OpenResty在同程旅游的应用

OpenResty™ | IT大咖说 知识共享平台

演讲目录

- 自我介绍
- OpenResty的实际应用
- 7层负载那些事
- 多功能反向代理Api-Gateway
- 性能优化一例
- Q&A

自我介绍

吴中骅

同程艺龙

机票事业群 - 资深架构师

Node.js 和 Go发烧友

《Node.js实战》两季作者

《同程技术故事》作者

喜爱摄影，旅游

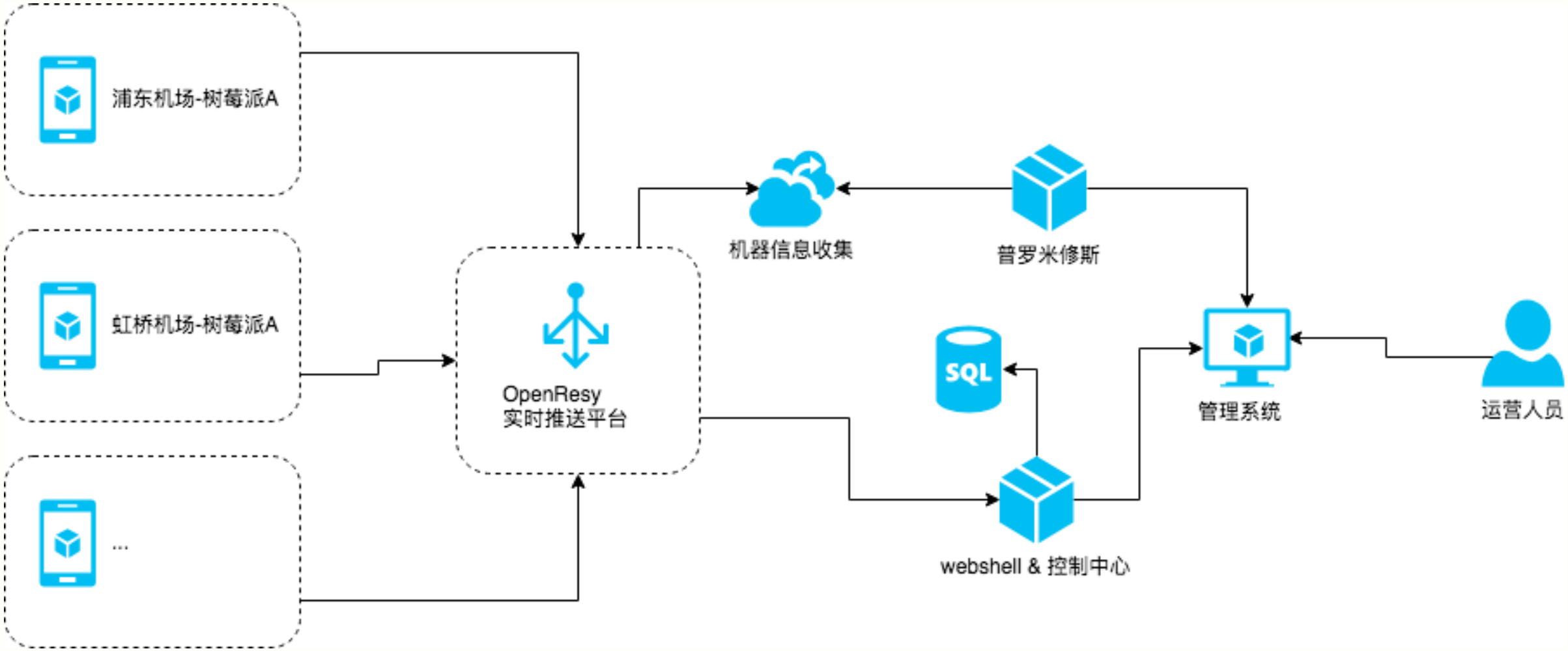
OpenResty™ | IT大咖说 知识共享平台

同程艺龙Openresty的应用

- ApiGateway 多功能7层反向代理
- Realtime 实时推送系统（基于ws协议）
- 基于Lua的ServerLess服务（做了2个，还有一个Node.js版本）
- waf 软防火墙（基于开源 lua-resty-waf，配置内存化，引入控制中心，动态下发基于url地址的不同waf配置）

