

CVTE容器云实践之路

许坤丰

CVTE 运维部技术经理

— 部件业务 —

36%

液晶显示主控板卡
全球出货量占比

— 未来教育 —

seevo

35.5%

交互智能平板市场
连续六年排名第一

— 企业服务 —

MAXHUB

24%

会议平板市场
销量销售额均排名第一

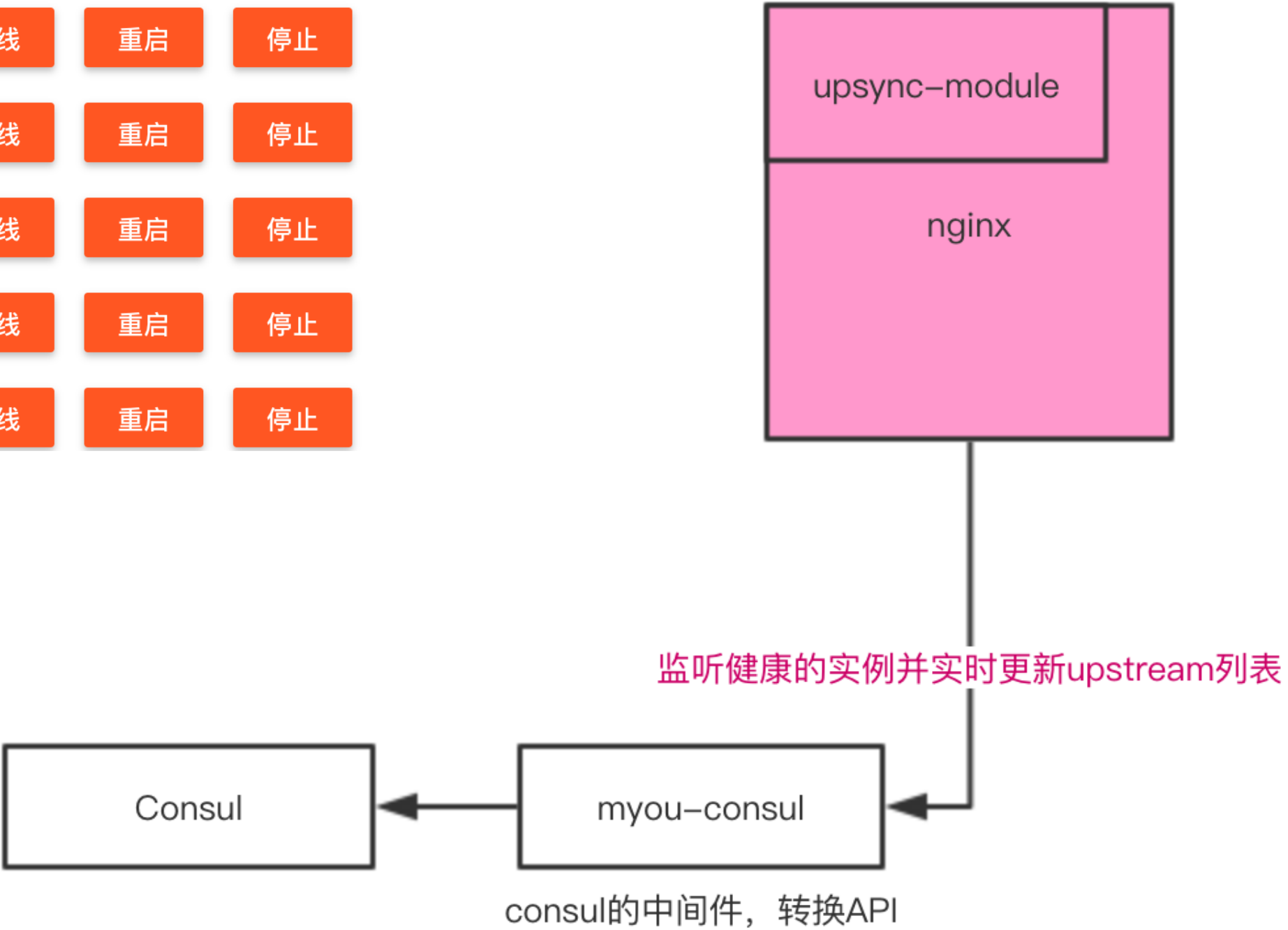
数据来源：视源股份2018年半年报摘要
奥维云网《《2018年Q2中国B2B IWB市场研究报告》
奥维云网《2017年中国会议平板市场研究报告》

