



OpenResty在又拍云容器平台中的应用



叶靖 @ 又拍云

2019.3.23 OpenResty OpenTalk Beijing

行业服务



移动端服务

小程序开发解决方案 提供支持苹果审核的 IPv6 服务
移动专网分发



视频服务

直播解决方案 点播解决方案 短视频解决方案

平台服务



云处理

图片处理 窄带高清 视频处理 音频处理
自适应 H.265 实时截图 自适应 WebP



人工智能

影像识别 人脸识别
文字识别 流媒体识别



云监控



CDP (智能调度)



云通信

短信服务
流量营销服务

基础服务



CDN&存储

对象存储 CDN
图片管家 PCDN
融合云存储 SCDN



弹性计算

云服务器 私有云
IDC 托管 边缘Rewrite
GPU 云处理 容器云



技术输出

CDN 系统 桌面云
Openstack Ceph
基于容器化的云处理集群框架



网络

专线 私有网络
SDWAN 网络优化
智能 DNS 支持 IPV6



安全

DDoS 防护 HTTPDNS
CA 证书 HTTPS/SSL证书
WAF 应用防火墙 流量清洗

<https://github.com/upyun/upyun-resty>

又拍云容器云平台



私有云

办公测试

输入服务名搜索



告警

API 文档

使用文档

平台组

总览

告警

分析

统计

基础设施

主机池

存储池

仓库管理

服务集群

选择服务集群

所有服务集群

服务部署

无状态服务

有状态服务

计划任务

服务治理

负载网关

输入服务名称搜索

创建

更新

伸缩

...

服务名	状态	Pod 个数	CPU 使用	内存使用	容器信息	总运行时间
<input type="checkbox"/> aaa	运行中	1	0.00%	4.41%	nginx:latest CPU 1核 内存 64MB	69 天 22 小时
<input type="checkbox"/> avalon 10.0.5.160:5100 /2.191:8500	更新中	1	0.00%	0.01%	avalon:0.1.20 CPU 1核 内存 1GB	110 天 22 小时
<input type="checkbox"/> brook 实时音频处理	运行中	1	0.02%	0.54%	brookgo:v0.1.0 CPU 1核 内存 1GB	111 天 5 小时
<input type="checkbox"/> default-http-backend	运行中	1	0.01%	22.6%	google_containers/defaultbackend:1.4 CPU 0.0核	17 天 4 小时
<input type="checkbox"/> dnspref dns压测	运行中	1	0.00%	0.26%	rmkn/queryperf:latest CPU 1核 内存 1GB	69 天 6 小时
<input type="checkbox"/> echo-hello-world	运行中	3	0.00%	2.27%	echo-hello-world:v0.0.3 CPU 1核 内存 1GB	17 天 1 小时
<input type="checkbox"/> fiona	运行中	1	0.00%	4.69%	nginx:latest CPU 1核 内存 64MB	19 天 2 小时
<input type="checkbox"/> heheda1	运行中	5	0.00%	4.39%	nginx:1.11-alpine CPU 1核 内存 64MB	94 天 1 小时
<input type="checkbox"/> imgprocess 实时图片处理	运行中	3	13.9%	0.17%	nami:0.6.13 CPU 1核 内存 4GB	26 天 3 小时

业务特点

1. 域名多：上千个域名，不同 SSL 证书
2. 服务多：上千种不同的服务
3. 调用关系复杂：各种服务间互相调用
4. 流量大：上传、图片处理、视频处理
5. 高可用：不能存在单点

解决办法

1. 域名多： API 网关
2. 服务多： 容器化
3. 调用关系复杂： 不同节点的容器间网络互通
4. 流量大： 高性能的负载网关、避免产生额外流量
5. 高可用： VIP、内部域名解析