OpenResty ™ | IT大咖说 知识共享平台

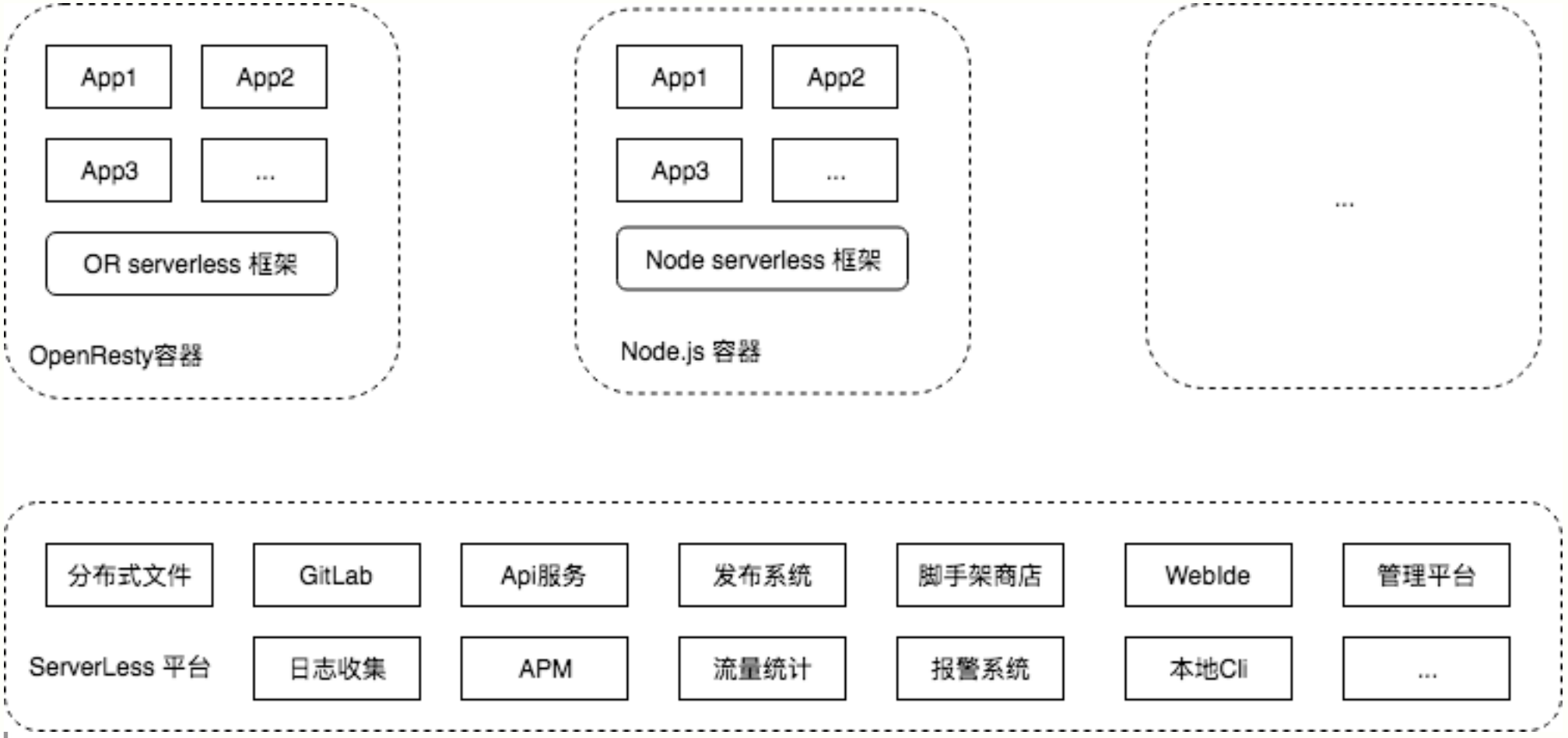
Realtime实时推送系统



OpenResty ™
IT大咖说
知识共享平台

ServerLess服务

- docker 容器做资源隔离，serverless框架做语言vm（虚拟机）代码隔离，按需运行项目



为什么要7层负载

1. 针对http应用做一些分流的策略，比如针对域名、目录结构

2. 通过端口检测到服务器内部的故障，比如根据服务器处理网页返回的状态码、超时等等，并且会把返回错误的请求重新提交到另一个节点

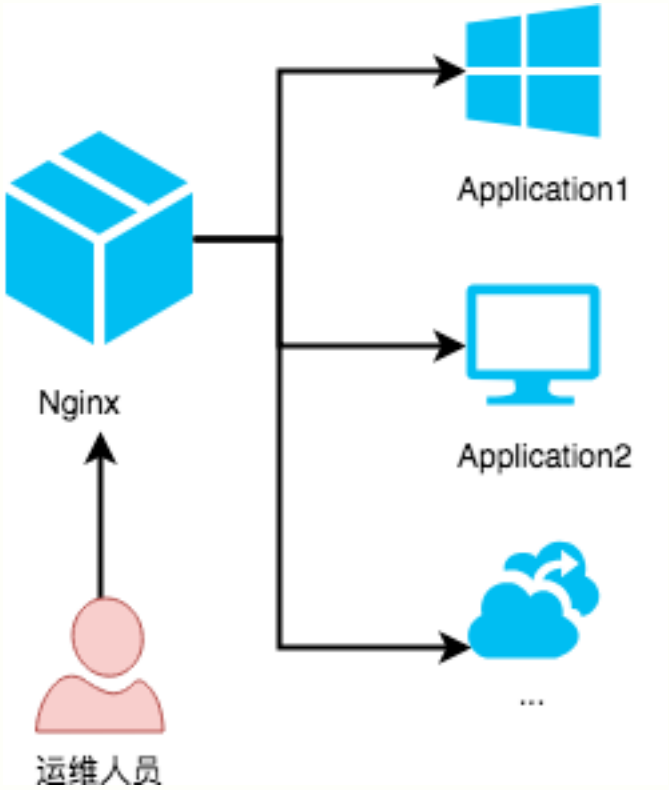
3. 开启缓存功能，反向代理加速

4. 可作为静态网页和图片服务器

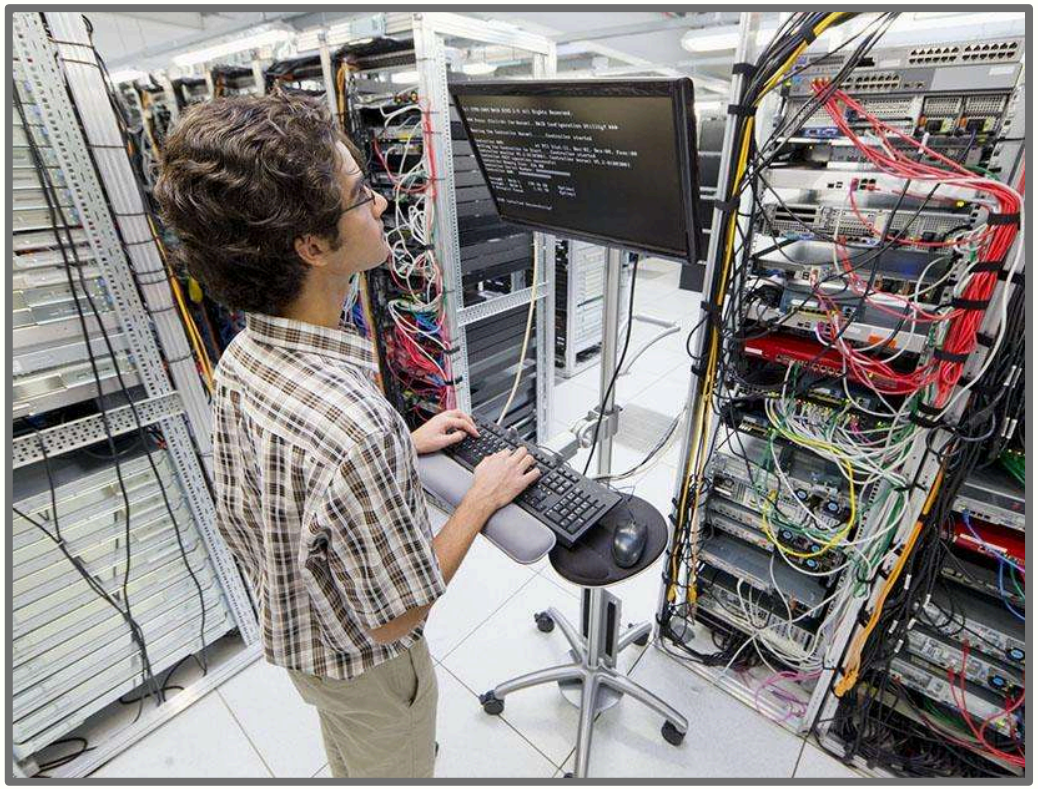
5. 访问日志收集，可以对后端应用异常进行监控

OpenResty ™ | IT大咖说 知识共享平台

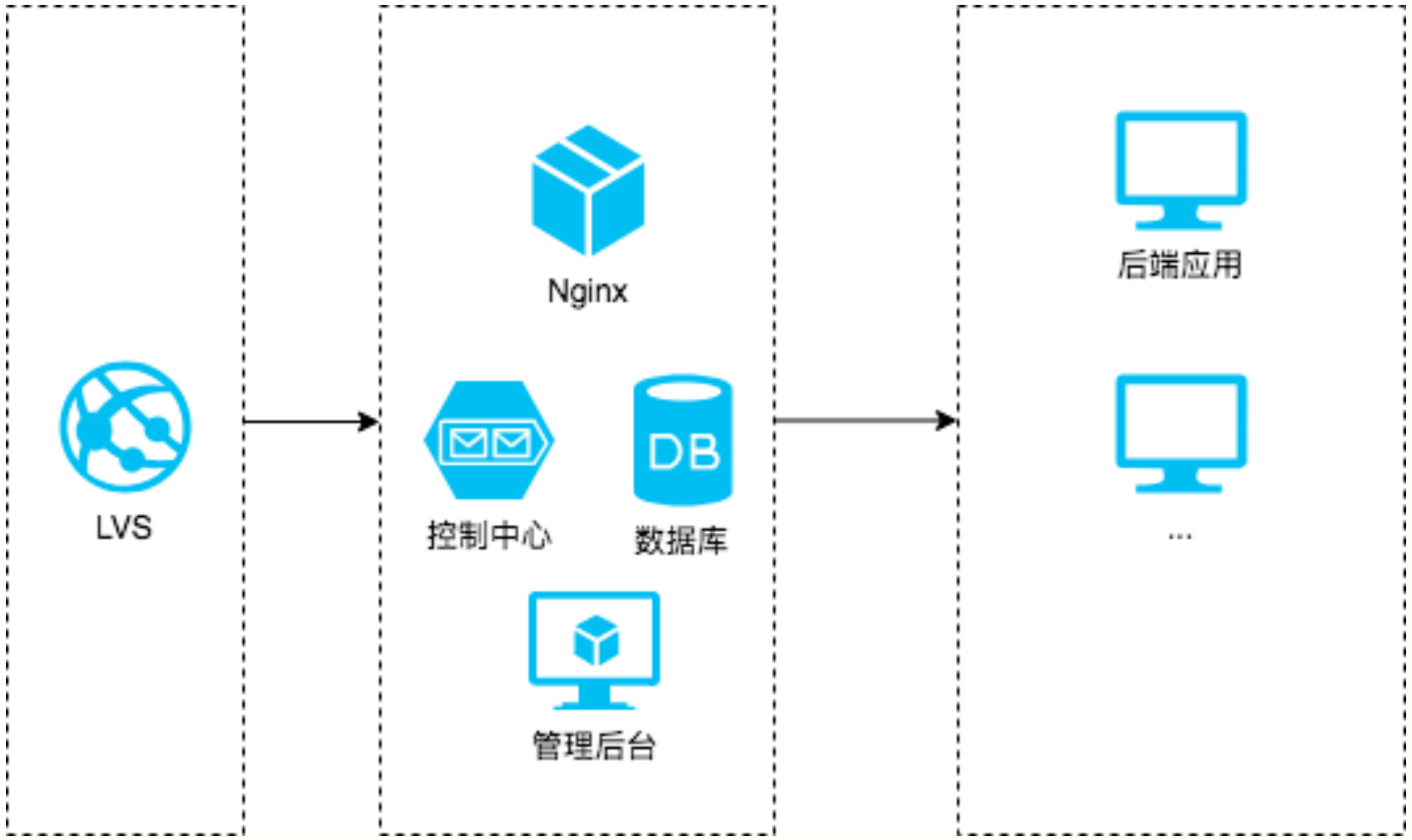
很久以前 ...



