur la *plupart des IngressControllers connus*

🔍 Et après ?

- GAMMA (Gateway API for Mesh Management and Administration)
- PolicyAttachment (Gestion des retries, timeout, healthcheck)
- Intégrations avec d’autres solutions en cours
  - Flagger (public-preview)
  - cert-manager (alpha)

Références

- Site du SIG : https://gateway-api.sigs.k8s.io/
- Gateway controller on GKE : https://cloud.google.com/kubernetes-engine/docs/concepts/gateway-api
- Gateway provider with traefik : https://doc.traefik.io/traefik/providers/kubernetes-gateway/
- Gateway API with contour : https://projectcontour.io/guides/gateway-api/
- Article de blog pour la GA de Gateway controller : https://cloud.google.com/blog/products/containers-kubernetes/google-kubernetes-engine-gateway-controller-is-now-ga?hl=en
- Fonctionnement alb-ingress-controller AWS : https://kubernetes-sigs.github.io/aws-load-balancer-controller/v2.2/how-it-works/
- GAMMA Initiative announcement on SMI site : https://smi-spec.io/blog/announcing-smi-gateway-api-gamma/

Crédits

- Icônes : https://github.com/kubernetes/community/tree/master/icons
- Extraits de code : https://carbon.now.sh/
- Schémas : https://gateway-api.sigs.k8s.io/

Des questions ?

https://github.com/pyaillet/kubernetes-demos/

@pyaillet@piaille.fr

@pyaillet