基于salt和consul的发布系统

批量发布

最近部署信息

主机信息	上个/当前版本	部署路径	发布人员	测试人员	部署状态	健康状态	机器状态	预发布			
10.10.10.10	0 / 47	/usr/local/tomcat/tomcat3/webapps	张三		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.11	0 / 47	/usr/local/tomcat/tomcat5/webapps	李四		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.12	0 / 47	/usr/local/tomcat/tomcat4/webapps	王五		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.13	0 / 47	/usr/local/tomcat/tomcat5/webapps	赵六		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.14	0 / 47	/usr/local/tomcat/tomcat3/webapps	钱七		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.15	0 / 47	/usr/local/tomcat/tomcat5/webapps	孙八		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.16	0 / 47	/usr/local/tomcat/tomcat2/webapps	周九		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.17	0 / 47	/usr/local/tomcat/tomcat3/webapps	吴十		✓成功	😊正常	▶运行	▶	下线	重启	停止
10.10.10.18	0 / 47	/usr/local/tomcat/tomcat2/webapps	郑十一		✓成功	😊正常	▶运行	▶	下线	重启	停止



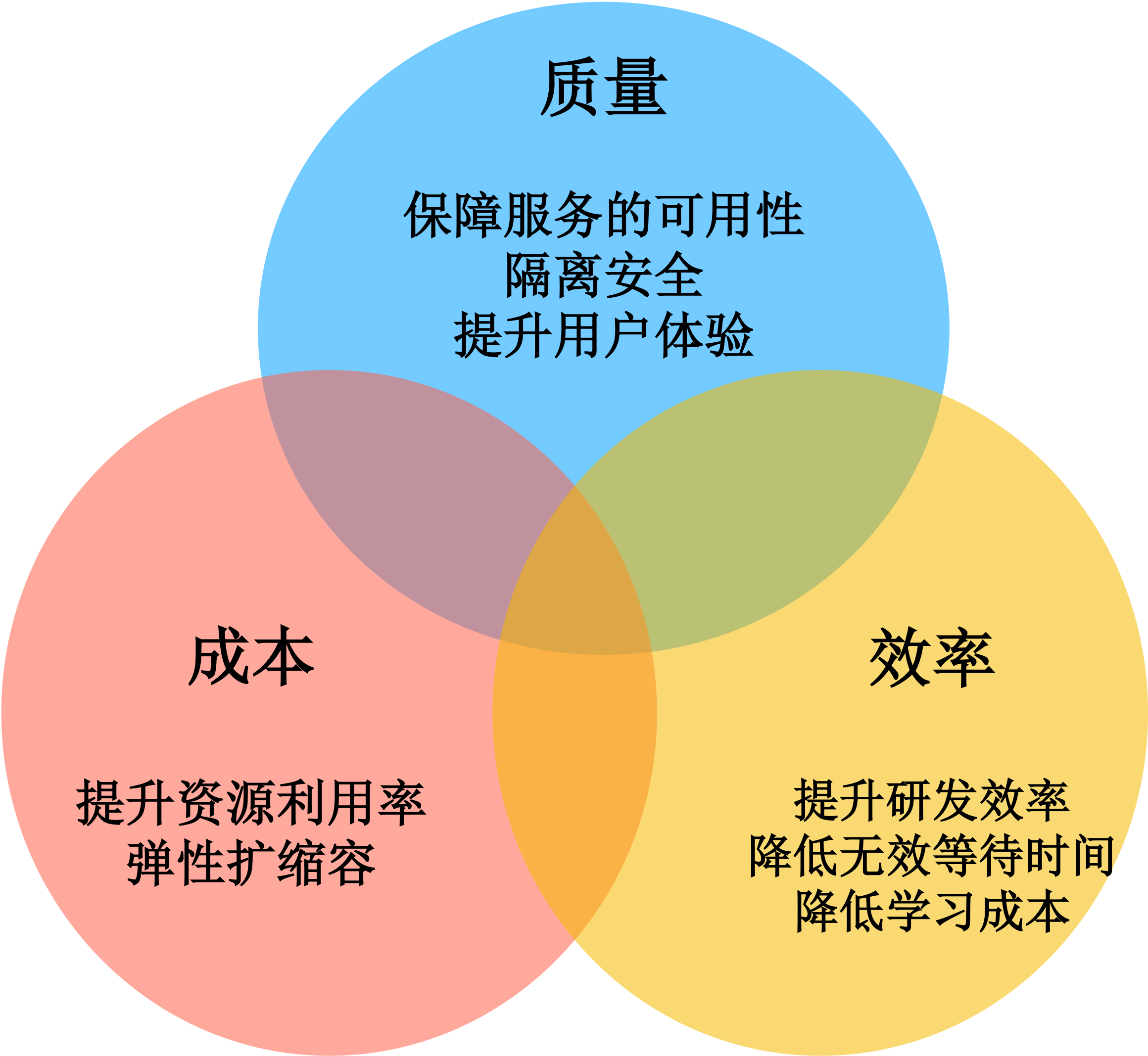
发布系统带来的改变

- 自助发版本成为可能
- 版本管理
- 故障屏蔽功能
- 跨区部署

新的问题

- 隔离性差
- 没有调度系统，资源冷热不均
- 支持的程序种类有限
- 难以支撑复杂的devops场景


容器平台目标



平台概览



界面

 鲸云

应用管理

调用关系

负载均衡

镜像构建

模版管理

nfs存储卷

项目配置

日志搜索

操作记录

🏠

👥 平台服务部

📄 sql审核

🔍 快速搜索

快速接入

🔗 帮助








👤 许坤丰






🔍 查找应用







批量更新

+ 新建应用

myou-sql-inspect ?

请配置日志采集路径，否则会丢失容器日志 ->   配置中心  预发版本    

v1 ● 上线     

容器ip	主机ip	镜像	健康	容器状态	重启次数	操作
172.30.54.43	172.17.84.144	myou-sql-inspect:master-e576225 	 正常 ?	运行 ?	0	登录终端 容器日志 
172.30.130.69	172.17.84.170	myou-sql-inspect:master-e576225 	 正常 ?	运行 ?	0	登录终端 容器日志 