日常操作



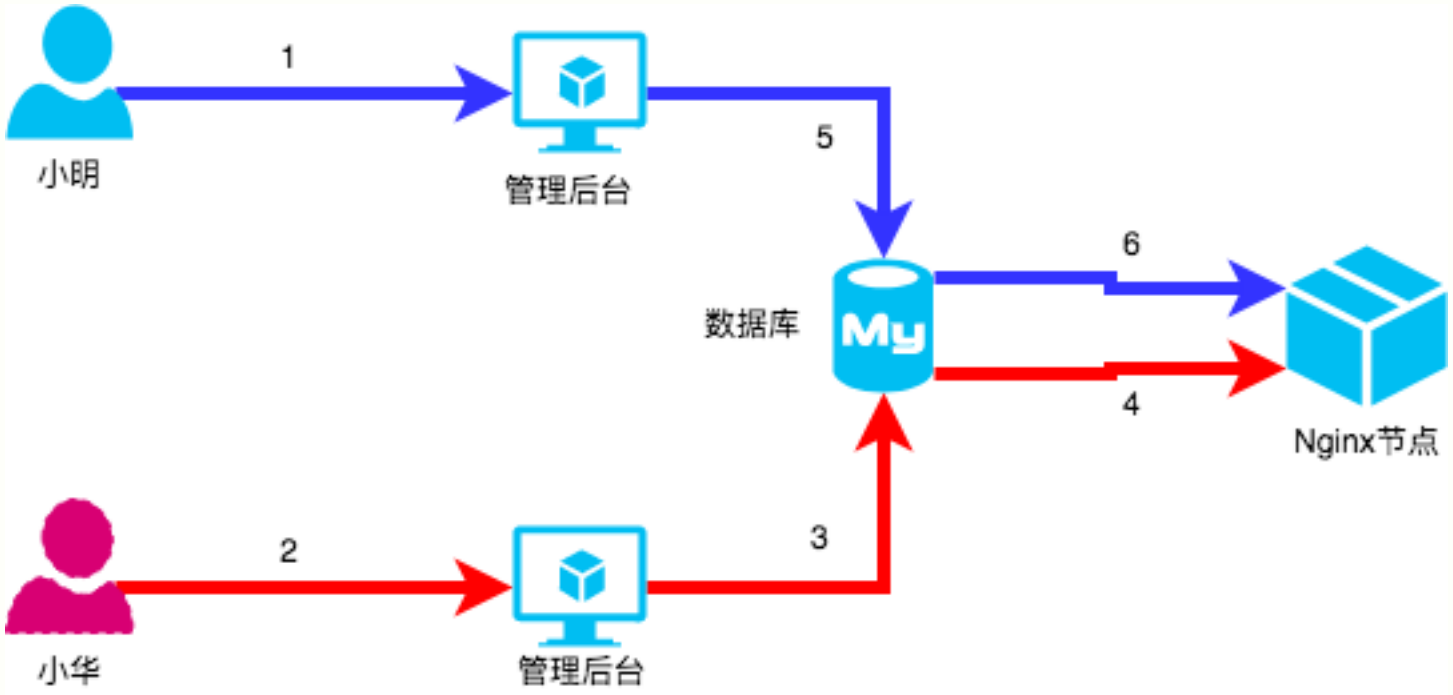
传统的7层负载



- 配置文件
- 多人操作
- 上下负载
- 添加 Location
- 健康检查
- 条件判断
- …

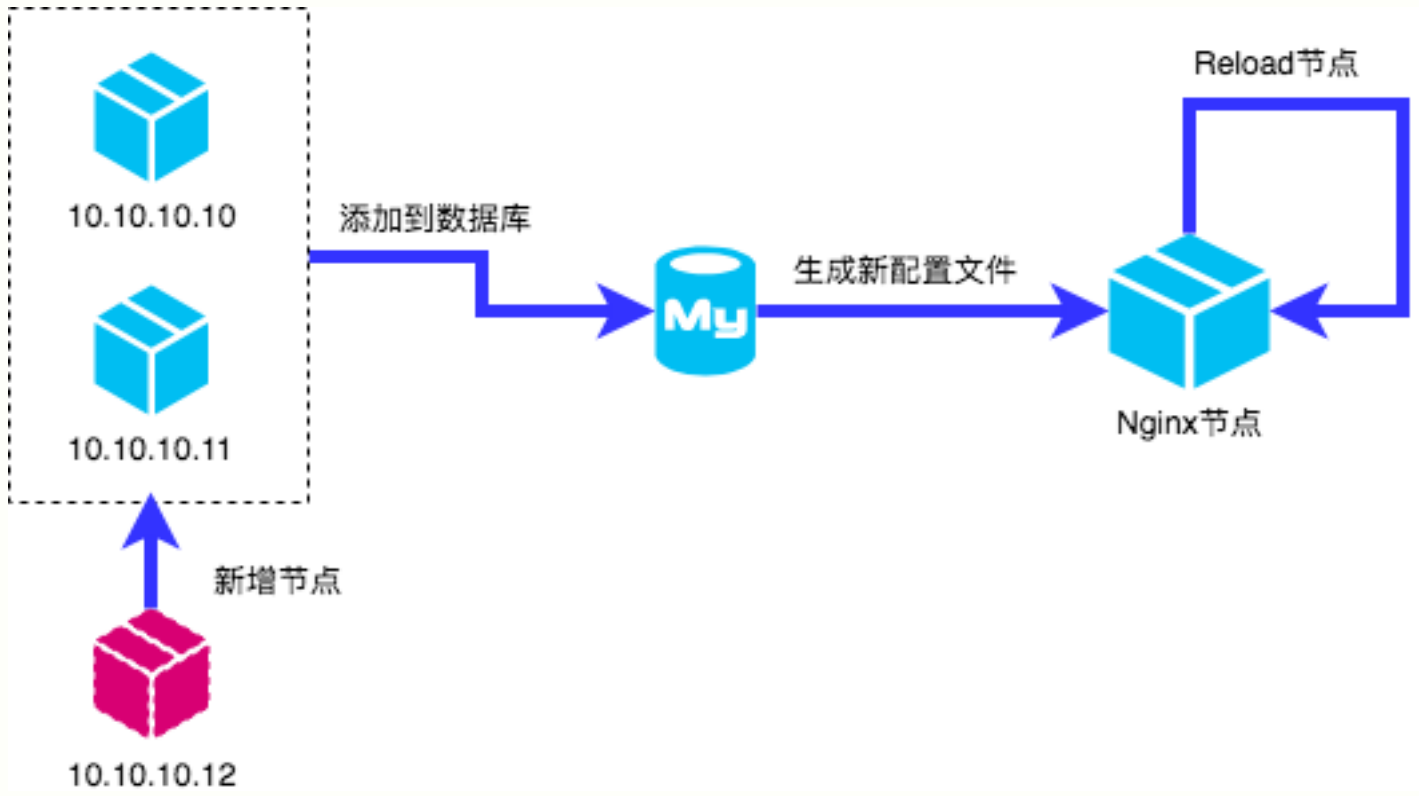
OpenResty™ | IT大咖说 知识共享平台

多人操作



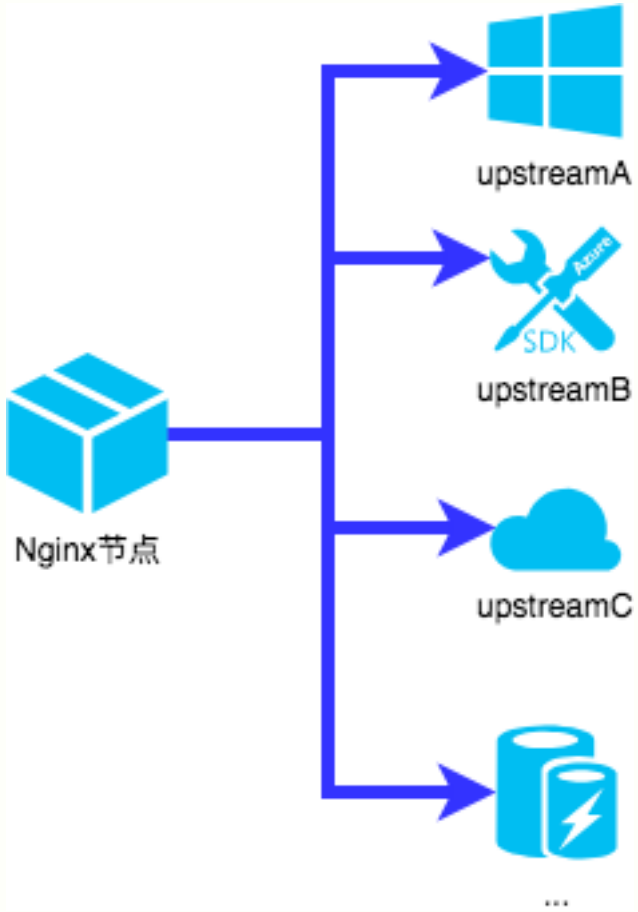
- 增改Location配置
- 配置覆盖
- 锁文件
- Reload

上下负载



- 全量生成配置文件
- Reload节点
- 频繁上下负载

健康检查模块




```
upstream Application{
    server 192.168.0.21:80;
    server 192.168.0.22:80;
    check interval=3000 rise=2 fall=5
    timeout=1000;
}
```

