The Design of Apache APISIX Ingress Controller

Chao Zhang

About Me

- Chao Zhang
- https://github.com/tokers
- Apache APISIX PMC
- Open Source Enthusiast
- Contributor of OpenResty, Ingress-Nginx, and so on
- Tars Foundation Ambassador

Agenda

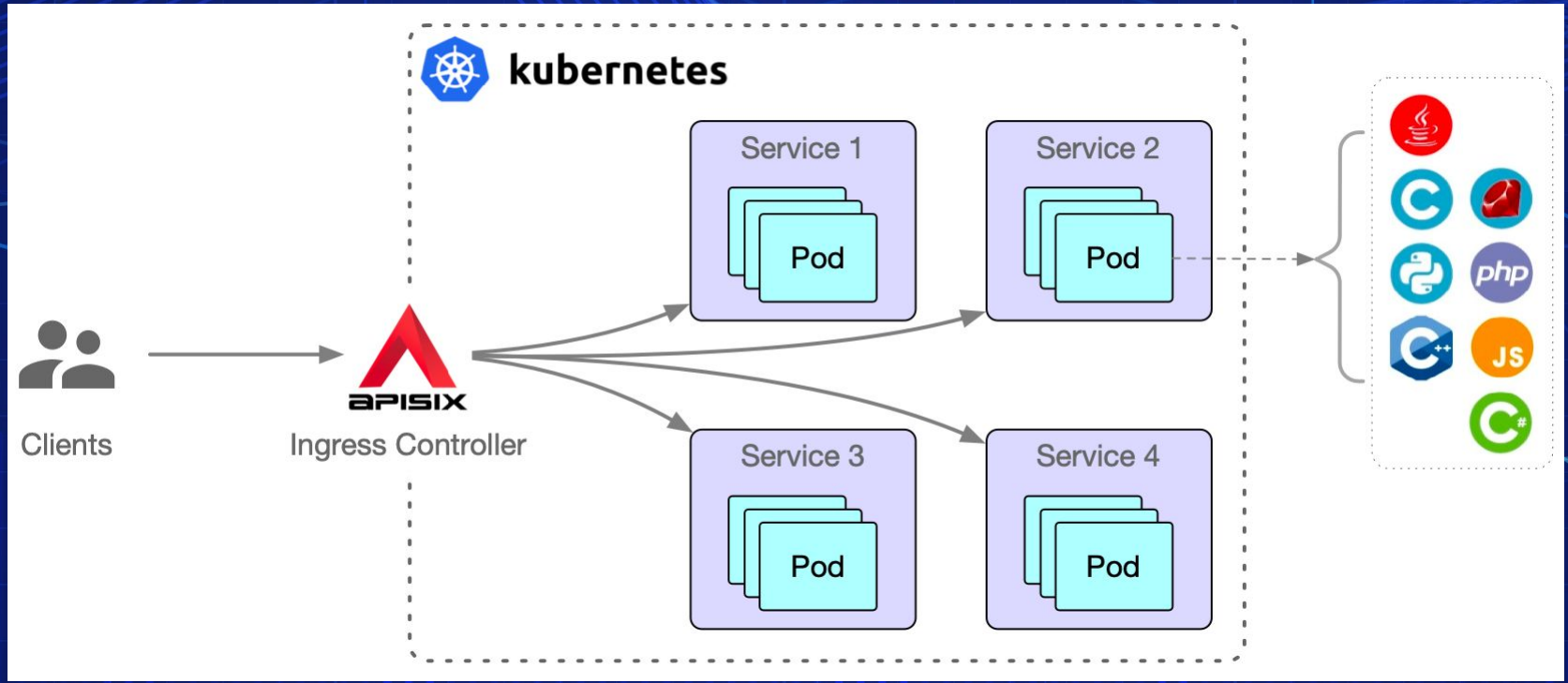
- Why does it appear?
- What does It provide?
- The Deployment Architecture
- Roadmap

Why does it appear?

Apache APISIX

- High performance
- Dynamic
- Extensible
- Richful plugins
- Healthy, active community

Kubernetes Ingress Layer



Why does it appear?

Use Apache APISIX in Kubernetes naturally

- ❏ Declarative Configurations
- ❏ Controller Loop

What does it provide?

Dynamic Configurations

Watch configuration changes and apply them without reloading Apache APISIX.

- ❑ Short feedback period
- ❑ No connections draining
- ❑ No utilization peak (cpu, memory)

Support Kubernetes Ingress

- Easy to migrate from Ingress-Nginx
- All Ingress Versions support
  - networking/v1
  - networking/v1beta1
  - extensions/v1beta1

Support Traffic Split

- ❑ Canary Release
- ❑ Blue Green Deployment
- ❑ Weight-Based & Conditions-Based


apiVersion: apisix.apache.org/v2alpha1
kind: ApisixRoute
metadata:
name: httpbin-route
spec:
http:
- name: httpbin-get
match:
hosts:
- httpbin.org
paths:
- /get
backends:
- serviceName: httpbin-v1
servicePort: 80
weight: 95
- serviceName: httpbin-v2
servicePort: 80
weight: 5

```yaml
apiVersion: apisix.apache.org/v2alpha1
kind: ApisixRoute
metadata:
 name: httpbin-canary-part1
spec:
 http:
 - name: httpbin-v1
   priority: 0
   match:
     hosts:
     - httpbin.org
     paths:
     - /get
   backend:
     serviceName: httpbin-v1
     servicePort: 80
```

```yaml
apiVersion: apisix.apache.org/v2alpha1
kind: ApisixRoute
metadata:
 name: httpbin-canary-part1
spec:
 http:
 - name: httpbin-v2
   priority: 1
   match:
     hosts:
     - httpbin.org
     paths:
     - /get
     exprs:
     - subject:
         scope: Cookie
         name: SESSIONID
       op: In
       set: ["13", "133", "1333"]
   backend:
     serviceName: httpbin-v2
     servicePort: 80
```

