Modern HTTP Routing

LISA 2018
2018-10-31

SANDOR SZÜCS
@sszuecs
teapot engineer
WE ARE CONSTANTLY INNOVATING TECHNOLOGY

HOME-BREWED, CUTTING-EDGE & SCALABLE technology solutions

~ 2,000 employees from 77 nations

7 tech locations (HQs in Berlin)

help our brand to WIN ONLINE
ZALANDO TECH’S INFRASTRUCTURE
FOUR ERAS AT ZALANDO TECH

2010
PHP
Data center
PHP files

2015
ZOMCAT
Data center
WAR
LXC (zcloud)

STUPS
AWS
Docker
Cloud Formation
Low level (AWS API)

2016
KUBERNETES
AWS
Docker
Cloud Formation
Kubernetes manifest
Higher abstraction level
LARGE SCALE?
HTTP Routing
HTTP Routing

- % curl https://example.org/a/path?q=value
- HTTP headers
  
  GET /a/path?q=value HTTP/1.1
  Host: example.org
  User-Agent: curl/7.49.0
  Accept: */*
  Cookie: Zm9vCg==
  Authorization: Bearer <token>
HTTP Routing by Host header

HTTP router

client example.org

foo.net

example.org
HTTP Routing by path

z.com/shoes
TCP / TLS LB
http

HTTP router
/shoes
/pay

catalog service
Payment service
HTTP Router basic features

- Healthchecks
- Metrics
- Access logs
- opentracing
Modern HTTP routing
Modern HTTP Routing

● Many possibilities
  ○ Visibility (logs, metrics, tracing)
  ○ Change requests and responses
  ○ Resiliency (ratelimits, circuitbreaker)
  ○ blue / green deployments
  ○ Shadow traffic (clone)
  ○ A/B tests
  ○ Authnz
  ○ API Gateway
  ○ Kubernetes
Skipper a modern HTTP router

https://github.com/zalando/skipper
Extending Skipper

- HTTP router binary
- 1st class library
- Go Plugins
- Lua script support
Skipper: Predicate

Routing Table

- Route
- Predicate
- Filter

Request
Skipper Route

Eskip Syntax:

RouteID: Predicate1 && Predicate2
-> Filter1 -> Filter2
-> “http://backend/url”
Skipper Route

Eskip Syntax:

RouteID: Predicate1 && Predicate2
    -> Filter1 -> Filter2
    -> “http://backend/url”

R1: Host(“example.org”) && Path(“/api”)
    -> modPath(“/api”, “/”)
    -> “http://backend.example.org/”
Ship to production - deployment patterns

- Skipper high level patterns
  - Shadow traffic
  - Blue-green deployments
  - A/B tests
Shadow Traffic

- request
- Response from live
- Drop response
- live
- new
- zalando

Response from live
Drop response

```
https://new.example.org
```
Shadow Traffic - Route

R1: Host("example.org")
    -> tee("https://new.example.org")
    -> "https://backend.example.org/"
Blue-Green deployment

Skipper

90%

v1

10%

v2
Blue-Green deployment

R1: Host("example.org") &&
    Traffic(0.1)
    -> "http://v2.example.org/
R2: Host("example.org")
    -> "http://v1.example.org/"
Blue-Green deployment automation

https://github.com/zalando-incubator/stackset-controller
A/B tests part 1

- Skipper
  - 10% request
  - rest
  - Response with Cookie

A

B

https://github.com/zalando/skipper
A/B test

2 initial routes required which set a Cookie

R1: Traffic(0.1)
   -> responseCookie("version", "A")
   -> "http://serviceA/"

R2: *
   -> responseCookie("version", "B")
   -> "http://serviceB/"
A/B tests part 2

Request with Cookie → Skipper
- version=A
- version=B

Response

A

B

https://github.com/zalando/skipper
A/B test

2 routes with Cookie predicate

R3: Cookie("version", "A")
   -> "http://serviceA/

R4: Cookie("version", "B")
   -> "http://serviceB/"
Visibility

- Metrics
- OpenTracing

Created by Andrey Vasiliev from Noun Project
Metrics

Backend response slow? Yes / No
Metrics

- Requests

Created by Lloyd Humphreys from Noun Project
Opentracing

which service is slow?
Opentracing

- Request
- Skipper add tracing-headers
- with tracing headers
KUBERNETES
A deployment creates a set of Pods
A service is an cluster internal TCP load balancer to Pods
INGRESS

AN EXTERNAL ACCESS POINT TO SERVICES
- Configures http router
Logical Ingress
Real Ingress

- ALB
  - TLS
  - HTTP

- Node
  - Skipper
  - MyApp

- Node
  - Skipper
  - MyApp

- K8s network
- EC2 network
CHALLENGES
- Kubernetes Ingress
Kubernetes - challenges for ingress controllers

Planned cases

- Deployment
- Pod Autoscaling
- Cluster Autoscaling
  - Node shutdown
  - Node creation
- Pod to Node rebalancing

--> require update of pool members or routing table
master

Updates state

Get fresh routing table information
Kubernetes - challenges for ingress controllers

Timeouts to apiserver

- Most controllers can’t detect hanging kube-apiserver calls, because of client-go
- [https://github.com/kubernetes/client-go/issues/374](https://github.com/kubernetes/client-go/issues/374)
Kubernetes - challenges for ingress controllers

Race Conditions to populate changes

- start/stop Pod → update Endpoints
- [Link](https://opensource.zalando.com/skipper/kubernetes/ingress-backends)
- change Kubernetes Service implementation on all nodes (iptables/ipvs config on each node)
Kubernetes - challenges

Unplanned

- Hanging calls to Kubernetes apiserver
- kernel panics → workers can not update master
- Cloud provider node terminations
Please write the title in all capital letters.

Put images in the grey dotted box "unsupported placeholder".

Use bullet points to summarize information rather than writing long paragraphs in the text box.
Kubernetes - solutions for ingress controllers

Faster http routing table updates
● Autoscaling
● Deployments are online
Kubernetes - solutions for ingress controllers

Faster http routing table updates
● Autoscaling
● Deployments are online

Observe connections
● Broken endpoint detection → mark as dead
● Retry connection
Skipper HTTP Ingress Router
https://github.com/zalando/skipper

Skipper documentation
https://zalando.github.io/skipper

Kubectl plugin skipper
https://github.com/szuecs/kubectl-plugins

Kube AWS Ingress Controller
https://github.com/zalando-incubator/kube-ingress-aws-controller

External DNS
https://github.com/kubernetes-incubator/external-dns

Zalando Cluster Configuration
https://github.com/zalando-incubator/kubernetes-on-aws
QUESTIONS?

SANDOR SZÜCS
TECH INFRASTRUCTURE
TEAPOT ENGINEER

sandor.szuecs@zalando.de
@sszuecs
Illustrations by @01k
One last thing

Cluster Ratelimits
Cluster Ratelimit

- clients

Cluster Ratelimit

Skipper

Node

Skipper

Node

Skipper

Node

Skipper

Node

Skipper

Node

Skipper swarm
Kubernetes - solutions

Retries
Retry

Skipper

1st request
Retry

Skipper

retry