- Nginx集群大，健康检查接口压力
- 应用程序多，Nginx检查压力

条件判断

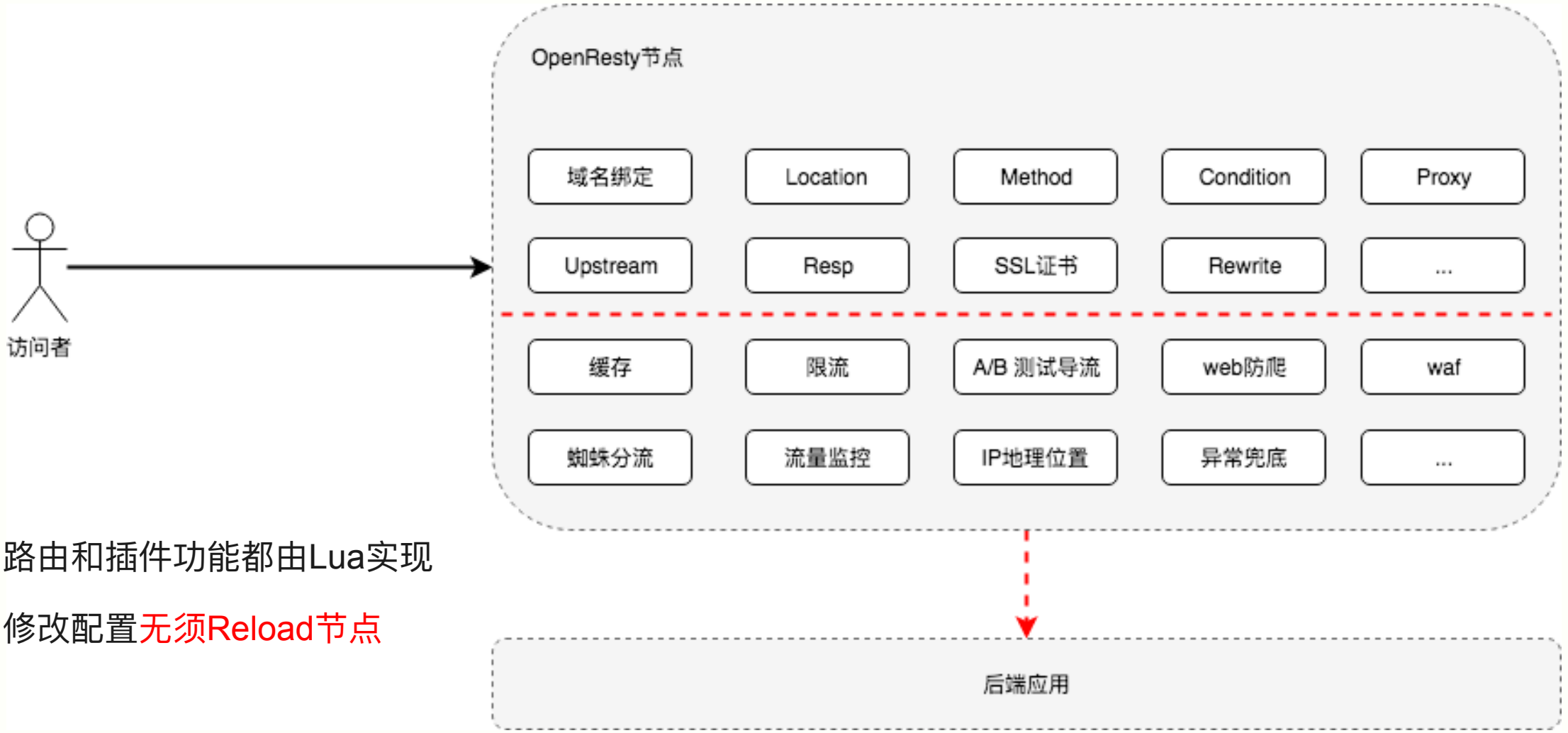
```
location  ~* ^/flight {
      set $air_ly_com    air_ly_com;
      set $NSJ_T2_Leonid    NSJ_T2_Leonid;
      set $NSJ_WAN_apigateway    NSJ_WAN_apigateway;
      #proxy_set_header    HOST air.ly.com;
      if ($http_user_agent ~* "spider") {
      rewrite    ^/flight(.*)$ /flight$1 break;
      proxy_pass http://$air_ly_com;
      }
      if ($http_user_agent ~* "bot") {
      rewrite    ^/flight(.*)$ /flight$1 break;
      proxy_pass http://$air_ly_com;
      }
      if ($scheme = "http") {
      #rewrite ^/flight(.*)$ https://www.ly.com/flight$1 permanent;
      rewrite    ^/(.*)$ https://$host/$1 permanent;
      }
      proxy_pass http://$NSJ_WAN_apigateway;
      access_log    "syslog:user:info:[redacted]" scribe;
}
```

