

美团HTTP服务治理实践

美团基础架构部 张志桐



个人简介

张志桐

毕业于哈尔滨工业大学。

15年加入美团，目前主要负责

- Oceanus 七层负载均衡网关；
- Mtrace 分布式链路跟踪系统；
- KMS 密钥管理服务

Nginx 的老用户

OpenResty 的新玩家

美团七层负载均衡网关

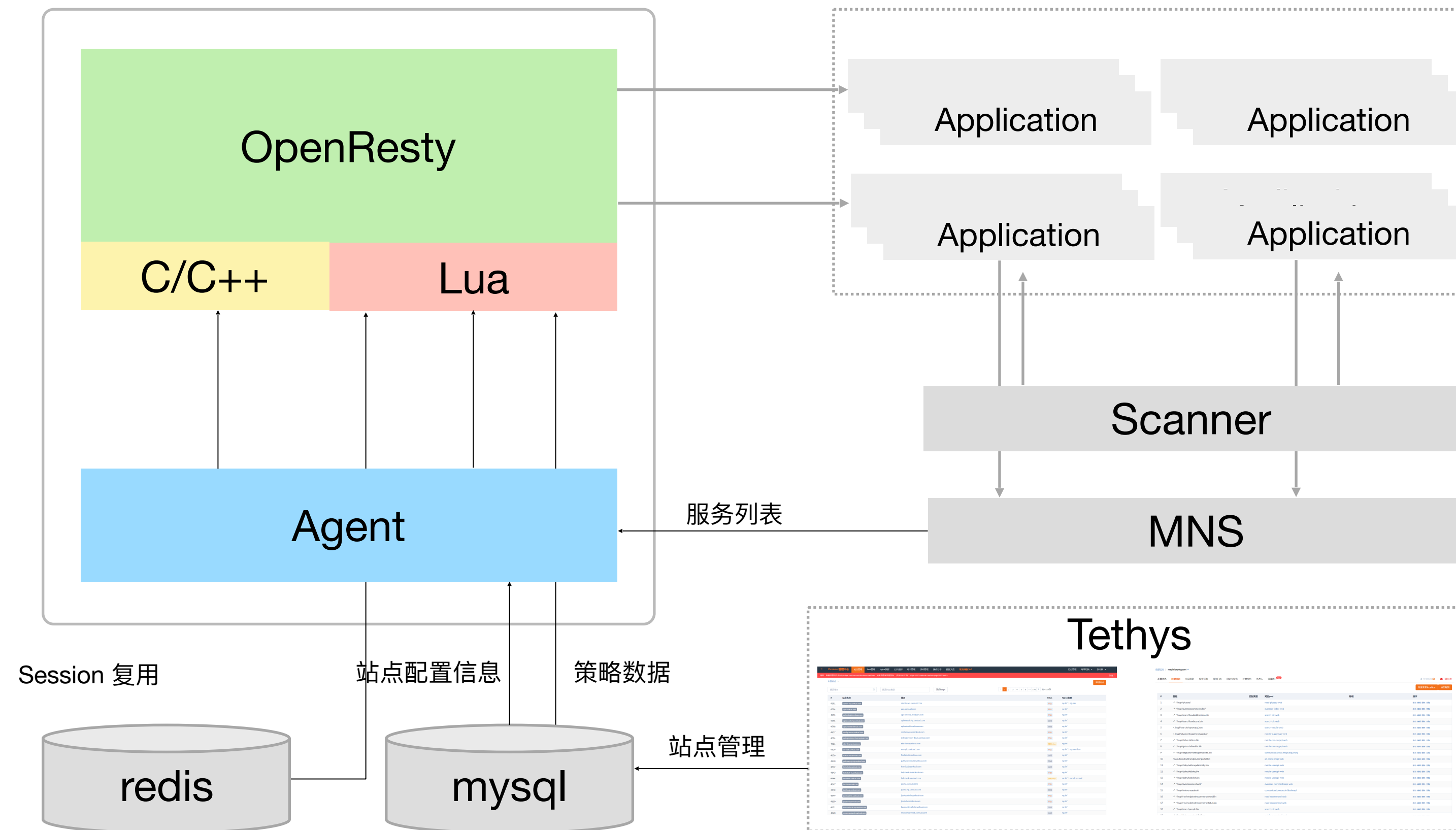
Oceanus

Oceanus 美团七层负载均衡网关

- 每天千亿级别的调用量，几千服务站点，近万个注册服务
- 提供HTTP服务治理能力：服务注册发现、健康检查、可视化管理
- 提供Session复用、动态HTTPS、监控、日志、waf、反爬虫、限流等网关功能

美团七层负载均衡网关

Oceanus 系统架构



proxy_pass

Nginx 配置反向代理

```
location xxx {  
    ...  
    proxy_pass backend;  
}  
upstream backend{  
    server 127.0.0.1:8080;  
    ...  
}
```

问题

- 写死的服务地址
- 需要reload
- 文件配置，很容易出问题

解决方案

Nginx 配置反向代理

问题

- 写死的服务地址
- 需要reload
- 文件配置，很容易出问题

解决方案

- 动态服务注册
- 动态生效
- 结构化管理

服务注册

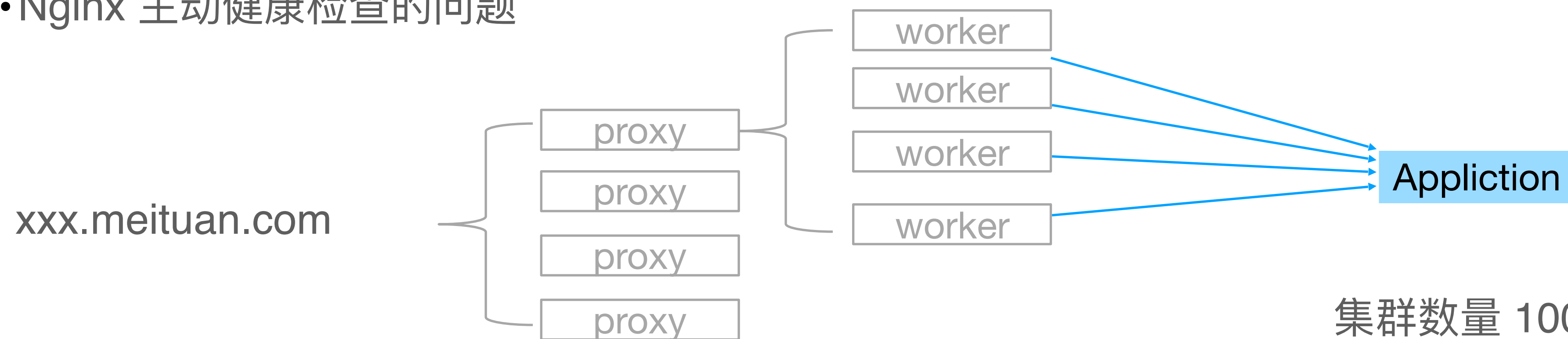
- <https://github.com/Meituan-Dianping/octo-ns/tree/master/nsc>