解决办法



kubernetes

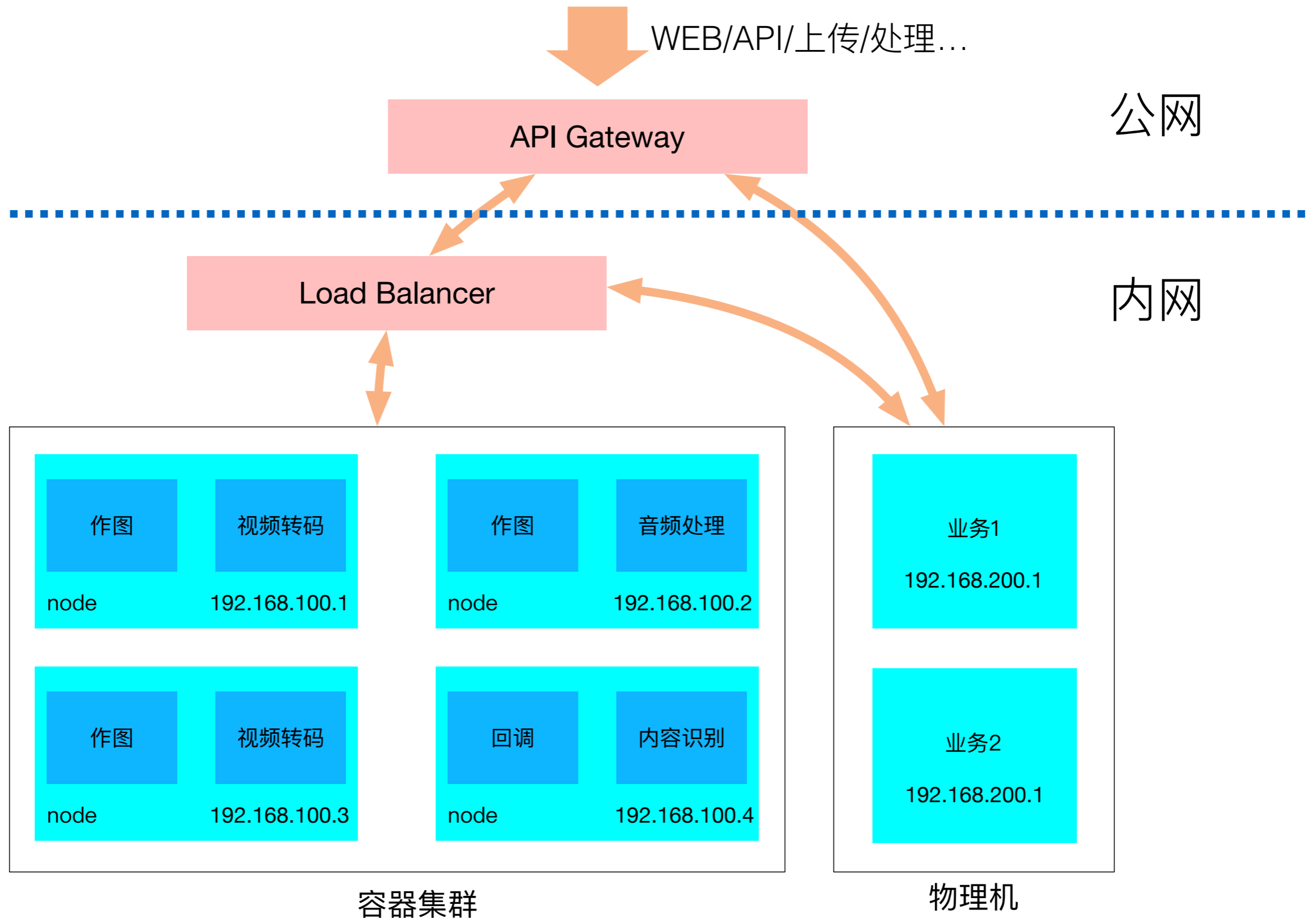
- 服务多
- 调用关系复杂
- 高可用

+



- 域名多
- 流量大

业务拓扑



第一个 OpenResty 应用



对外的 API 网关



Kong作为容器网关

- 域名管理
- 证书调度
- 访问控制
- 权限认证
- 速率控制
- 流量整形
- API 管理

PLUGINS

response-transformer oauth2 acl correlation-id pre-function jwt cors ip-restriction basic-auth key-auth rate-limiting
request-transformer http-log file-log hmac-auth ldap-auth datadog tcp-log zipkin post-function request-size-limiting
bot-detection syslog loggly azure-functions udp-log response-ratelimiting aws-lambda statsd prometheus
request-termination

第二个 OpenResty 应用



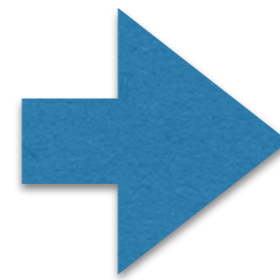
Nginx Ingress Controller

容器云平台内部负载均衡

Nginx Ingress Controller



- 多 SSL 证书调度
- 动态 upstream 管理
- 灰度更新
- ~~TCP 负载均衡~~
- ~~WAF、链路追踪.....~~



动态更新配置

Nginx Ingress Controller 原理



更新监听域名、路径等配置

nginx.conf

Nginx

Lua

Nginx Ingress
Controller

Controller
(Golang)