快速接入

Jenkinsfile 帮助应用快速生成jenkins 项目

应用类型

从已有Jenkins项目导入信息 ☒

方舟/萌友项目的Jenkins地址

镜像发布类型

生成Jenkins Pipeline

Jenkins Pipeline生成

Dockerfile生成

应用类型

从已有Jenkins项目导入信息 ☐

git地址

git分支

docker仓库项目名

镜像名

构建命令

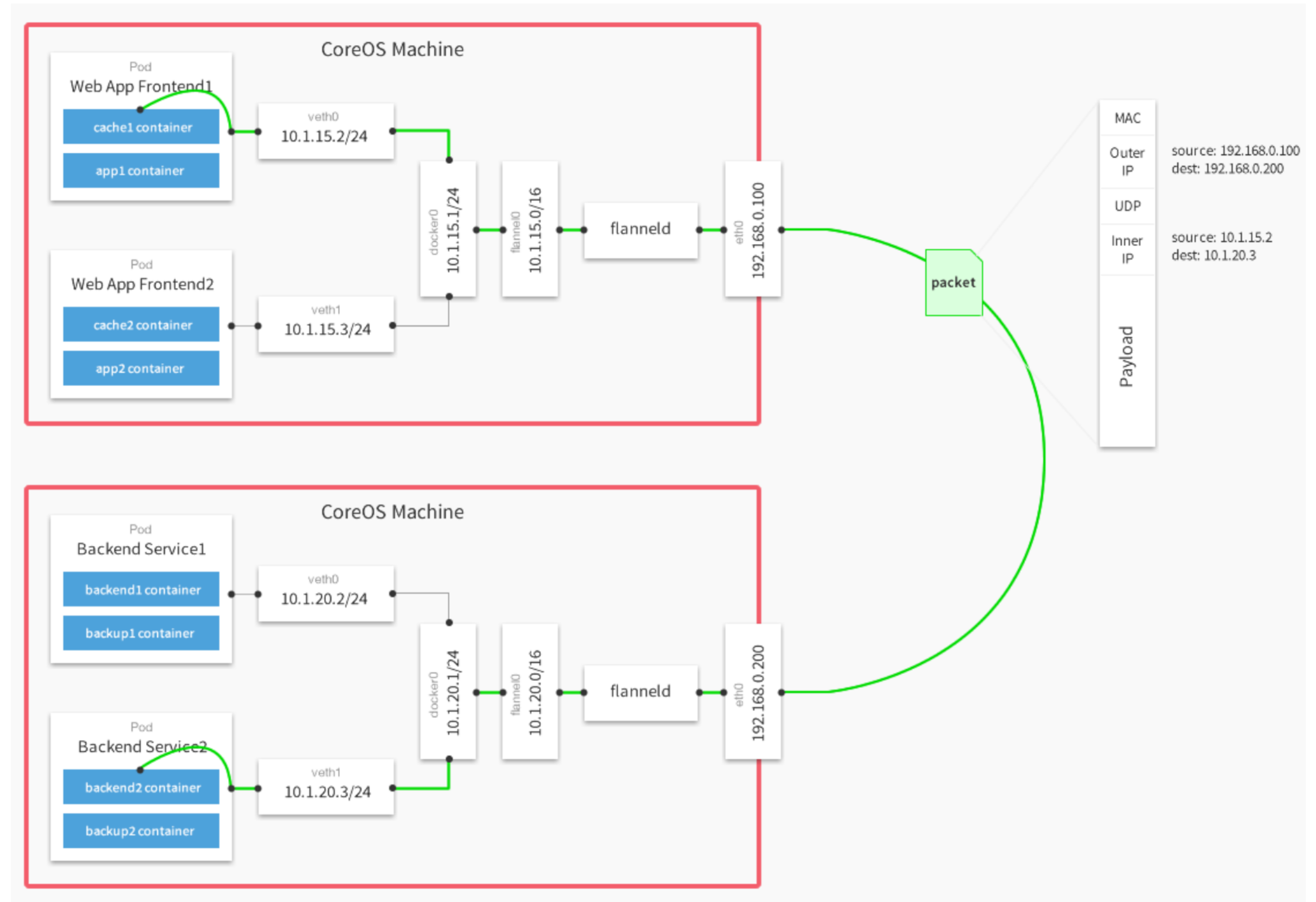
包路径

镜像发布类型

生成Jenkins Pipeline

网络 flannel

- 开发测试环境采用UDP
- 生产环境采用Ali VPC



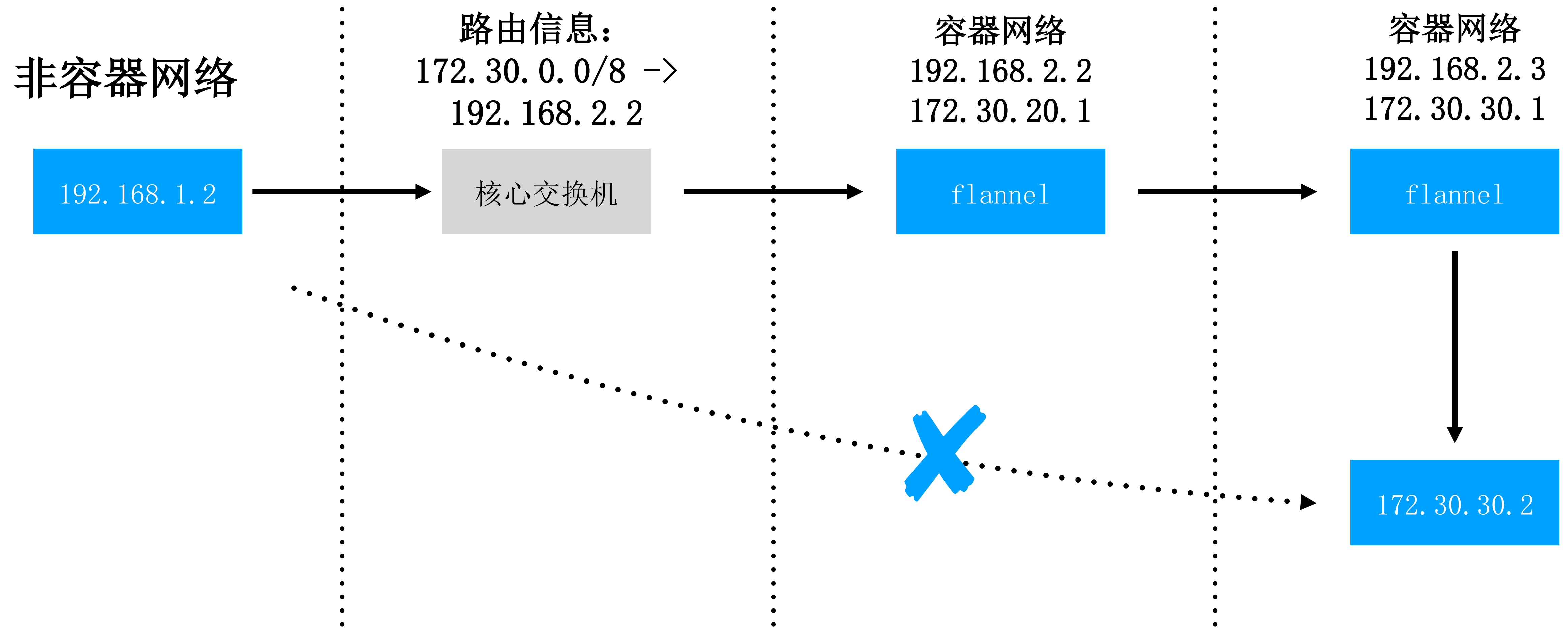
flannel下打通容器和非容器网络

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
0.0.0.0	10.10.14.1	0.0.0.0	UG	100	0	0	eth0
10.10.14.0	0.0.0.0	255.255.254.0	U	100	0	0	eth0
172.20.0.0	0.0.0.0	255.255.0.0	U	0	0	0	br-bf7711cdcdd8
172.30.0.0	0.0.0.0	255.255.0.0	U	0	0	0	flannel.1
172.30.43.0	0.0.0.0	255.255.255.0	U	0	0	0	docker0

通过在发起集群外机器上增加路由信息
或者
在核心交换机上增加路由信息

flannel下打通容器和非容器网络



节点选择

选择相对高规格的机器

- 节点上镜像重复使用率高
- 网络规格可以更高
- 资源调节更灵活

集群管理

集群扩容的烦恼

- 安装环境不标准，复杂
- 各种实现监控和自动化的agent下发
- 节点配置的变更：主机名，内核参数等

集群管理

节点资源全自动交付

- 使用虚拟机镜像实现标准化
- 使用cloud-init 实现个性化