- 缓存服务信息
- 批量拉取注册服务
- 推拉结合方式保证数据的实时和准确

Scanner

健康检查

- Nginx 主动健康检查的问题



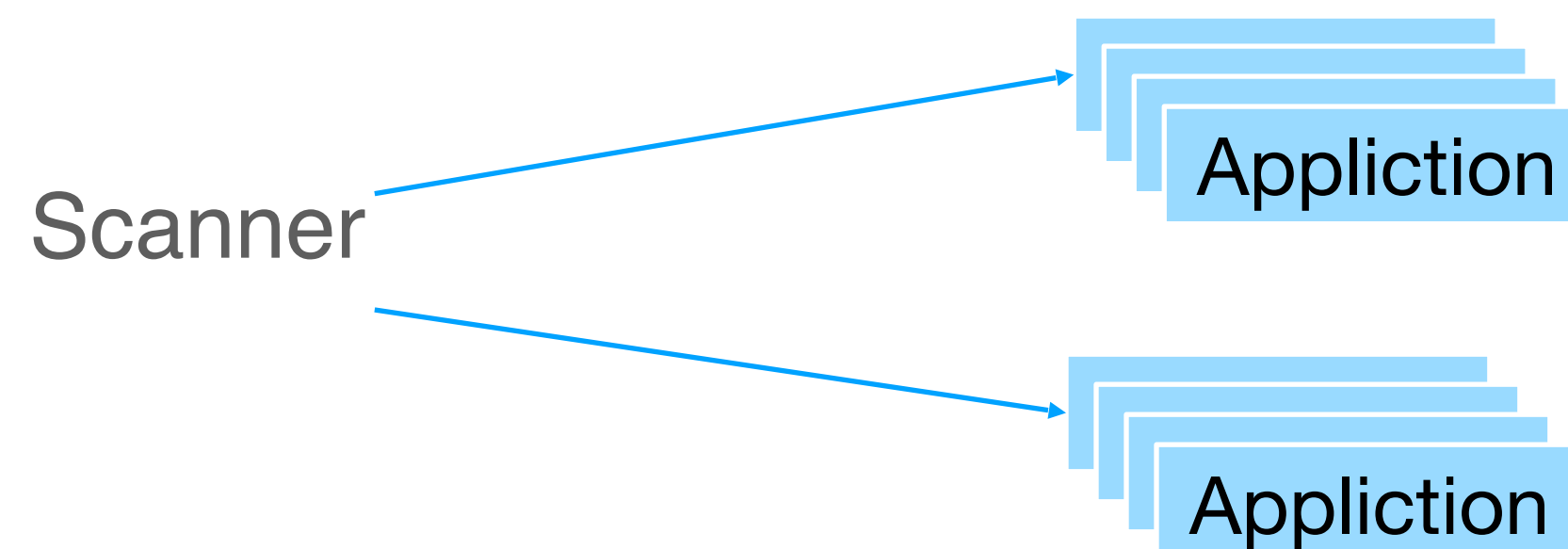
- Scanner 健康检查：周期检查、自定义心跳、快慢线程

<https://github.com/Meituan-Dianping/octo-ns/tree/master/scanner>

Worker数量 32

间隔 1s

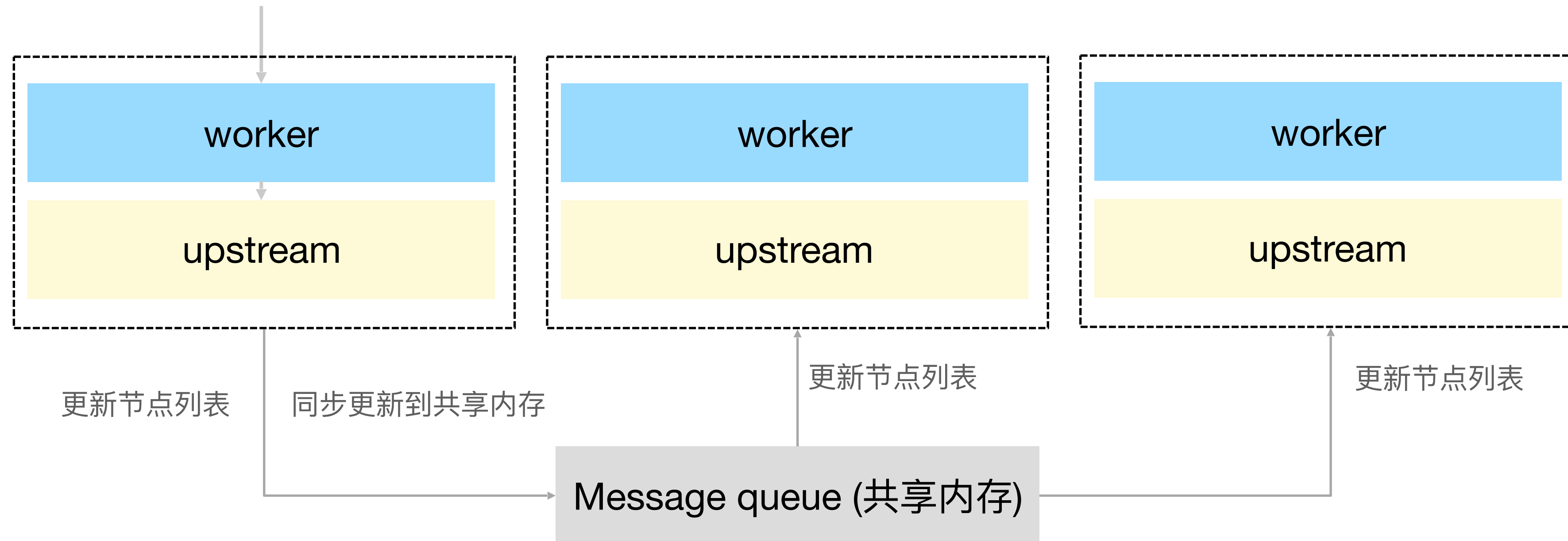
健康检查 QPS $100 * 32$



动态生效

动态upstream

dyups http://engine.taobao.org/document_cn/http_dyups_cn.html
dyups api



持久化

动态upstream

dyups 的问题

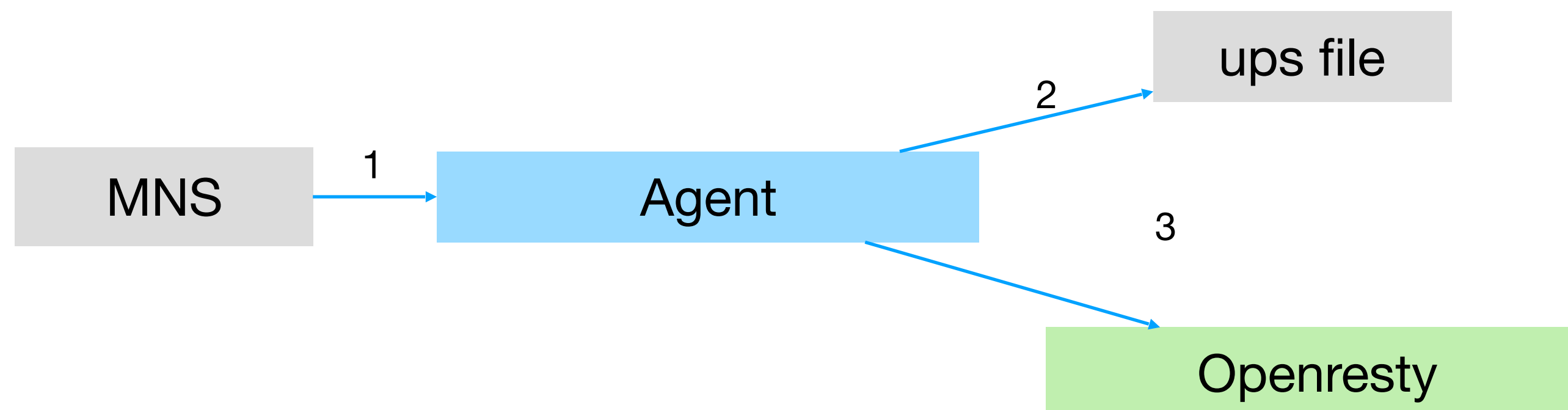
- 内存生效，下次reload丢失

1. Agent 定期感知服务列表变化
2. 落地更新upstream
3. 落地之后调用dyups api

注：reload 和调用dyups api 的并发问题

解决方案

- Agent 托管节点的更新和文件落地



流量倾斜

动态upstream

dyups 的问题

- 每台机器同一时刻更新节点，初始序列是一样的，导致流量倾斜

解决方案

- 随机化轮训的初始节点

```
1380 1380 static ngx_int_t
1381 1381 ngx_dyups_add_server(ngx_http_dyups_srv_conf_t *duscf, ngx_buf_t *buf)
1382 1382 {
1383 1383     ngx_conf_t            cf;
1384 1384     ngx_http_upstream_init_pt init;
1385 1385     ngx_http_upstream_srv_conf_t *uscf;
1386 1386     ngx_http_dyups_upstream_srv_conf_t *dscf;
1387 1387     + ngx_http_upstream_rr_peers_t *peers, *backup;
1388 1388
1389 1389     uscf = duscf->upstream;
1425 1425 @@ -1425,6 +1426,14 @@ ngx_dyups_add_server(ngx_http_dyups_srv_conf_t *duscf, ngx_buf_t *buf)
1426 1426     if (init(&cf, uscf) != NGX_OK) {
1427 1427         return NGX_ERROR;
1428 1428     }
1429 1429 +
1430 1430 + /*add init_number initialization*/
1431 1431 + peers = uscf->peer.data;
1432 1432 + peers->init_number = ngx_random() % peers->number;
1433 1433 + backup = peers->next;
1434 1434 + if (backup) {
1435 1435 +     backup->init_number = ngx_random() % backup->number;
1436 1436 + }
1428 1428
1429 1429     dscf = uscf->srv_conf[ngx_http_dyups_module.ctx_index];
1430 1430     dscf->init = uscf->peer.init;
```

dyups添加服务接口

duscf为dyups中更新的upstream

取出upstream赋值给uscf

通过一个随机值去打乱uscf里面数据的顺序

Ocaenus Tethys

结构化~~管理~~

创建站点

- 平台创建
- 文件导入

全部站点 > 新建站点

配置信息 导入配置

站点名称 ? *

添加域名 ?