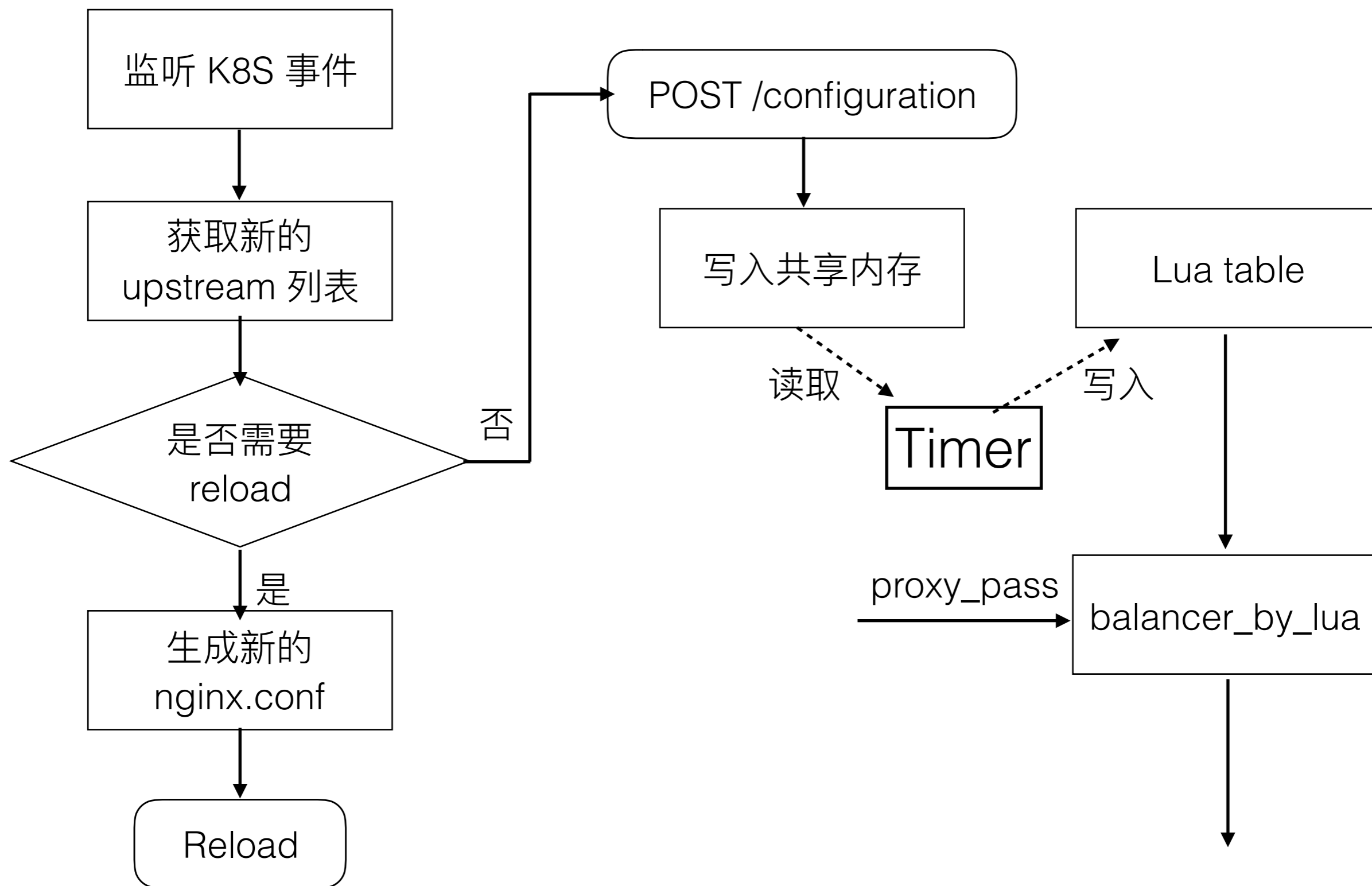
启动

动态 upstream、灰度、证书调度

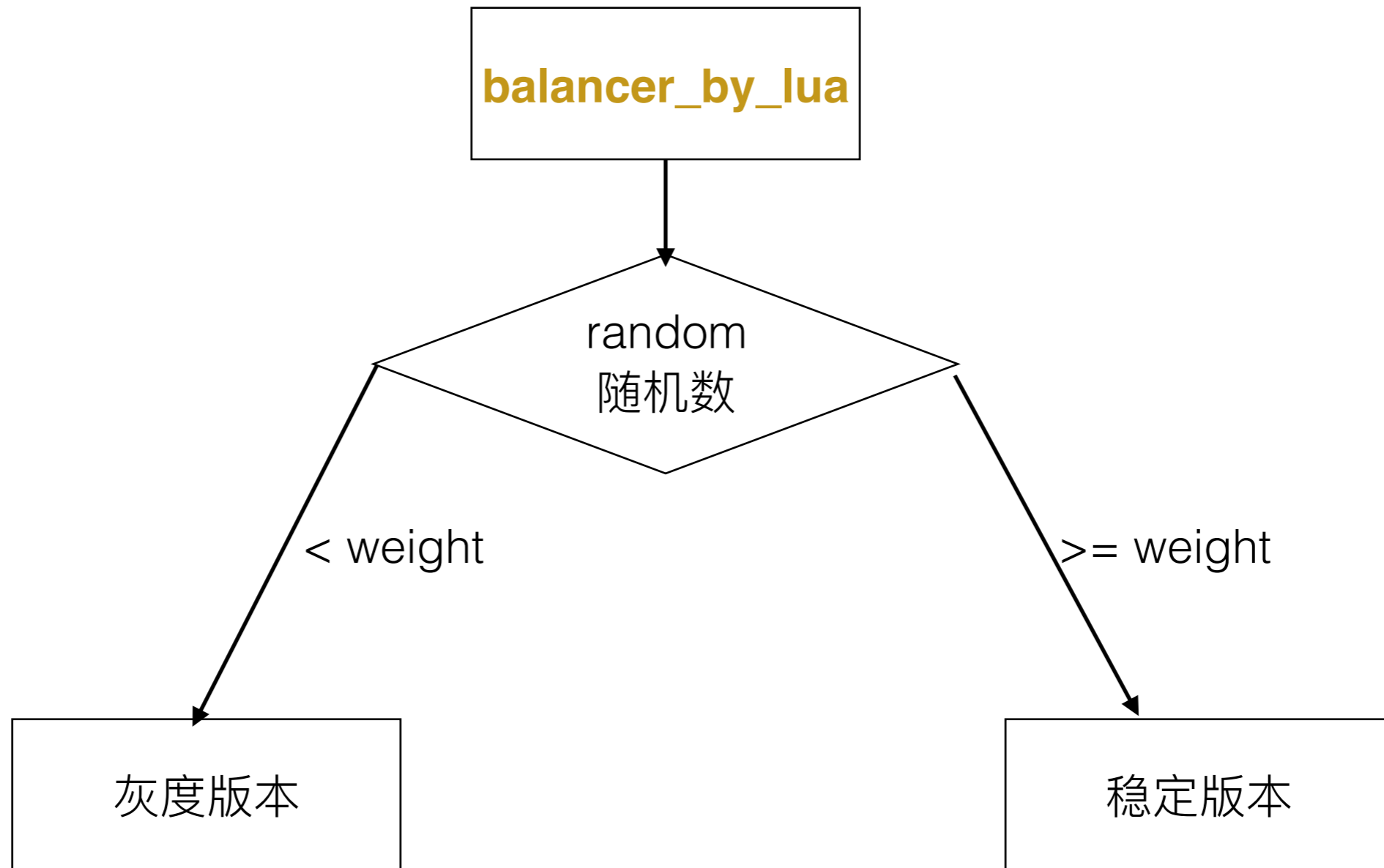
监听K8S事件，获取配置和upstream更新信息



动态更新 upstream 原理



灰度更新原理



SSL 证书调度原理

```
ssl_certificate_by_lua_block {  
    certificate.call()  
}
```

```
local hostname = ssl.get_servername()  
local pem_cert_key = get_pem_cert_key(hostname)  
ssl.clear_certs()  
set_pem_cert_key(pem_cert_key)
```