- 条件判断局限性
- 更灵活的条件：

  if 杭州用户

  if method=get && cookie.uid=123 && …

API-Gateway 主要功能

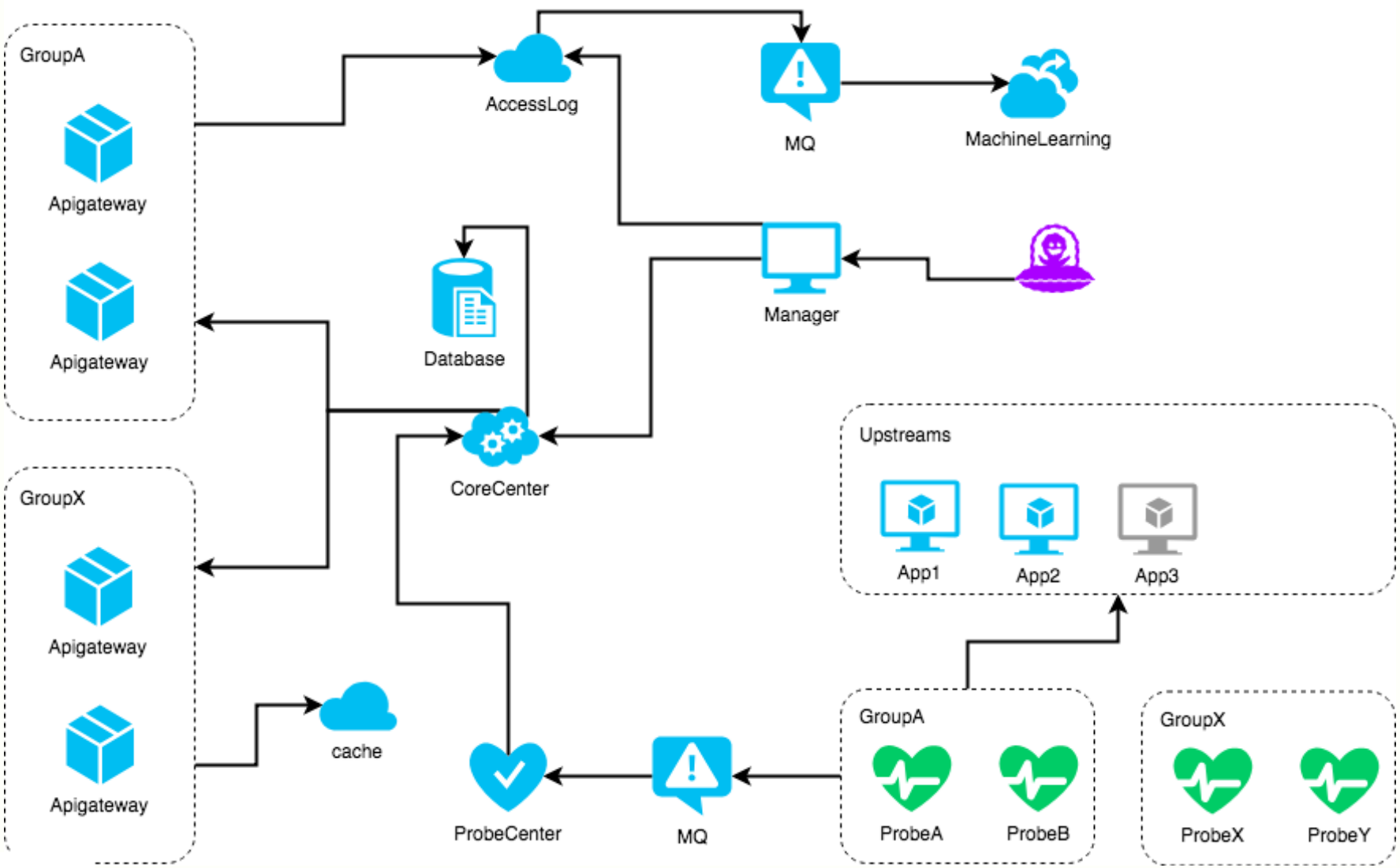


路由和插件功能都由Lua实现

修改配置无须Reload节点

OpenResty™ | IT大咖说 知识共享平台

整个API-Gateway架构

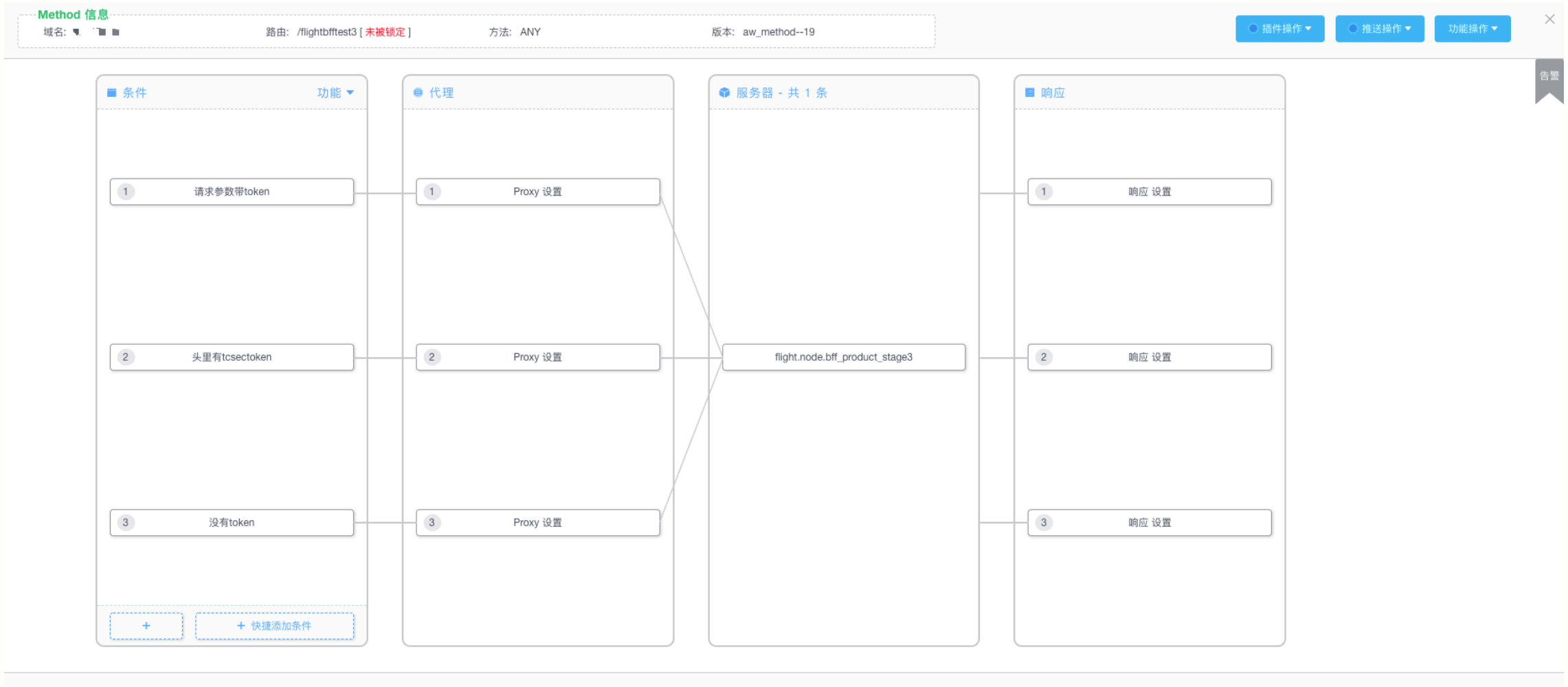


OpenResty ™
IT大咖说 知识共享平台

后台截图-Location配置

OpenResty™ | IT大咖说 知识共享平台

后台截图-铁路图



OpenResty ™
IT大咖说 知识共享平台

乐高积木方式的插件

ApiGateway平台 = 乐高积木底座

各种功能插件就像乐高积木那样，

堆叠在整个平台之上，

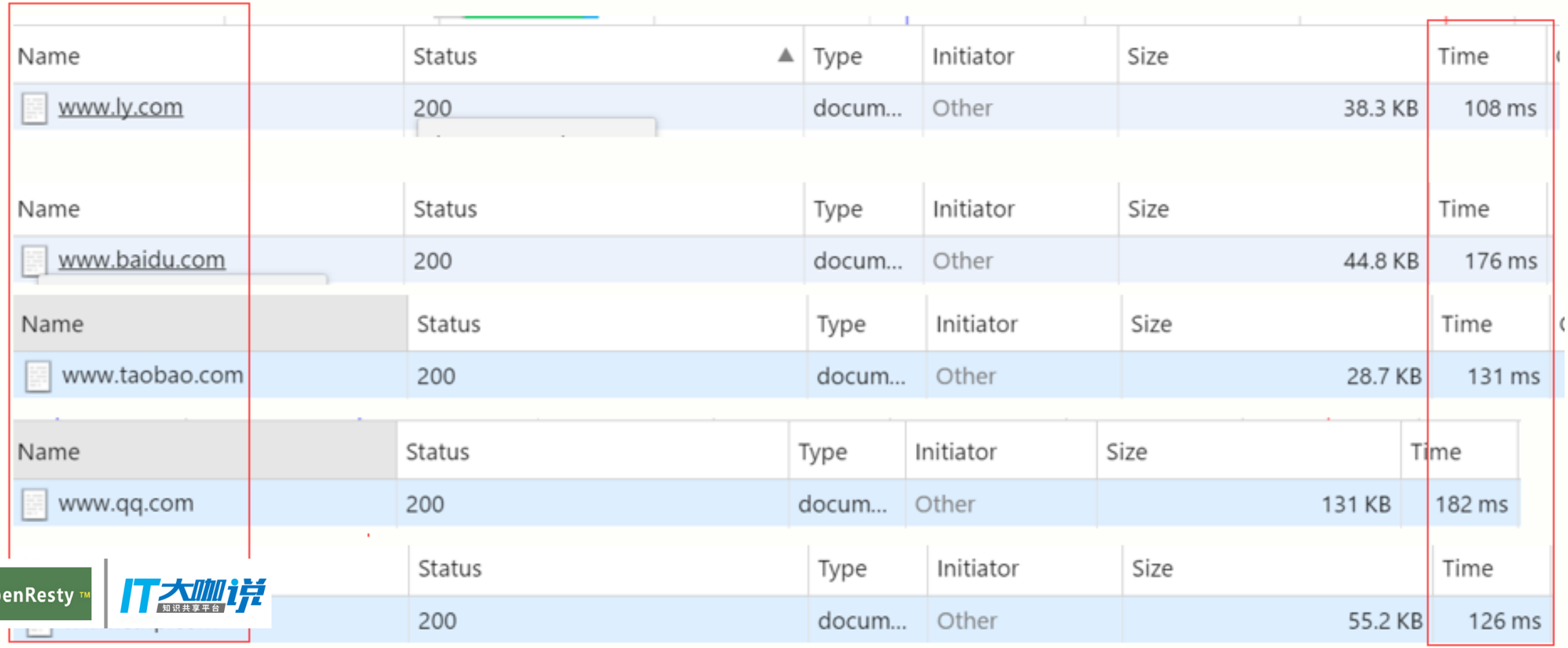
最终完成 作品

OpenResty™ IT大咖说 知识共享平台

高性能内存级缓存

共享内存 shareDict，高吞吐量，低延迟缓存

数以百万的kv缓存，存储管理，批量清除



OpenResty ™ IT大咖说 知识共享平台