+

添加ngx集群 ?

+

开启https

关闭 开启 强制http 强制https 仅https

ssl证书 ?

ssl端口

443

提交

Ocaenus Tethys

结构化

管理

关联服务

- 配置规则
- 校验规则

Oceanus管理中心

站点管理

Pool管理

Nginx集群

公共规则

证书管理

发布管理

操作日志

数据大盘

全部站点

 >

tethys.inf.dev.sankuai.com

▼

配置信息

映射规则

公用规则

发布预览

操作日志

自定义发布

灰度发布

负责人

流量统计

new

#	路径	匹配类型	对应pool
1	/sso/callback		com.sankuai.oceanus.tethys
2	/api/		com.sankuai.oceanus.tethys
3	/		com.sankuai.oceanus.tethyspage
4	/test		com.sankuai.oceanus.tethys

结构化管理

指令配置

设置Header、设置超时、rewrite、自定义指令等

全部站点 > tethys.05.inf.dev.sankuai.com

配置信息

映射规则

公用规则

发布预览

操作日志

自定义发布

灰度发布

负责人

流量统计

返回全部

编辑

名称

test

类型

前置规则

后置规则

添加指令

proxy_pass

ifelse

rewrite

return

static-resource

access_log

custom

more_clear_headers

more_set_headers

验签

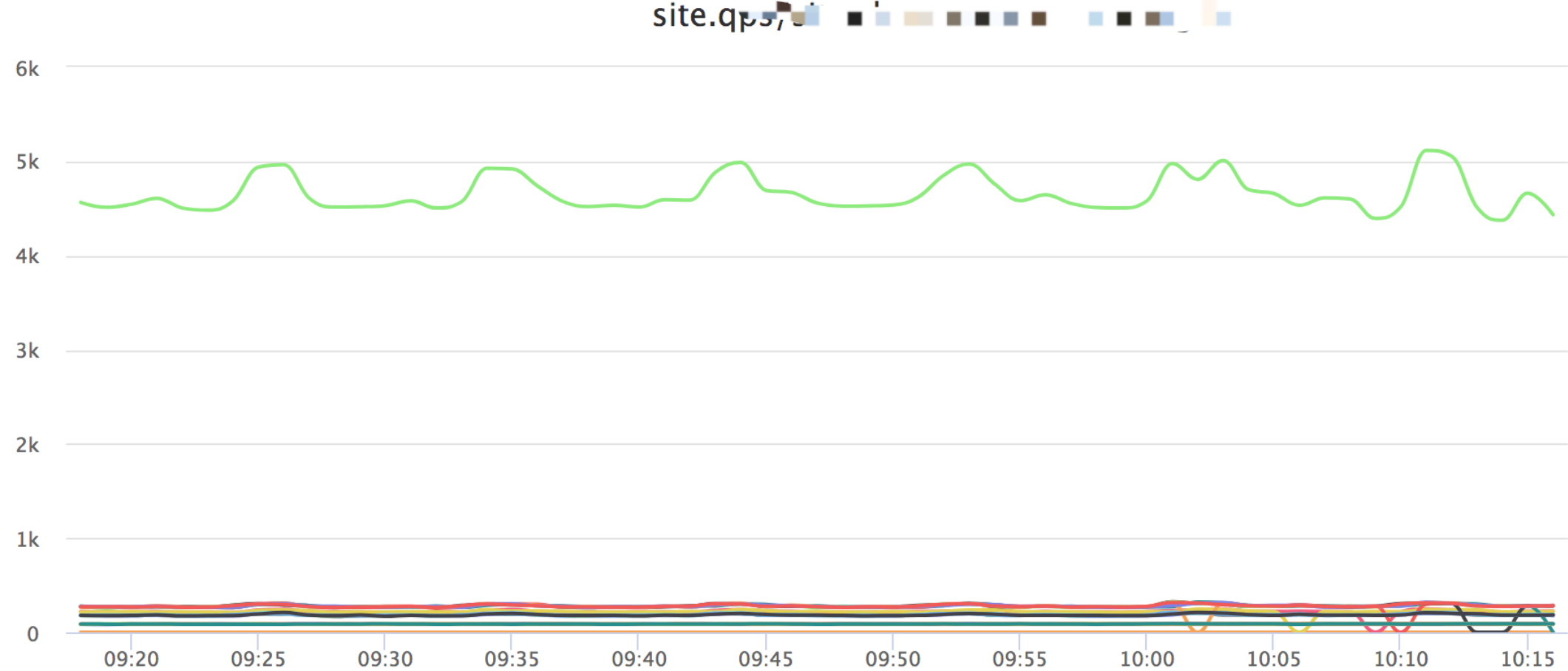
反爬虫

安全伞

指令列表

流量统计

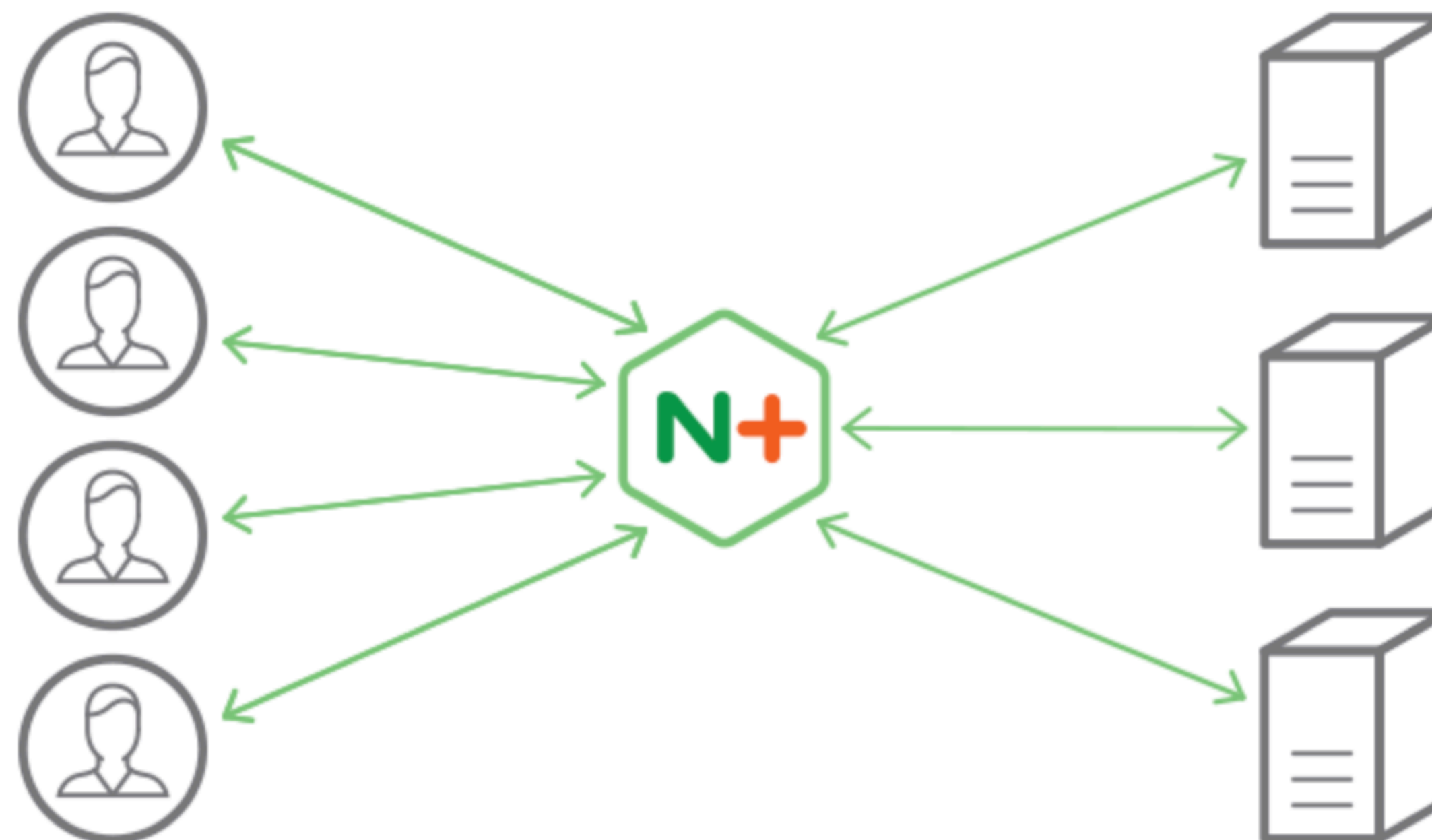
QPS、HTTPS QPS、4XX、5XX



目录

负载均衡

- 精细化分流
- 泳道
- 单元化



负载均衡

精细化分流

对某个地域的用户灰度新的功能特性

按百分比引流

基于流量特征的后端服务物理隔离

配置信息

节点管理

相关站点

报文修改策略

分流策略

返回全部

编辑

提交

取消

策略名

分流信息

header

输入分流字段

Q

分流条件