部署与更新



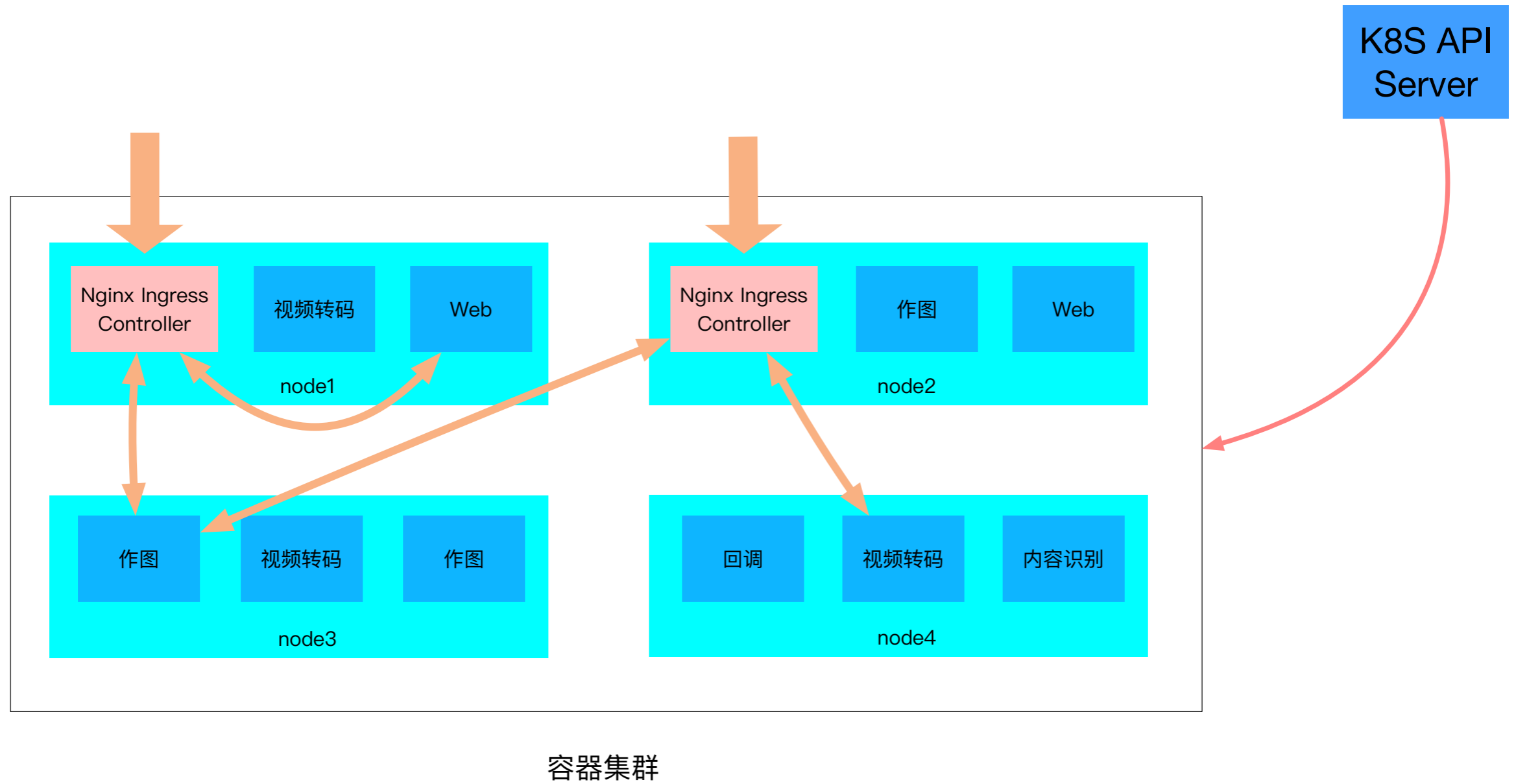
```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/master/deploy/mandatory.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/master/deploy/provider/baremetal/service-nodeport.yaml
```

```
spec:
  serviceAccountName: nginx-ingress-serviceaccount
  containers:
  - name: nginx-ingress-controller
    image: quay.io/kubernetes-ingress-controller/nginx-ingress-controller:0.23.0
    args:
      - /nginx-ingress-controller
      - --configmap=$(POD_NAMESPACE)/nginx-configuration
      - --tcp-services-configmap=$(POD_NAMESPACE)/tcp-services
      - --udp-services-configmap=$(POD_NAMESPACE)/udp-services
      - --publish-service=$(POD_NAMESPACE)/ingress-nginx
      - --annotations-prefix=nginx.ingress.kubernetes.io
    securityContext:
      allowPrivilegeEscalation: true
```