秒杀限流

根据后端应用的实际吞吐量，控制每秒透过负载的请求数

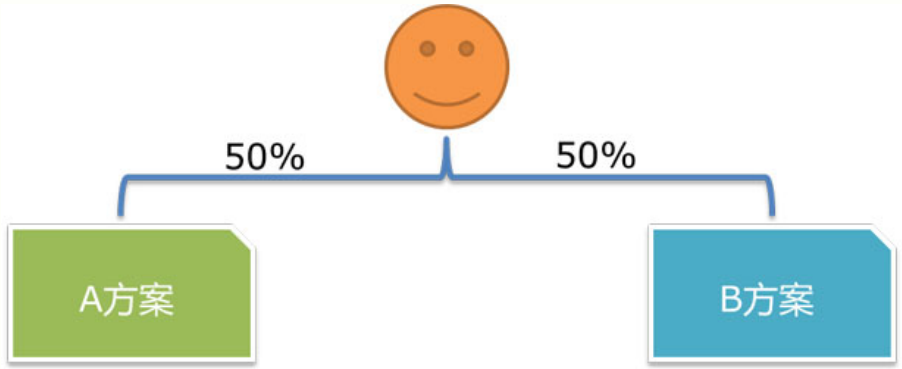
还具备动态限流，自动根据后端应用的响应时间调整限流请求数

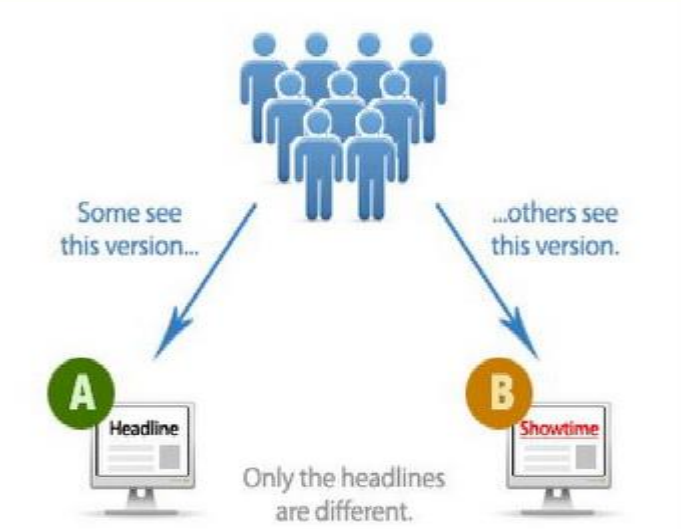
A/B环境测试，流量镜像

参数，cookie，地区，概率 等多维度A/B测试

镜像线上流量到测试应用中，调试程序

高峰期间，每天10亿+的用户访问，提供系统正常灰度发布功能





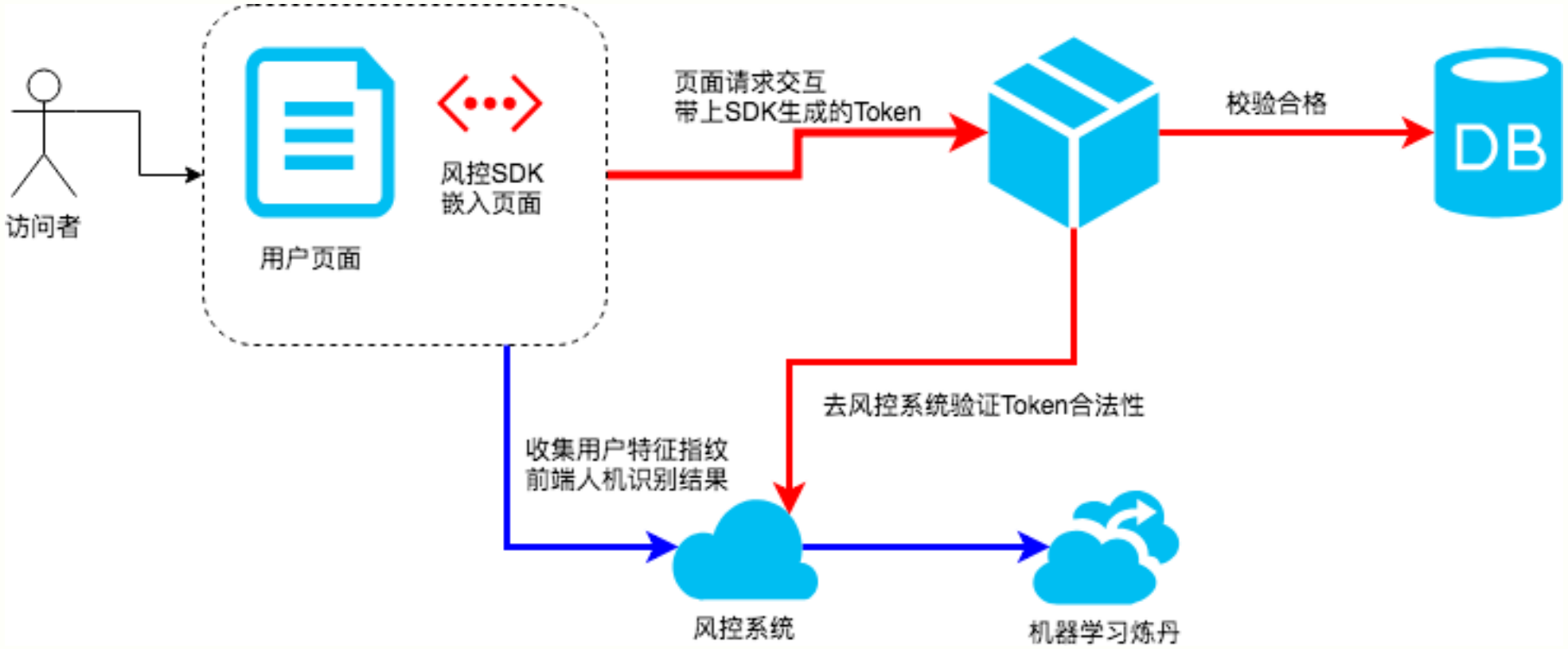
爬虫与对抗

价格数据，抢购秒杀奖品，注册帐号刷新手红包 …

这些非人类正常访问严重影响：公司财力，运营数据，活动效果

OpenResty ™
IT大咖说 知识共享平台

传统风控系统



红色：业务流程主线　　蓝色：异步风控流程

接入繁琐，上线和下线这套系统，对原业务系统流程侵入很大

OpenResty ™ | IT大咖说 知识共享平台

一行注释的接入

```
<!-- apigateway anti sdk -->

<script src="https://js.static.com/apigateway/anti/sdk.min.js"></script>
```