新增

删除

等于

不等于

大于

大于等于

小于

小于等于

value

请输入内容

lvalue

请输入内容

与

或

分流分组

分流开关

打开

关闭

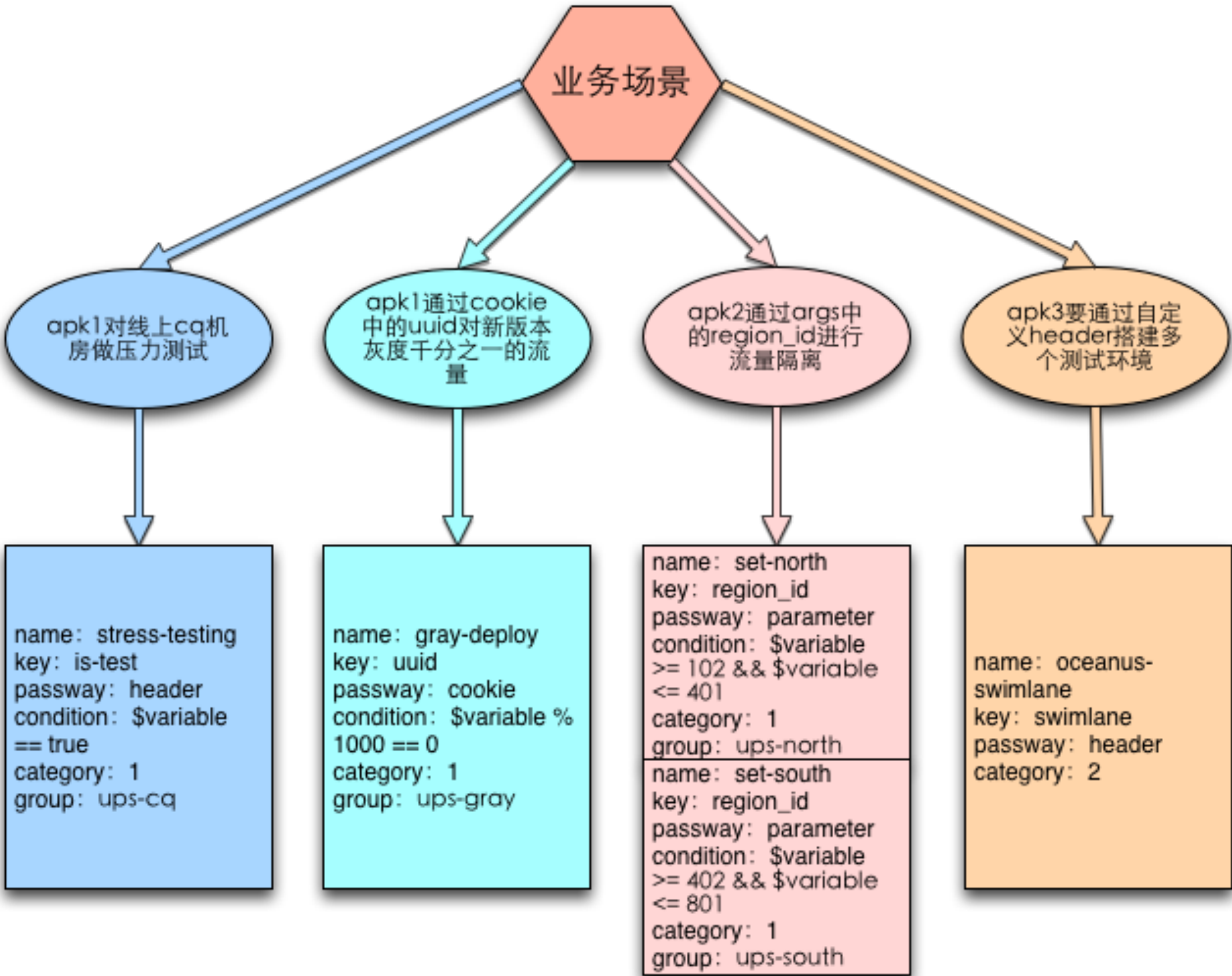
fallback路由

打开

关闭

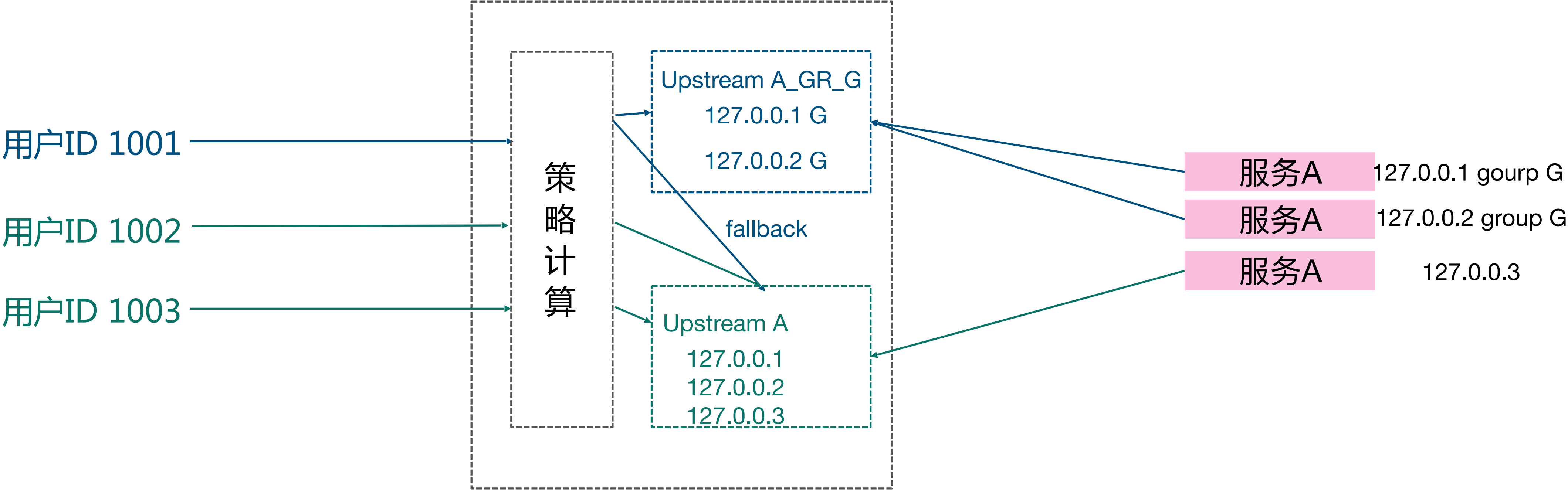
策略描述

请输入内容



负载均衡

精细化分流



用户ID mod 100 = 1 的请求路由到灰度分组G

策略拉取

精细化分流

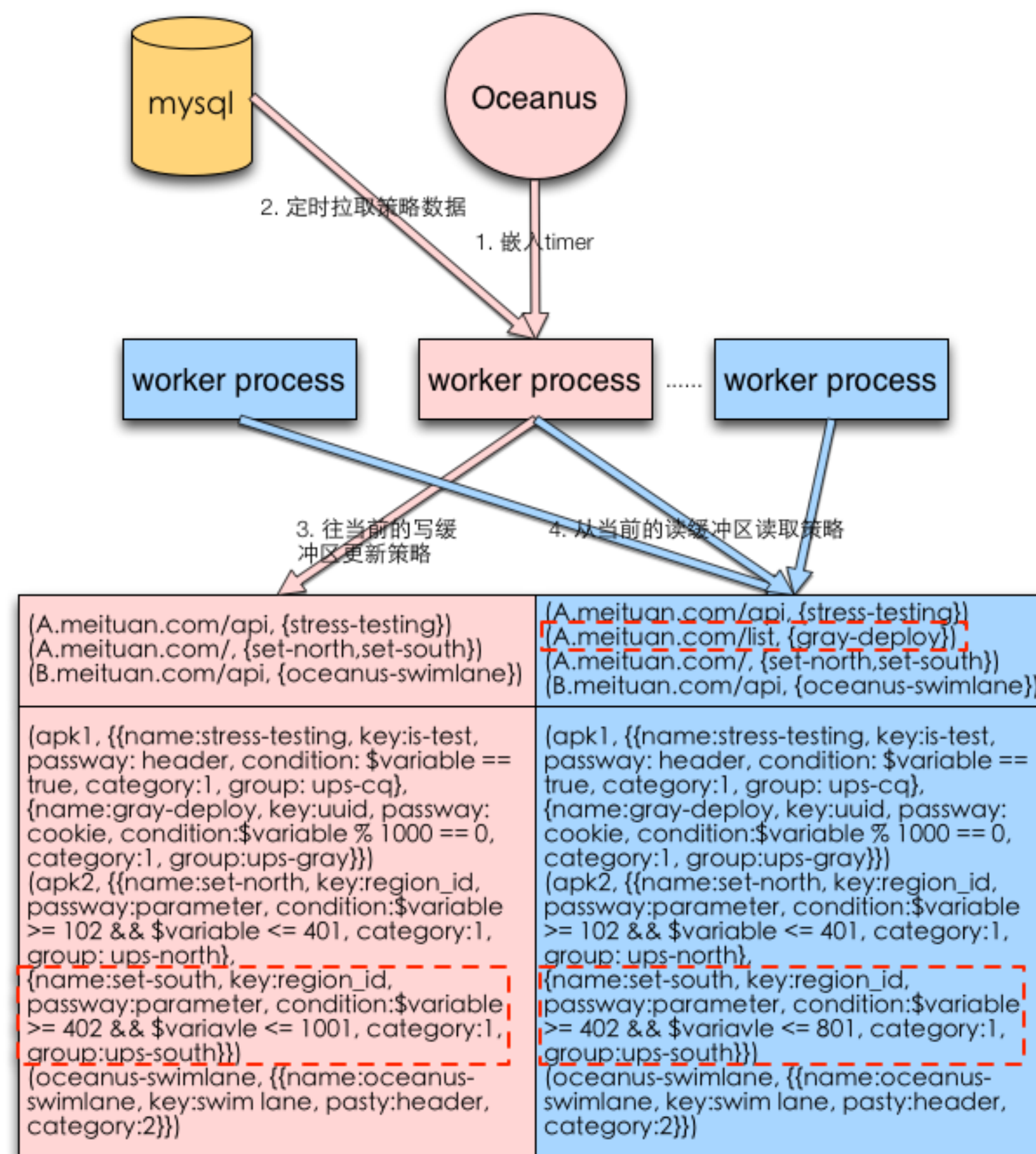
一个worker进程嵌入timer，拉取策略配置

DB配置结构化写入共享内存的双buffer

worker处理请求的时候，从共享内存中读取策略

策略生效粒度：Host + Location + appkey

私有策略纬度：appkey



策略查询
精细化分流

获取请求上下文

根据host + location 查找使用的策略集合

公共策略直接生效；私有策略按appkey查找

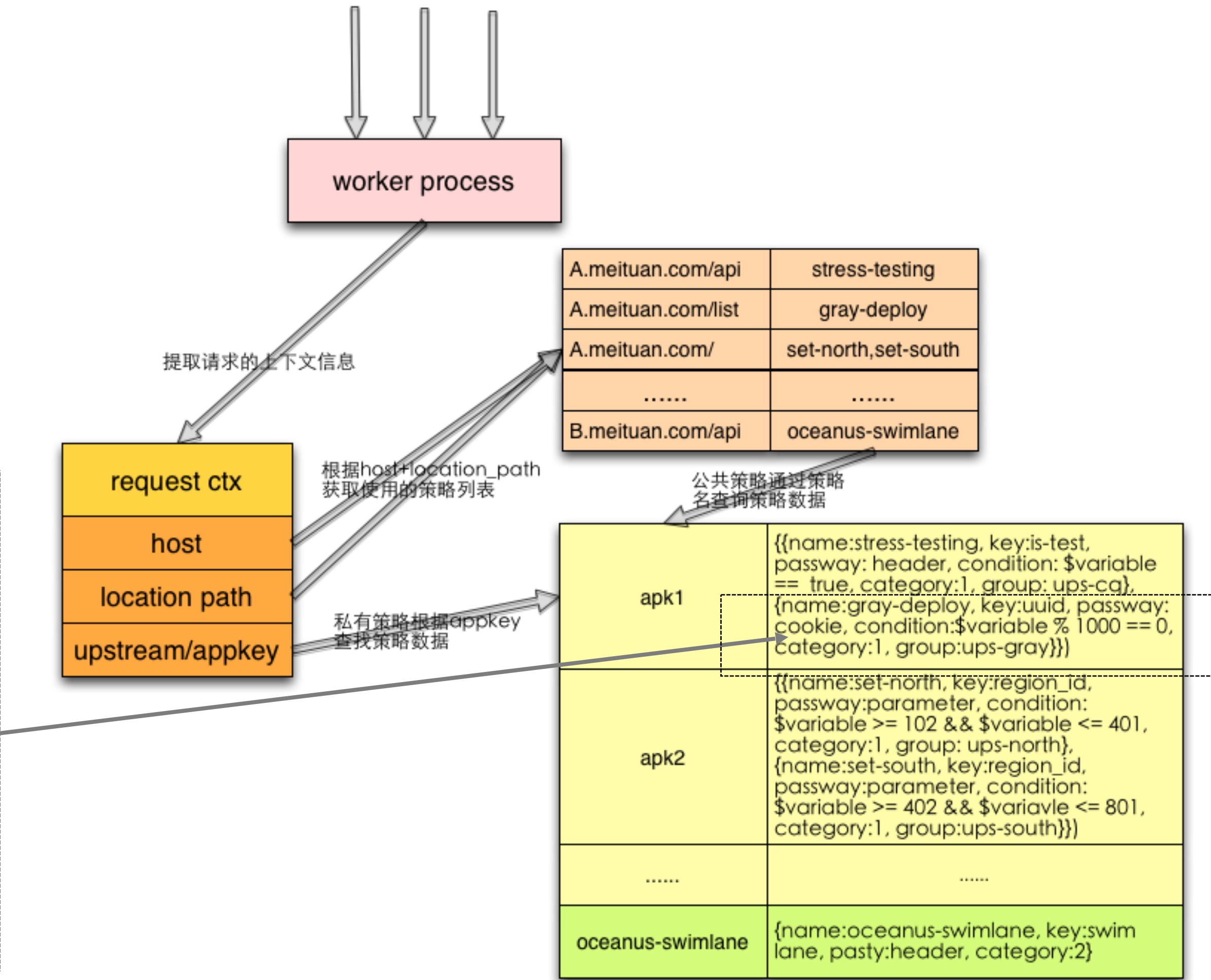