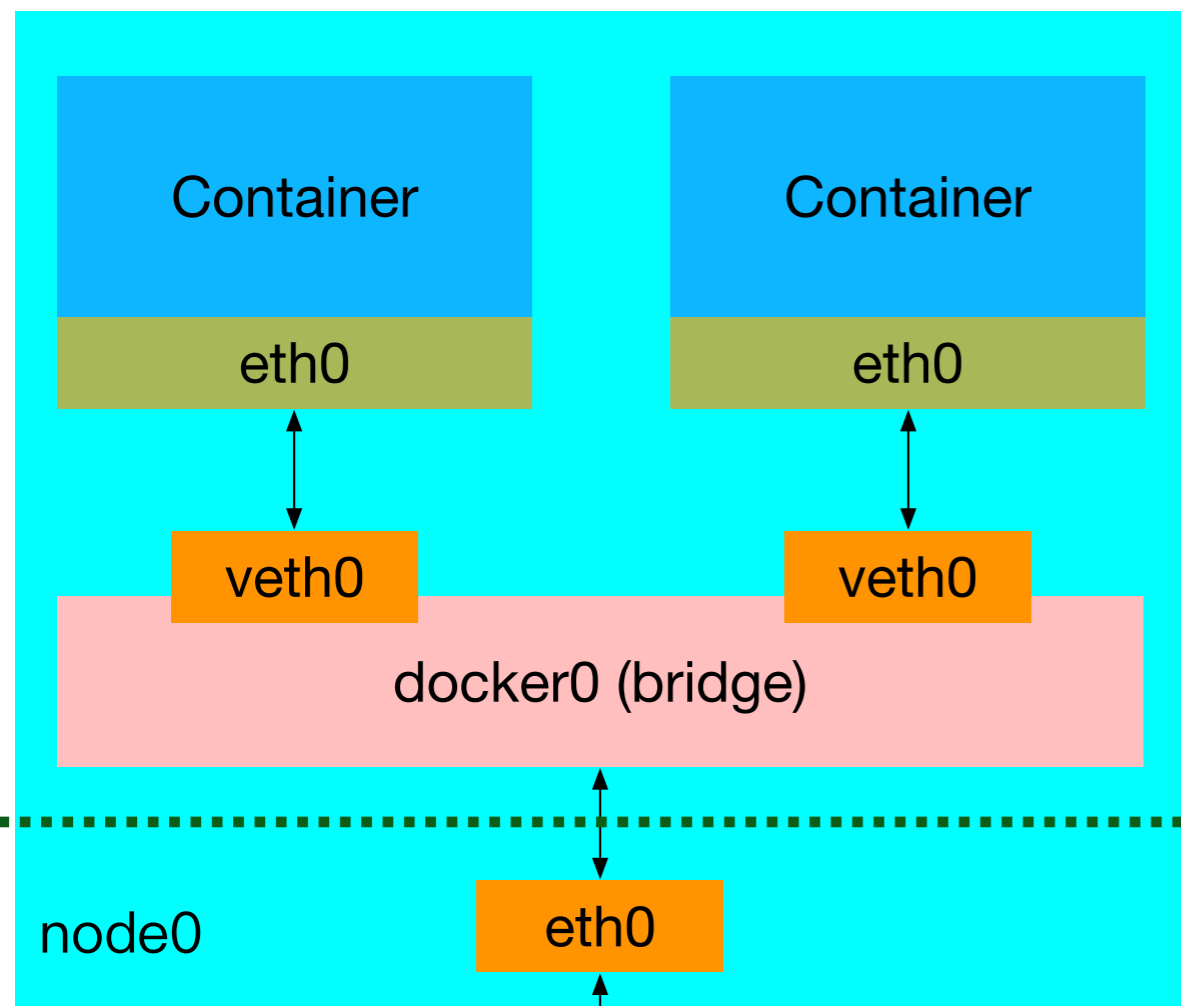
修改版本号即可

部署拓扑

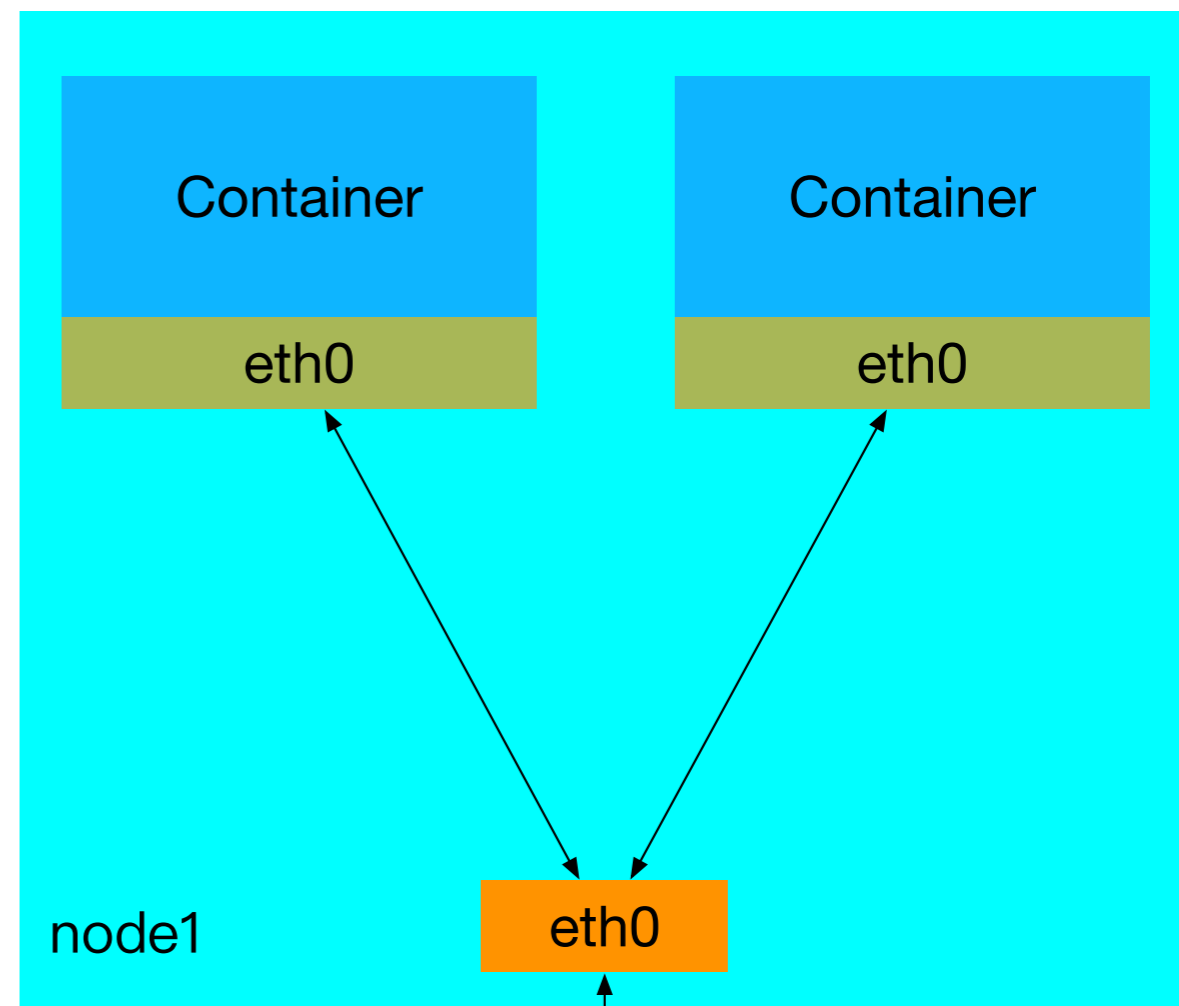


Host or Bridge

内部网络 172.17.0.0/24



Bridge 模式



Host 模式

问题

1. 如何热更新负载均衡网关？

- 直接更新 Nginx Ingress Controller 会产生 5xx
- 业务分散在多个部门，切流量更新非常痛苦
- 灰度更新网关本身不方便

2. 负载均衡网关能否不加入 K8S 集群？

- 负载均衡网关机器专机专用，与容器机器不同配置，且通常不会变动
- Nginx Lua 项目如何直接访问到容器

我们的方案

需要一个能独立部署在物理机上的 Nginx，并且能直接访问到 K8S 集群中的容器



1. 把 Ingress Nginx 移动容器外
2. 打通 Nginx 所在机器与容器集群的网络

第三个 OpenResty 应用



Ingress OpenResty

独立部署的内部负载均衡

Ingress OpenResty

1. 分离 Controller 与 Nginx

- Controller 与 Nginx 相互独立
- Nginx 可独立 reload, hot update
- Controller 可以直接重启

2. 支持 K8S 集群外部署

- CentOS7 物理机部署
- 负载均衡机器不需要加入到 K8S 集群（需要负载均衡与容器在同一个二层网络下）

分离 Controller 与 Nginx

- Nginx 配置模板
 - 从 ingress-nginx 拷贝 (rootfs/etc/nginx/template/nginx.tpl)
 - 去掉不需要的配置项
- Lua 代码
 - 从 ingress-nginx 拷贝 (rootfs/etc/nginx/lua)
- Controller 去除启动与停止 Nginx 的代码
 - 修改 nginx.go 中相关代码
- 编译一个新的 Nginx
 - 从 OpenResty 编译