```
cat << EOF > /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
[Service]
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf"
Environment="KUBELET_SYSTEM_PODS_ARGS=--pod-manifest-path=/etc/kubernetes/manifests --allow-privileged=true"
Environment="KUBELET_NETWORK_ARGS=--network-plugin=cni --cni-conf-dir=/etc/cni/net.d --cni-bin-dir=/opt/cni/bin"
Environment="KUBELET_DNS_ARGS=--cluster-dns=172.21.0.10 --pod-infra-container-image=registry.cn-hangzhou.aliyuncs.com/acs/pause-amd64:3.0 --enable-controller-attach"
Environment="KUBELET_AUTHZ_ARGS=--authorization-mode=Webhook --client-ca-file=/etc/kubernetes/pki/ca.crt"
Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=0"
Environment="KUBELET_CGROUP_ARGS=--system-reserved=memory=300Mi --kube-reserved=memory=400Mi --eviction-hard=imagefs.available<15%,memory.available<10%,nodefs.inodesFree<5% --cgroup-driver=systemd"
Environment="KUBELET_CERTIFICATE_ARGS=--rotate-certificates=true --cert-dir=/var/lib/kubelet/pki"
Environment="KUBELET_EXTRA_ARGS=--logtostderr=false --log-dir=/var/log/kubernetes/kubelet --v=5 --max-pods=${max_pods} --feature-gates=CustomPodDNS=true"
ExecStart=
ExecStart=/usr/bin/kubelet \${KUBELET_KUBECONFIG_ARGS} \${KUBELET_EXTRA_ARGS} \${KUBELET_SYSTEM_PODS_ARGS} \${KUBELET_NETWORK_ARGS} \${KUBELET_DNS_ARGS} \${KUBELET_AUTHZ_ARGS} \${KUBELET_CADVISOR_ARGS} \${KUBELET_CGROUP_ARGS} \${KUBELET_CERTIFICATE_ARGS} \${KUBELET_EXTRA_ARGS}
EOF
sed -i "s/i-bp12ylu3ugxtsrcmmyinn/${instance_id}/g" /etc/systemd/system/kubelet.service.d/10-kubeadm.conf

mkdir -p /var/log/kubernetes/kubelet
systemctl daemon-reload

systemctl enable docker
systemctl enable kubelet
/usr/sbin/ip link delete cni0
systemctl start docker
systemctl start kubelet

kubeadm reset
kubeadm join --token "12:6443 --discovery-token-unsafe-skip-ca-verification"
```