Apk1

Name gray-deploy

Key:uuid, passway: cookie

Condition: \$variable % 1000 == 0

Group ups-gray



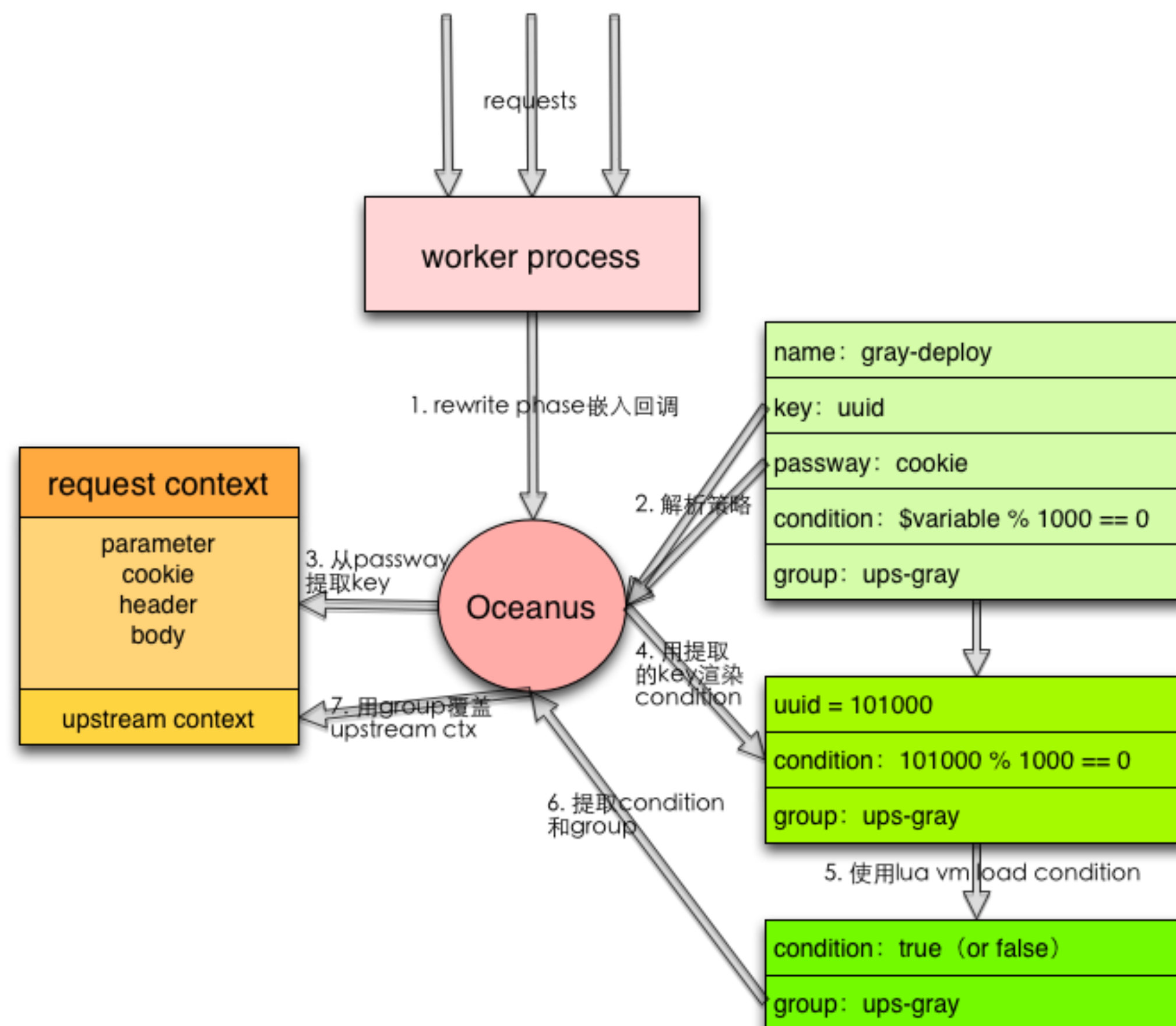
负载均衡

精细化分流

1. rewrite 阶段触发
2. 解析策略数据
3. 动态获取配置的请求参数
4. 通过参数和配置的表达式计算是否命中分流策略
5. 如果命中则改写路由的ups到指定的ups group,

否则不做修改

```
if (ngx.var.xxx % 1000 == 1)
    ups = ups + target_group;
```

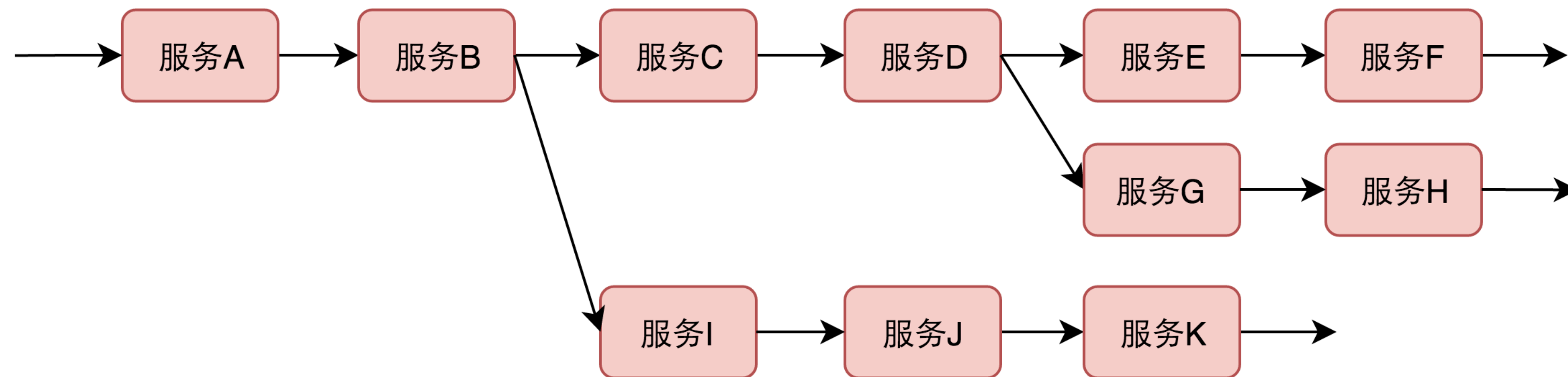


负载均衡

泳道

微服务框架下服务个数多、调用链路较长，其中一个服务出问题会影响到整条链路

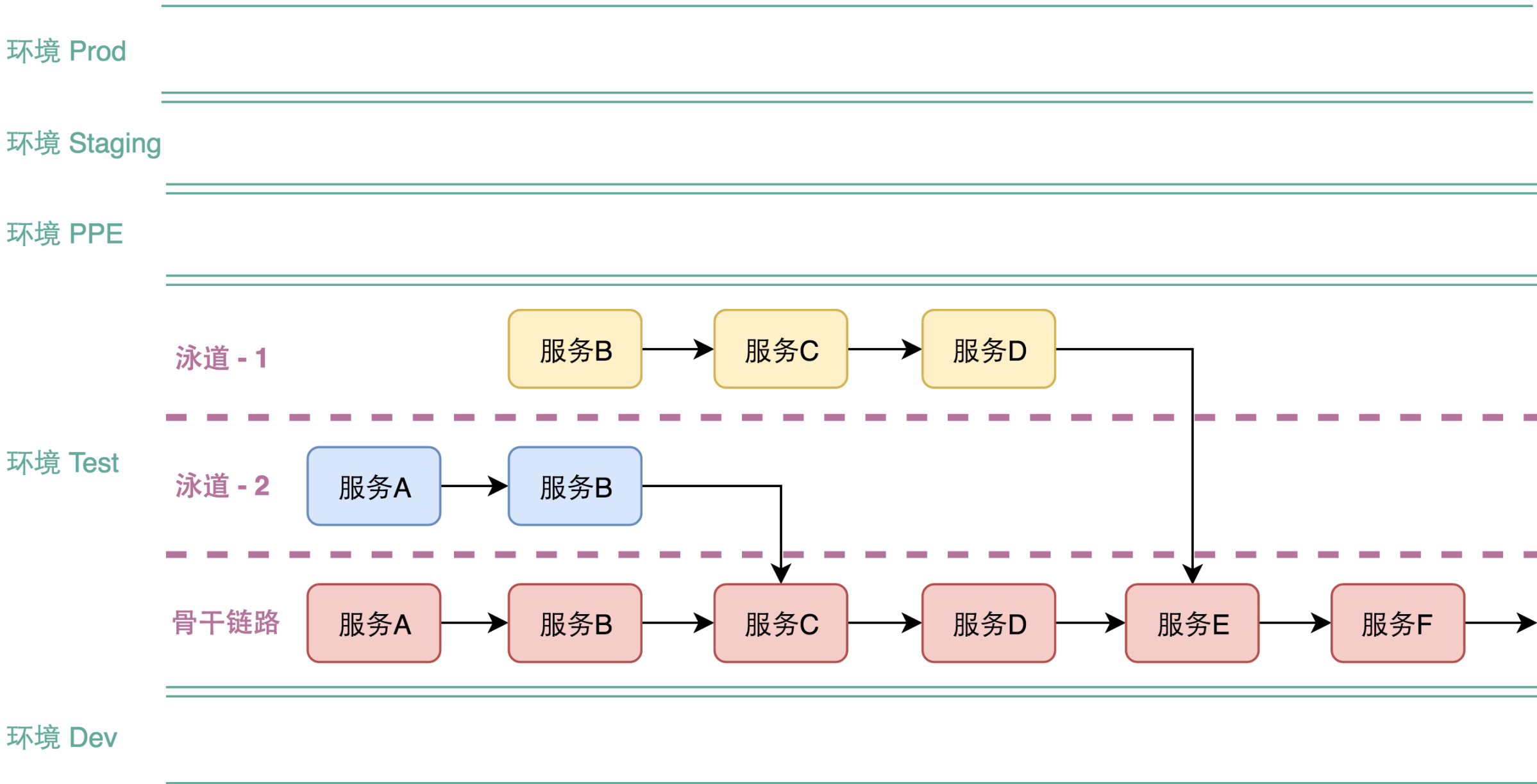
QA提测往往需要该条链路上的多个服务配套测试，甚至是同时测试一个服务的多个演进版本