Nginx 重启后，upstream 丢失！



upstream 信息持久化保存到磁盘，在
init_by_lua 阶段从磁盘加载回共享内存

Dockerfile

- 基于 CentOS 7.3
- 静态安装 pcre、OpenSSL
- 安装 OpenResty
- 安装 Lua Resty 库

<https://gist.github.com/yejingx/bb36cf78a149635ccd0581b311bcc403>

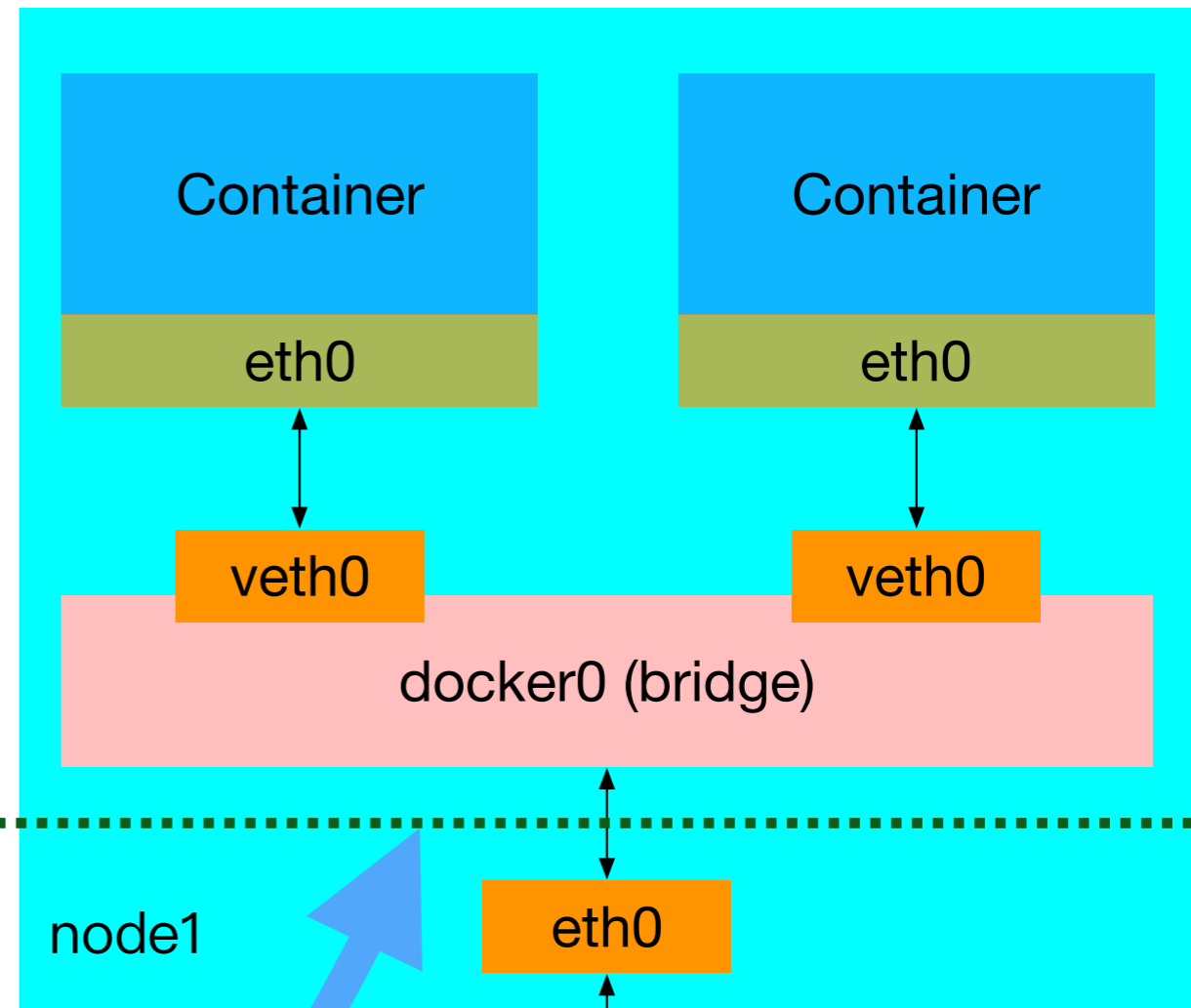
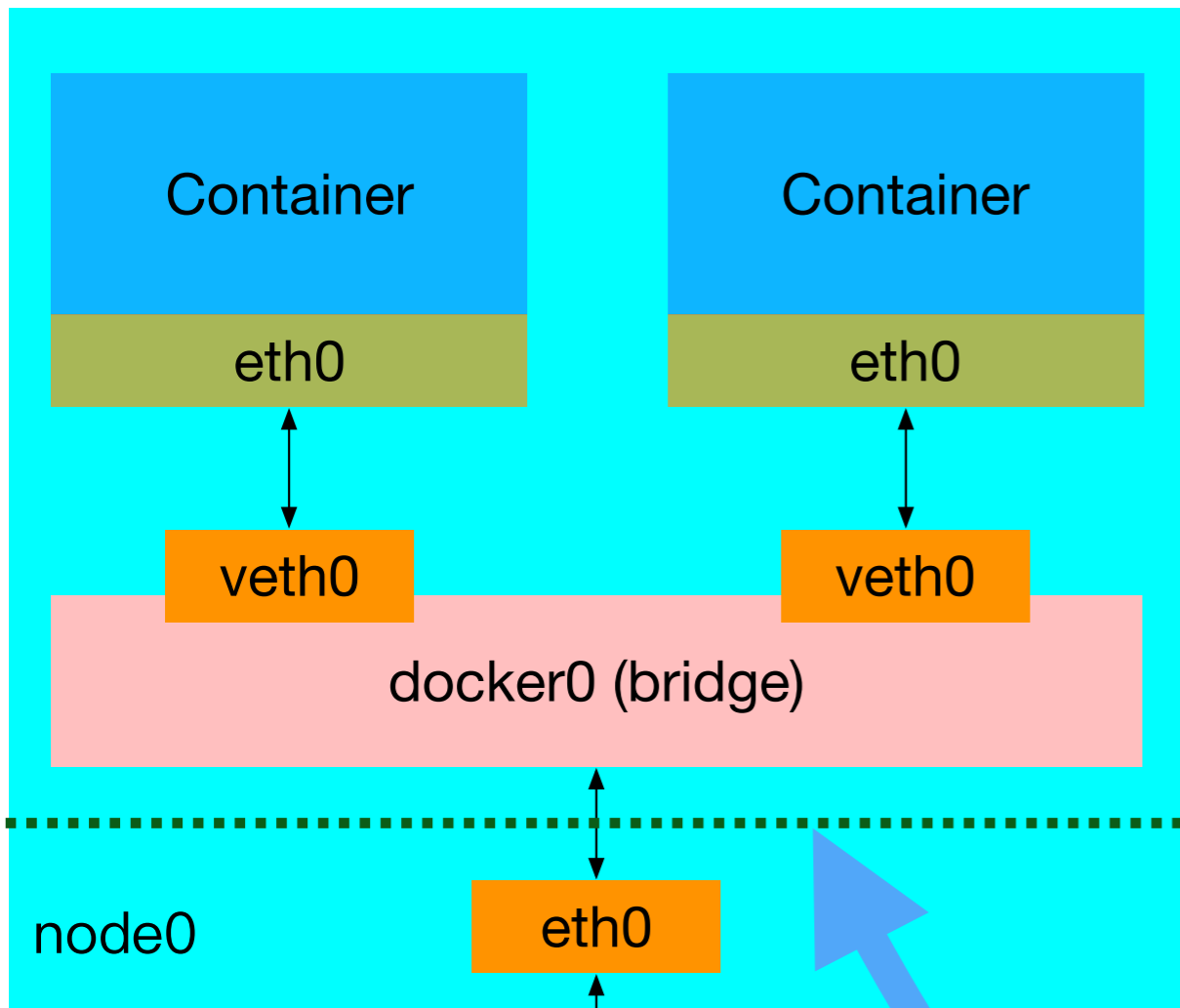
支持 K8S 集群外部署