Apache APISIX Plugins Integration

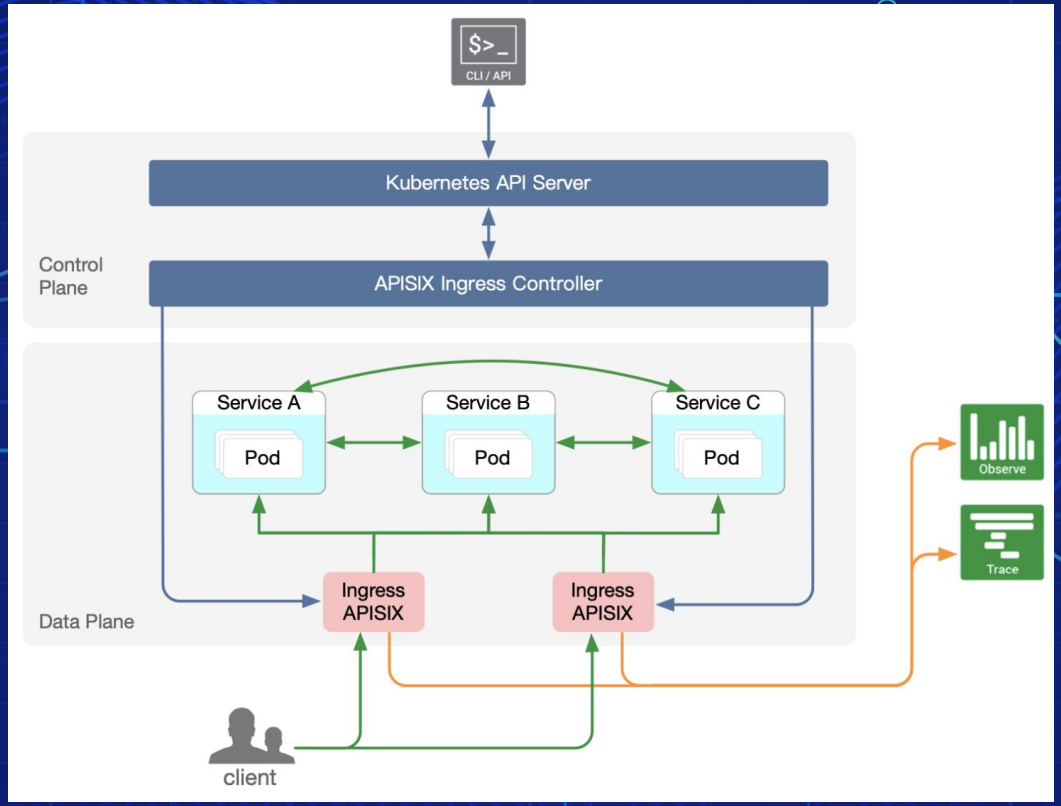
```yaml
apiVersion: apisix.apache.org/v2alpha1
kind: ApisixRoute
metadata:
 name: httpbin-route
spec:
 http:
 - name: httpbin-get
   match:
     hosts:
     - httpbin.org
     paths:
     - /get
   backend:
     serviceName: httpbin-v1
     servicePort: 80
   plugins:
   - name: limit-count
     enable: true
     config:
       count: 2
       time_window: 1
       key: remote_addr
```

```yaml
apiVersion: apisix.apache.org/v2alpha1
kind: ApisixRoute
metadata:
 name: httpbin-route
spec:
 http:
 - name: httpbin-status
   match:
     hosts:
     - httpbin.org
     paths:
     - /status/*
   backend:
     serviceName: httpbin-v1
     servicePort: 80
   plugins:
   - name: uri-blocker
     enable: true
     config:
       block_rules:
       - /status/200
       - /status/206
```

More and More

- Timeout, Load Balancing, Health Check Settings
- Service Discovery
- TCP Proxy
- WebSocket Proxy
- Dynamic Certificates
- ……

The Deployment Architecture


$>_
CLI / API
Kubernetes API Server
Control Plane
APISIX Ingress Controller
Service A
Pod
Service B
Pod
Service C
Pod
Observe
Trace
Ingress APISIX
Ingress APISIX
Data Plane
client

Separation Architecture

- Resilience - scale independently
- Operability - upgrade independently
- Migration - reuse existing APISIX Cluster
- Deployment-Agnostic
  - Deploy APISIX Ingress Controller on Kubernetes
  - Deploy APISIX on VM

Installation

```
helm add apisix https://charts.apiseven.com
helm repo update
helm install [RELEASE_NAME] apisix/apisix-ingress-controller
```

Roadmap

- Publish the GA release in next few months
  - Authentication
  - Observabilities
  - More Ingress annotations support
  - ……
- Will make it conformant with the Gateway APIs
- Make it an independent Apache Top Level Project


Thanks!