负载均衡

泳道

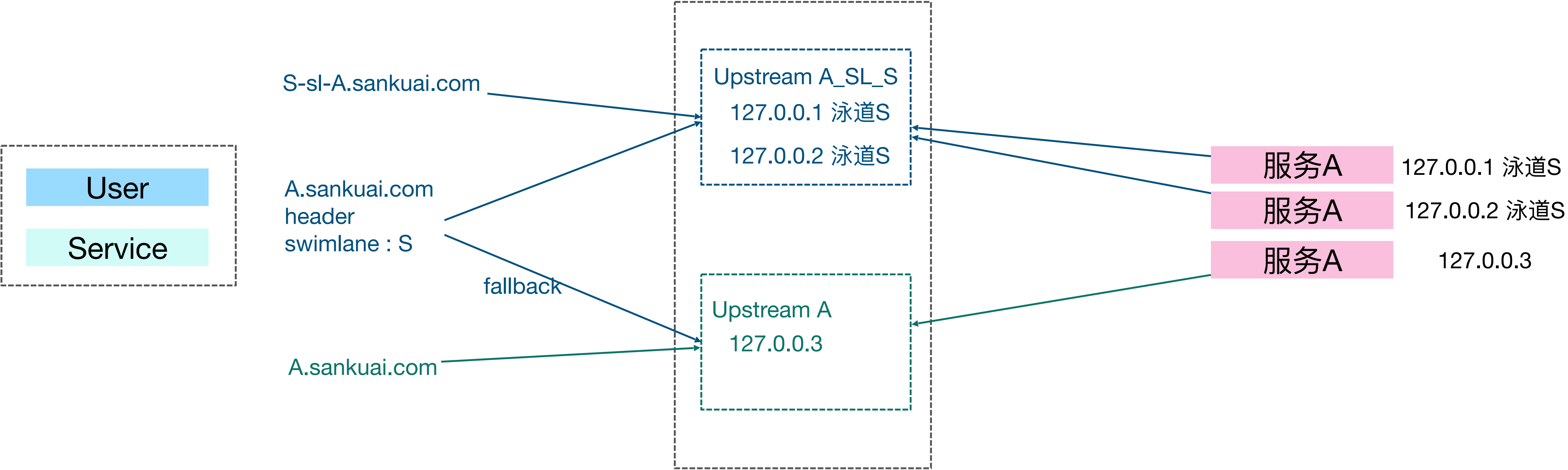
- ✓ 骨干链路稳定
- ✓ 多泳道并存：可保证多服务，多版本并行测试
- ✓ 错误隔离：泳道内服务异常，不会影响其他泳道



处于[泳道-1]内的服务B要调用服务C，若在[泳道-1]内部署了C服务，则B只能调用泳道内的C服务，而不能调用骨干链路或其他泳道的C服务；若在[泳道-1]内没有部署C服务，则流量会调回骨干链路