- 打通物理机与 K8S 集群的网络
 - K8S 的网络互通原理
- 流量大
 - 尽量避免代理产生新的流量

K8S网络模式

内部网络 172.17.0.0/24

内部网络 172.17.1.0/24



192.168.100.1

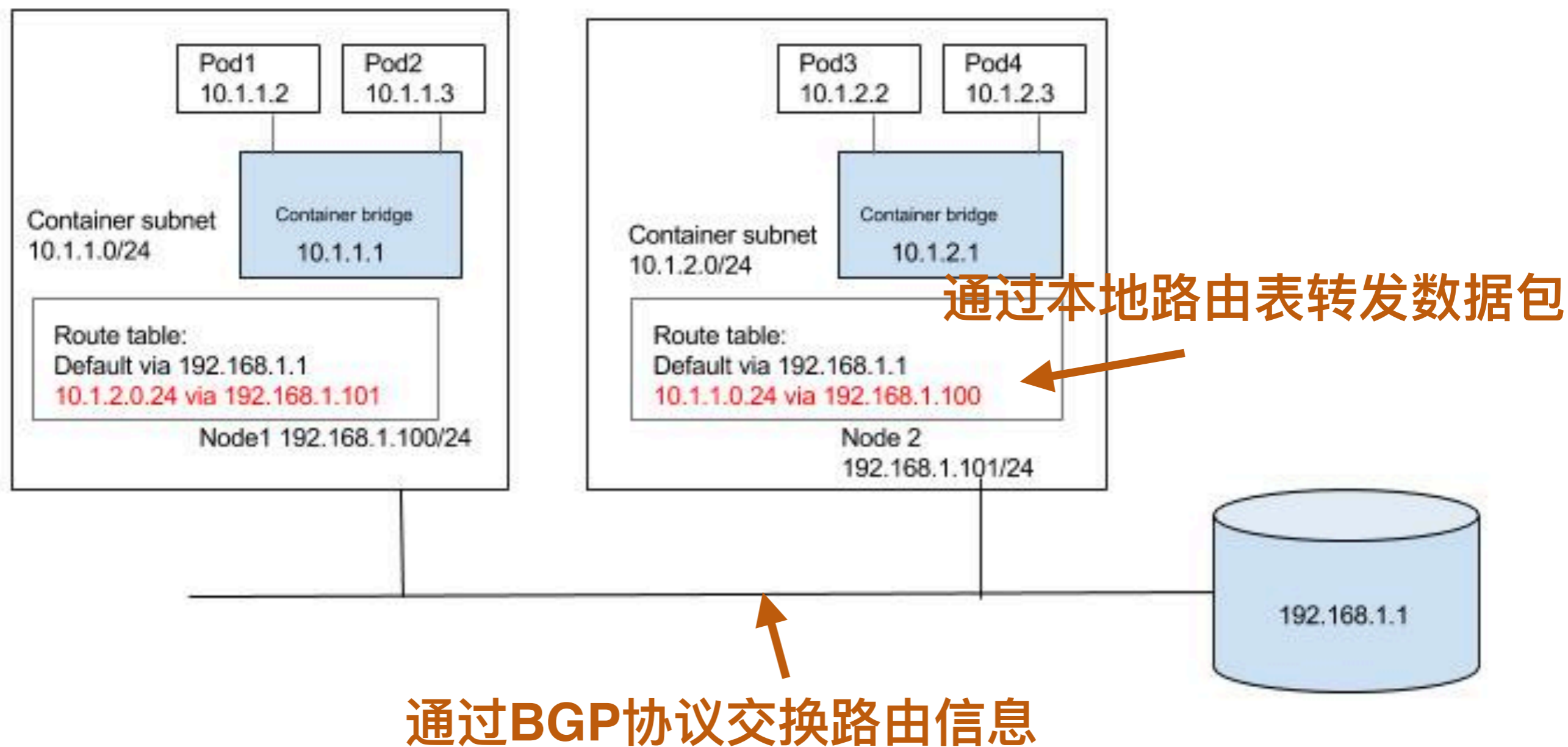
192.168.100.2

节点间网络互通

K8S网络组件

- Flannel (Overlay, route)
- Calico (route)
- **Kube-Router (route)**

Kube-Router 原理



图片来源: <https://cloudnativelabs.github.io/post/2017-04-18-kubernetes-networking/>

我们的方案