OpenResty ™ | IT大咖说 知识共享平台

一行注释的接入

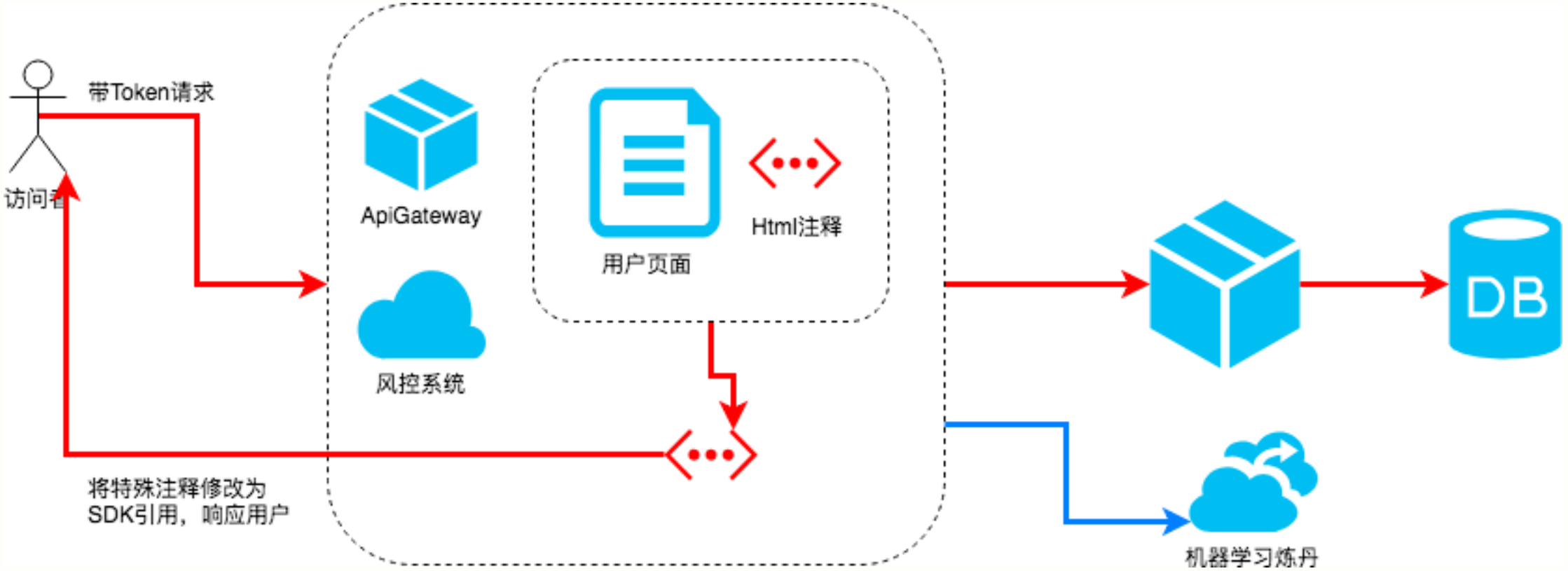
1、接入无侵入性，方便快捷

2、下线防爬也无需改代码，一行注释不影响使用

3、SDK更新不用让业务重新改代码发布

4、没有替换规则的页面直接输出，不会多引用一个无用的JS SDK

ApiGateway 防爬插件



业务系统只需要在需要接入防爬的页面，添加一行特殊的Html注释

对业务系统接入几乎无侵入，上线和下线无任何影响

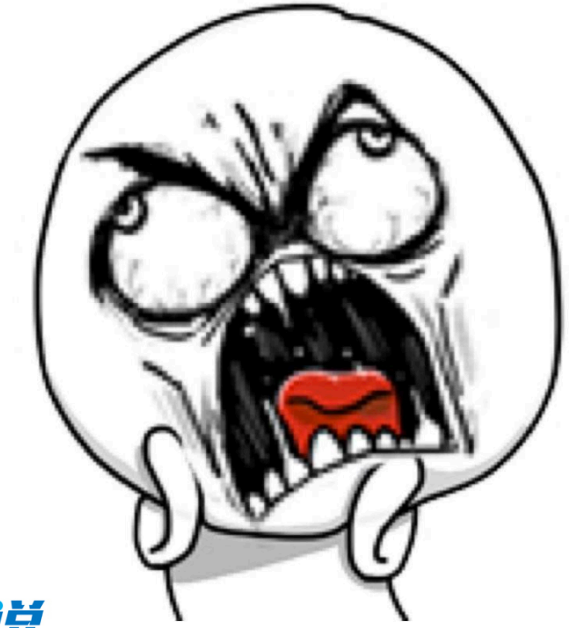
性能优化一例

- 因为是负载系统，所以实际情况，99%的配置，都是路由前缀匹配

```
-- routeRule is defined rules list
local matchItem
for i, item in ipairs(routeRule) do
    local from, to, err = ngx.re.find(ngx.var.uri, item.prefix, 'jo')
    if from == 1 then
        matchItem = item
        break
    end
end
-- todo matchItem
```

性能优化一例

- 压测环境，8cpu虚拟机
- 压测OpenResty，配置文件前缀路由，qps：44000
- 压测Apigateway，Lua实现的前缀路由，qps：42000
- 压测Apigateway，1000个前缀路由，匹配最后一个，qps：4000



难道Lua语言性能不行?

各主流语言 fibonaci(40) 计算耗时

Language	Times	Position
c	0m1.606s	#0
go	0m1.769s	#1
node + cpp module	0m2.216s	#2
luajit	0m2.583s	#3
nodejs	0m5.124s	#4
pypy	0m7.562s	#5
lua	0m34.492s	#6
python	1m11.647s	#7
php	1m28.198s	#8
perl	2m34.658s	#9
ruby 1.9.3	4m40.790s	#10
[illegible]	4m41.942s	#11