负载均衡

泳道



Lua 泳道模块判断Host 和 Header 选择转发到不同的ups

负载均衡

单元化

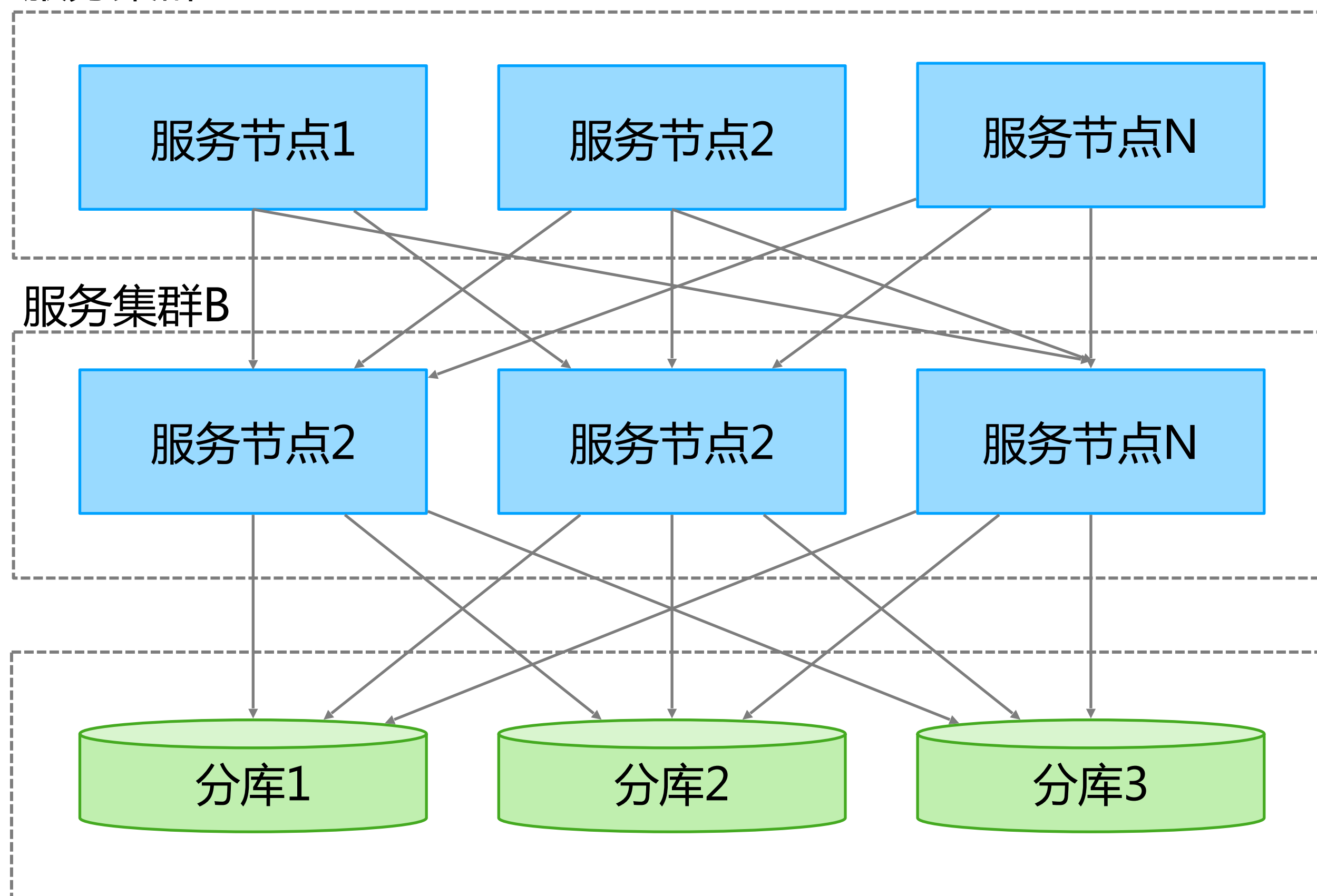
你的服务是否真的是水平扩展的？

你的服务是否真的做到了物理隔离？

每加一台服务器，都会新建N * 连接池个连接

任意节点有异常都可能影响所有的用户

服务集群A



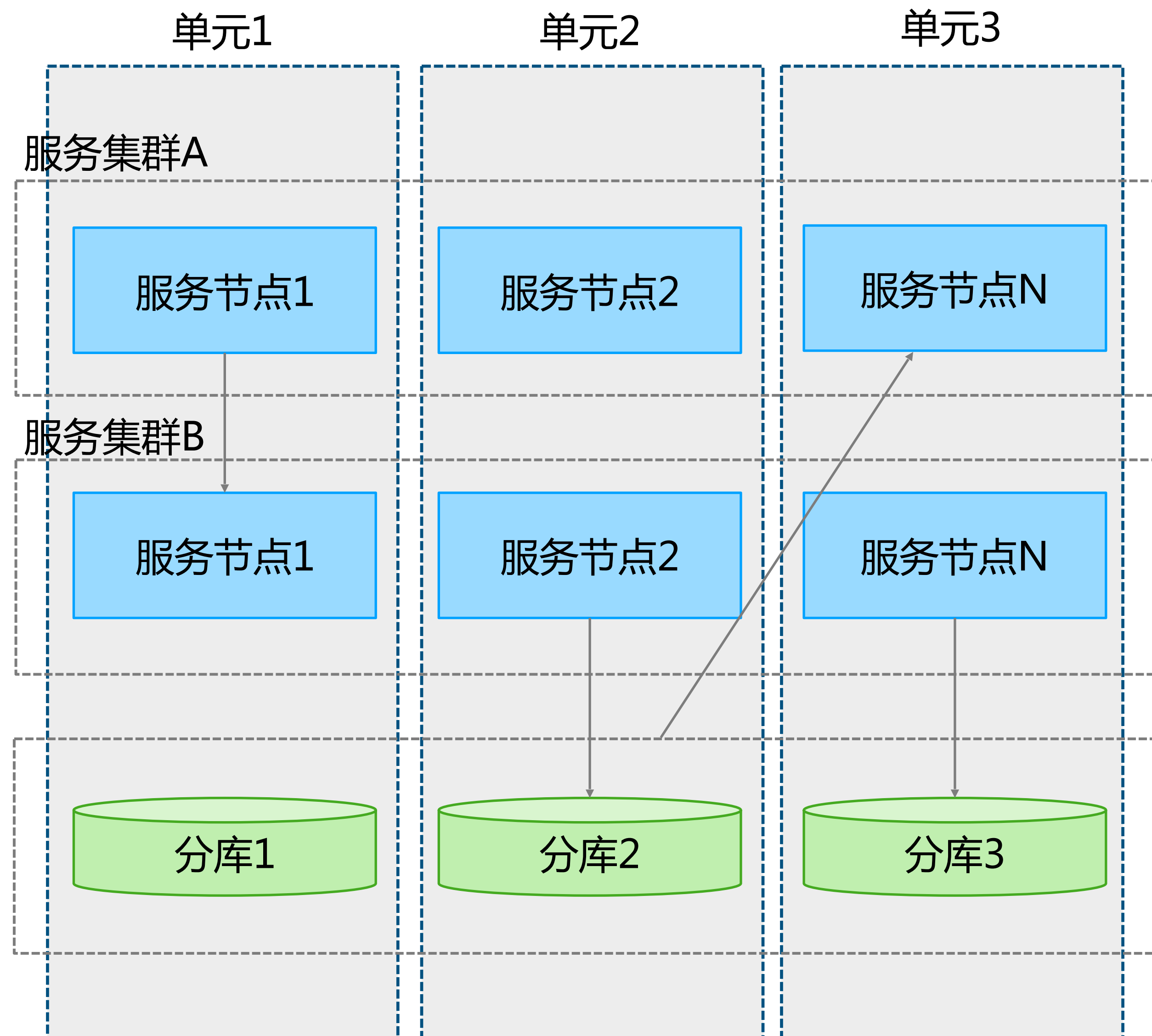
负载均衡

单元化

连接降为原来的 $1/N$

用户流量在某个单元内闭环

搭建新单元实现整个业务逻辑线的扩容
异地多活，单元间流量调整切换



负载均衡

单元化

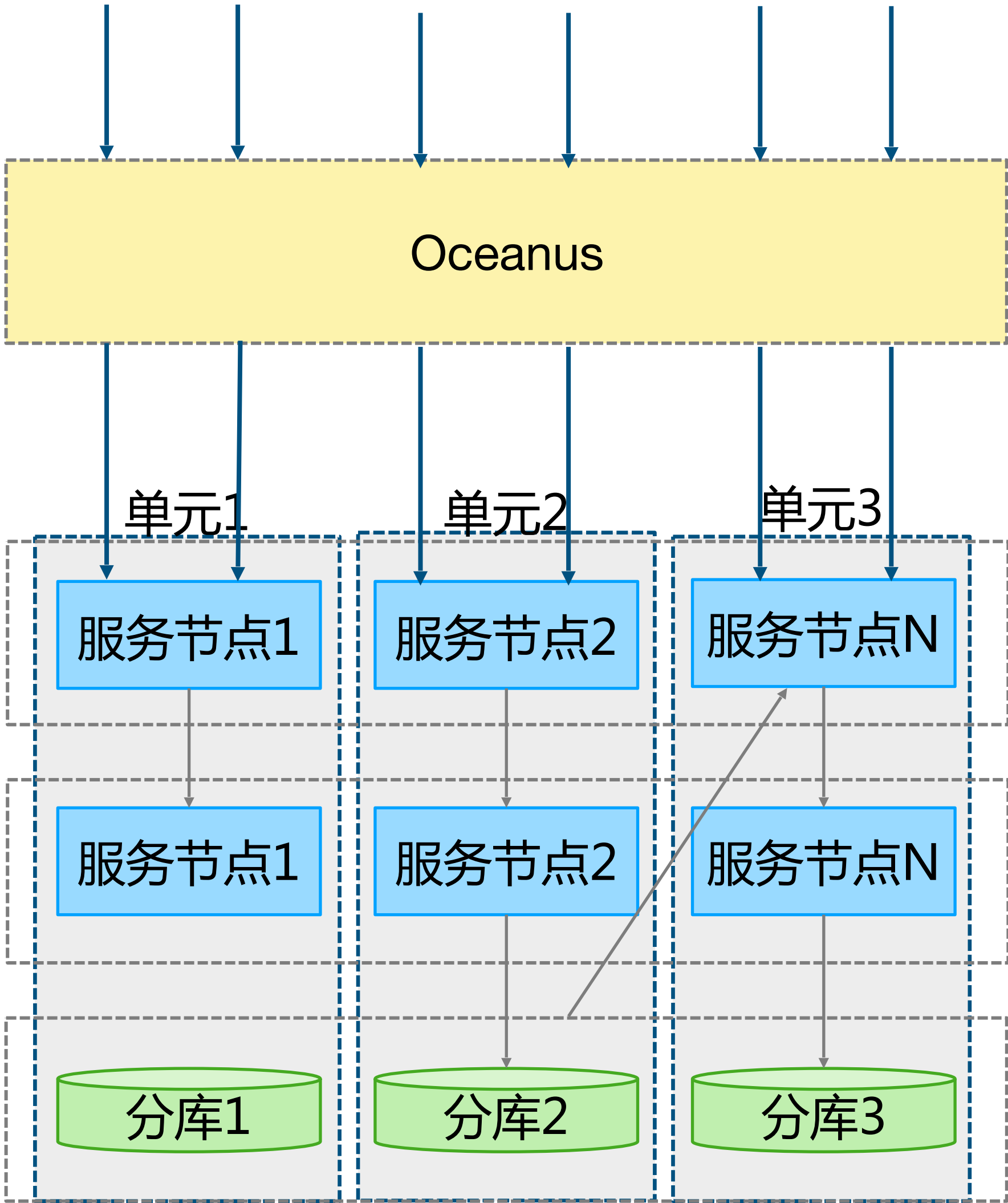
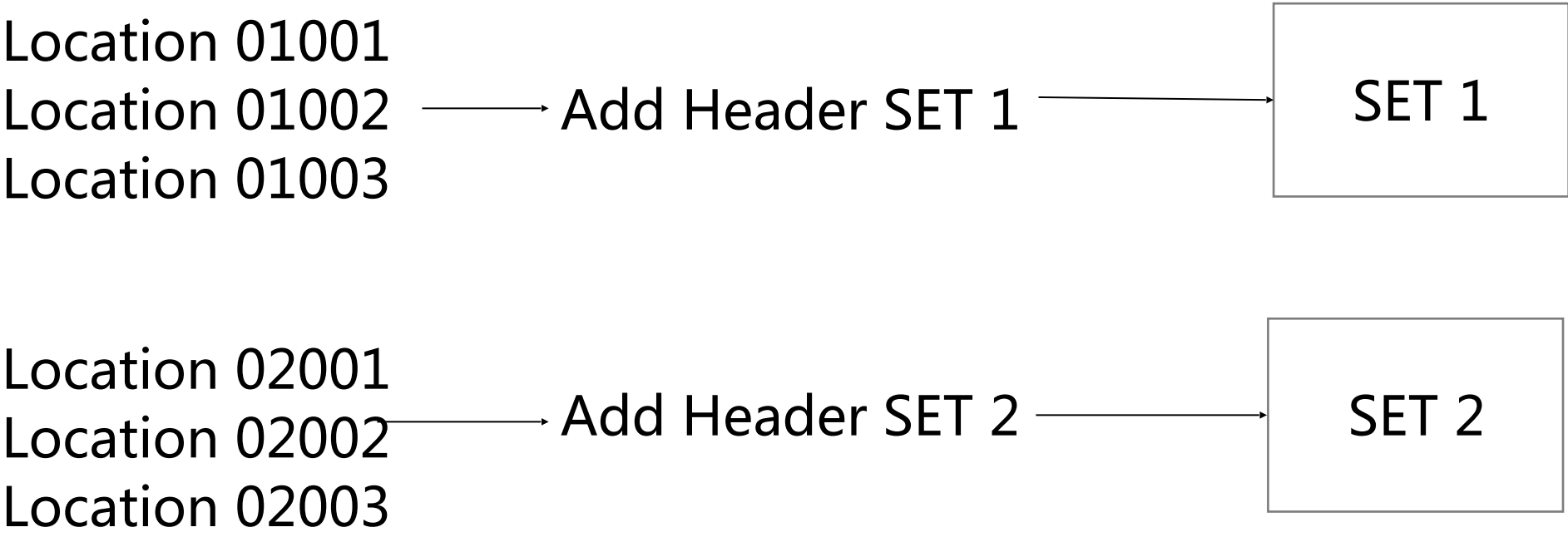
报文转换

支持根据某个Header或者Get参数来修改、删除、新加Header或者

Get参数

流量转发

将含有特定SET标记的流量转发的指定SET的集群单元



未来规划

配置动态化

插件管理：热部署与热升级

自动化运维：快速搭建集群、迁移站点

Q&A