Kube-Router 不支持

1. 在负载网关机器上部署 Kube-Router



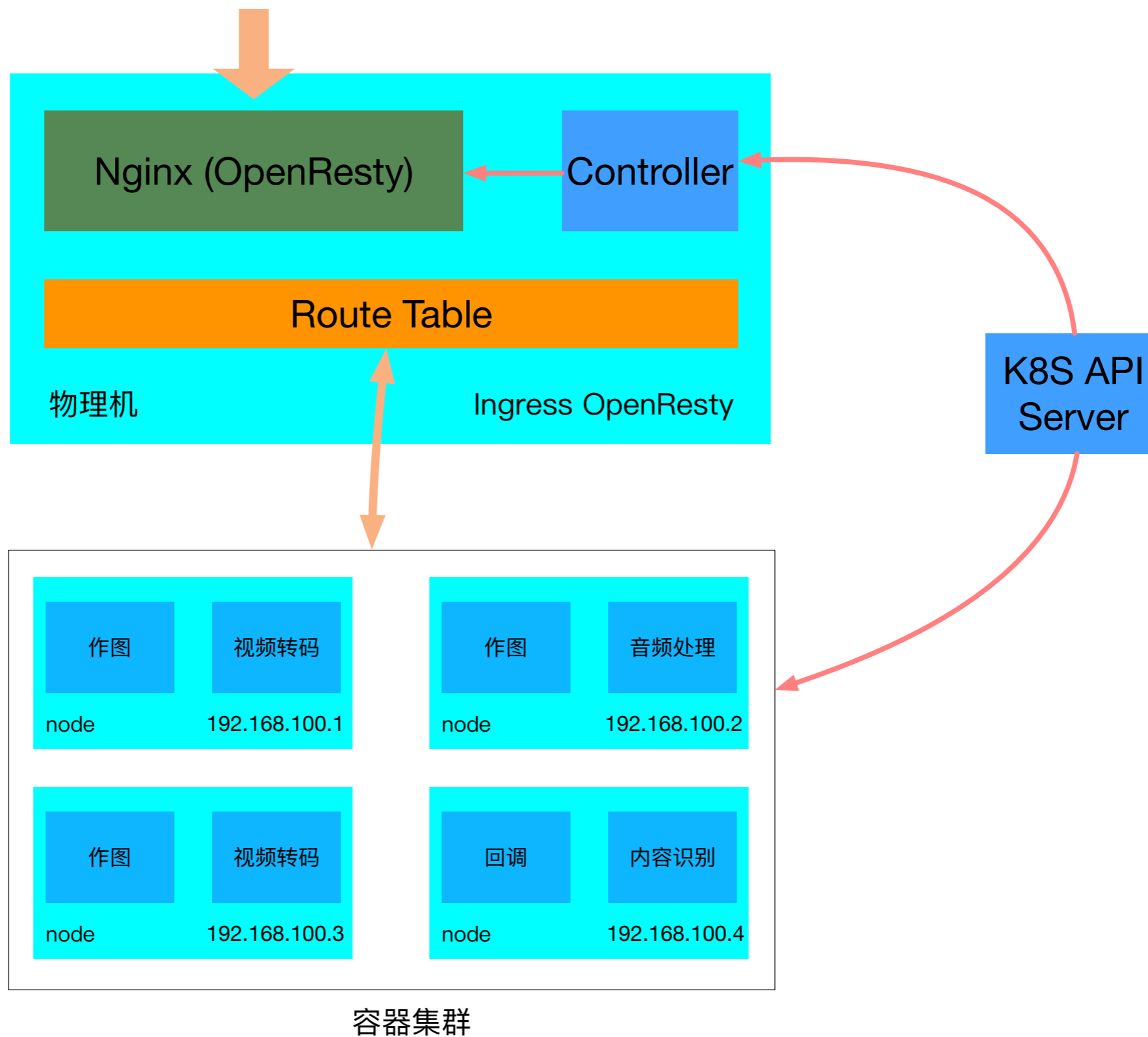
2. 在负载网关机器上部署 Quagga 同步路由规则



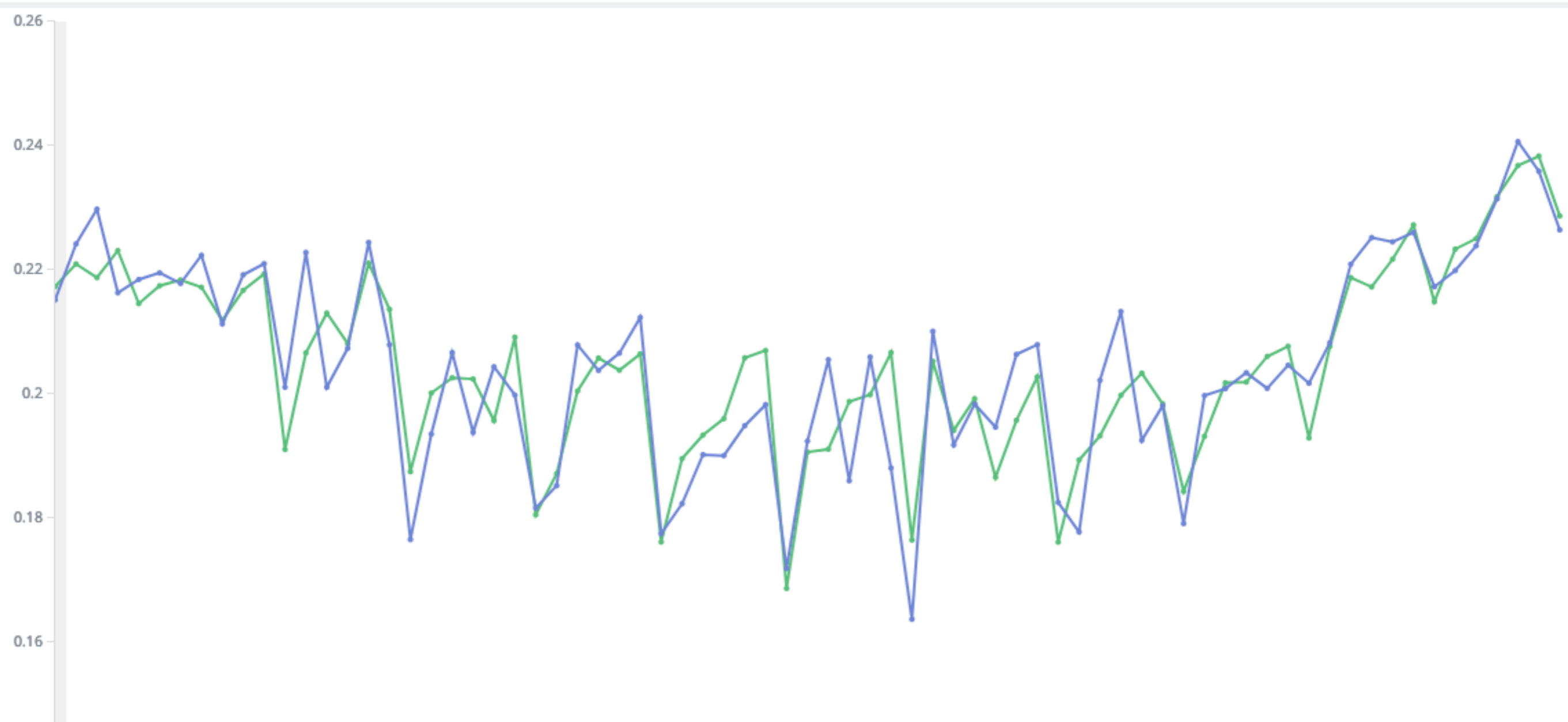
3. 在K8S节点中部署负载网关，并且不在节点上部署其它服务




部署拓扑



Request Time V.S.



 Ingress OpenResty

 Host 模式部署的 Nginx Ingress Controller

优缺点

- 优点
 - 平滑升级
 - 性能损耗小
 - 支持集群外部署
- 缺点
 - 需要自己维护项目
 - 需要与容器在同一个二层网络或者打通 IPIP 隧道

Thanks

Q & A