斐波那契自然数组：（fibonaci）

1，1，2，3，5，8

前面相邻两项之和，构成了后一项

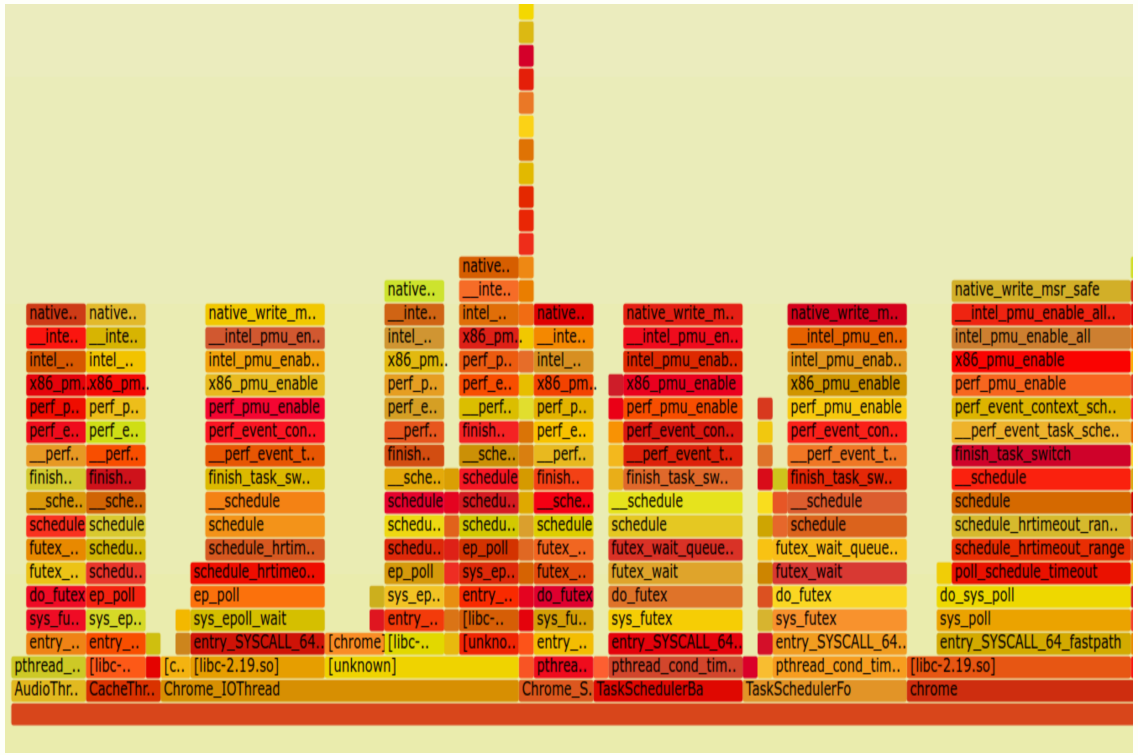
递归计算斐波那契数组

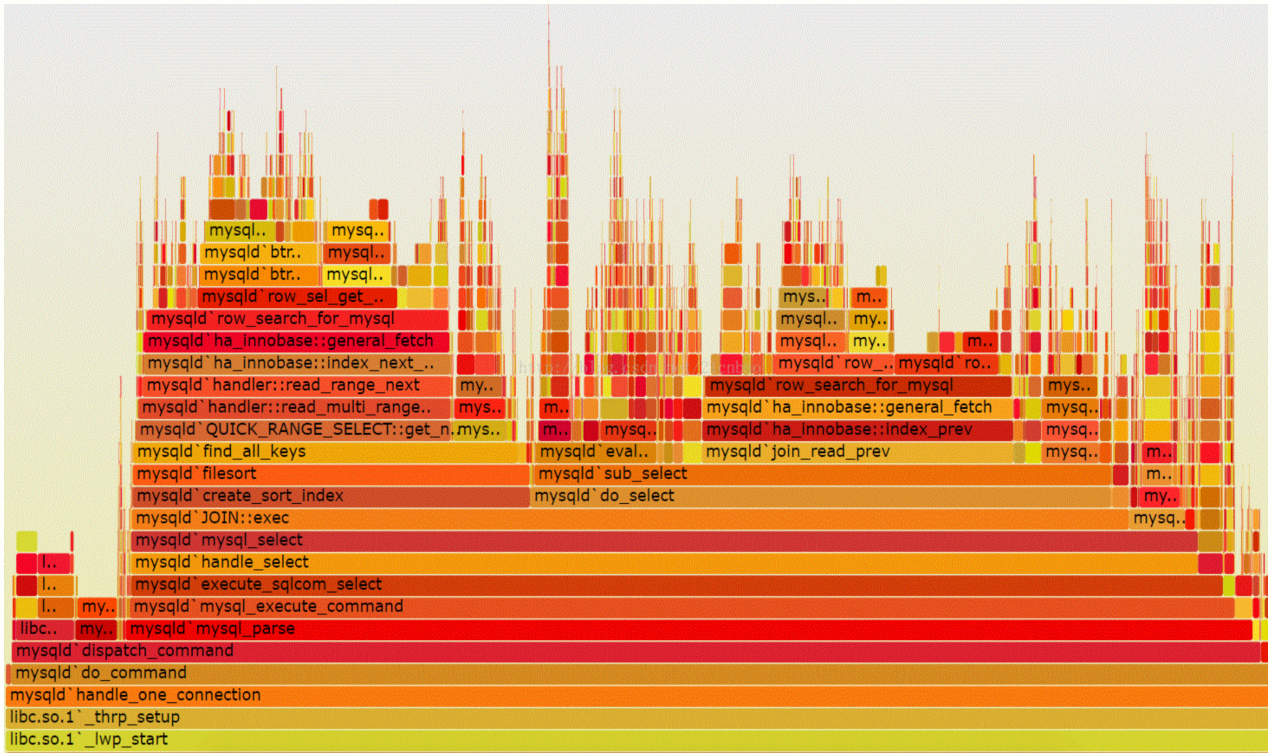
* 一定不是最优的算法

OpenResty ™ IT大咖说 知识共享平台

火焰图找到瓶颈

- 通过火焰图得到结论，性能瓶颈每次1000个前缀路由的前999次匹配上





OpenResty™ | IT大咖说 知识共享平台

tire树解决问题

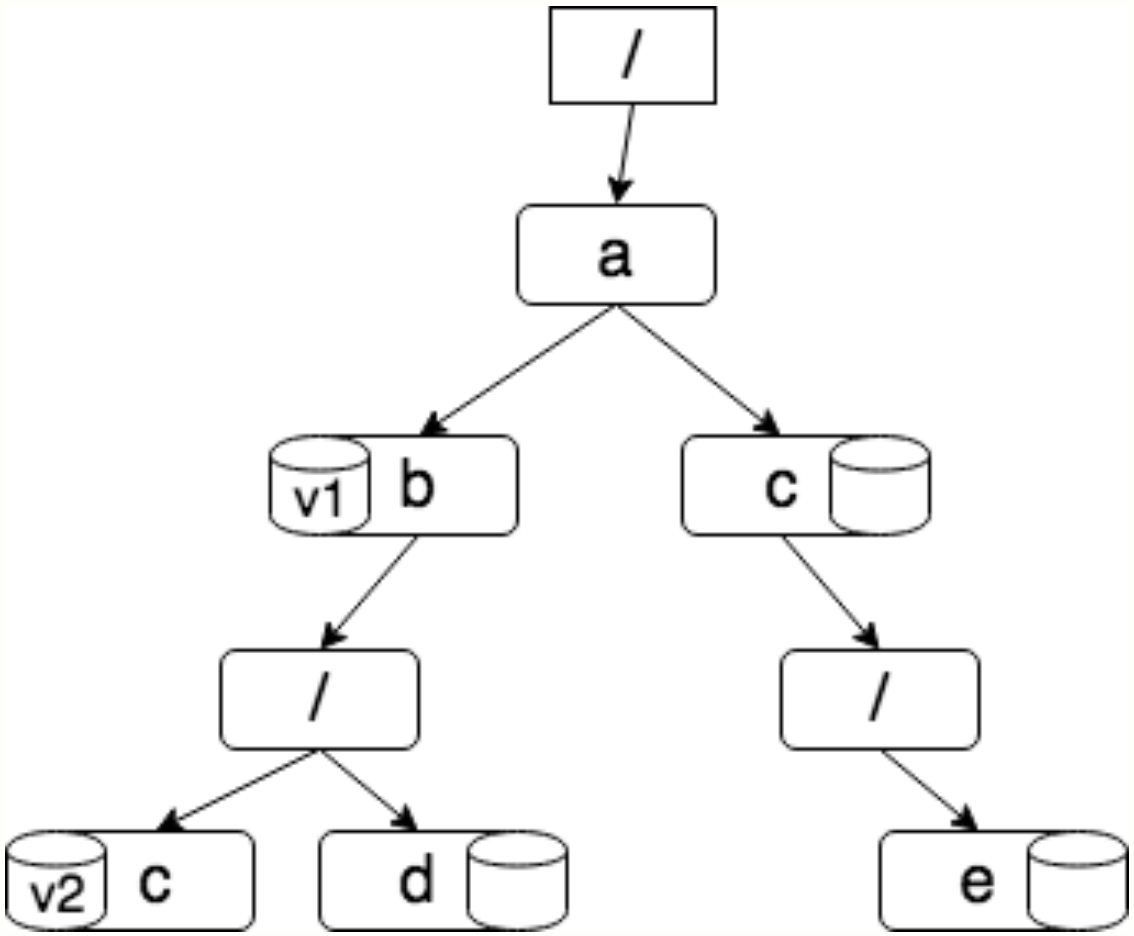
- 定义前缀匹配：

  /ab

  /ab/c

  /ab/d

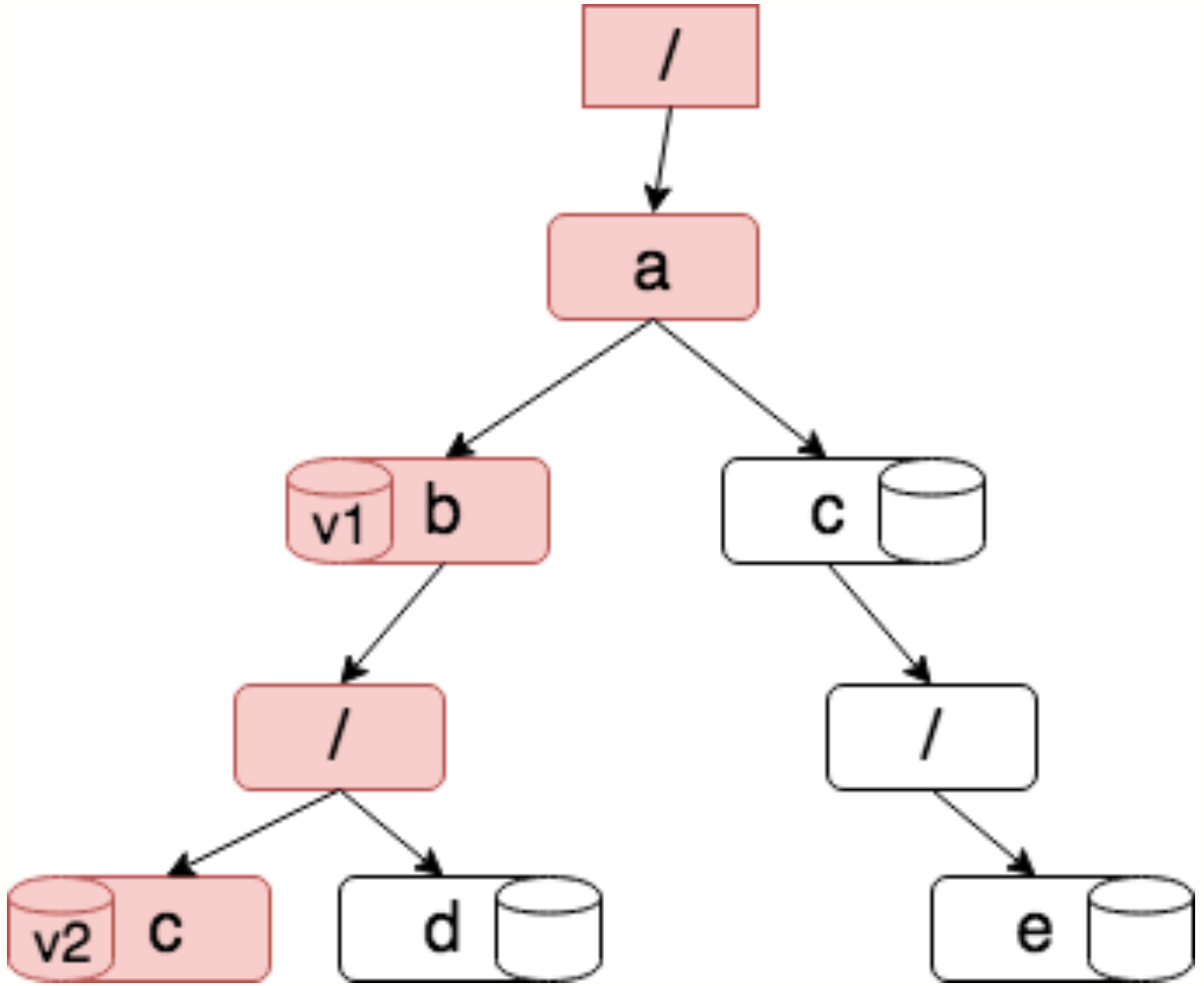
  /ac/e



tire树解决问题

- 实际请求 /ab/c/efg
- 获取叶子结点v1和v2
- 通过tire树算法，我们重新压测

  qps又回到：42000



OpenResty ™
IT大咖说
知识共享平台

QA