<input type="checkbox"/>	模板ID	模板名称	创建时间	默认版本	最新版本	操作
<input type="checkbox"/>	lt-bp18mmizfyna1 	k8s_node	2018年7月28日 09:13:35	4	4	创建实例 新建版本


```
.:53 {
log
errors

```

CoreDNS

```
reload

```

```
health

```

```
trace zipkin http://ccloud/cloud/api/v1/zipkin/api/v1/spans {

```

```
    every 1

```

```
    service k8s-coredns-prod

```

```
}

```

```
kubernetes cluster.local {

```

```
}

```

```
hosts /etc/hostfile {

```

```
    fallthrough

```

```
}

```

```
proxy gz.cvte.cn 10.10.10.10:53 {

```

```
    policy round_robin

```

```
}

```

```
proxy . /etc/resolv.conf {

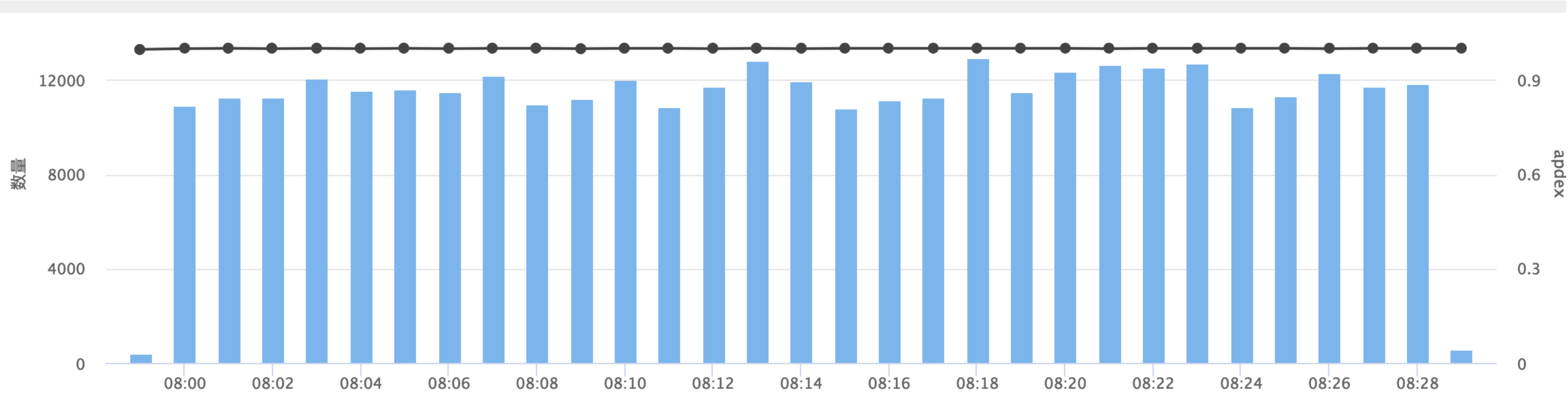
```

```
    policy first

```



CoreDNS



QUERY /logtail.cn-hangzhou.log.aliyuncs.com.gz.cvte.cn/A	HTTP_REQUEST	时延	5	-	HTTP 状态码 : 200	2019-01-20 08:02:23
QUERY /oss1.seewo.com/A	HTTP_REQUEST	时延	11	-	HTTP 状态码 : 200	2019-01-20 08:02:18

耗时(ms)	调用栈
5	100.00% QUERY /logta
5	100.00% errors
5	100.00% log
5	100.00% cache

```
2019-01-20 08:02:23[INFO] (#0) -
clientIp: 172.20.24.9:35176
serverIp: 172.20.23.9
result:
```

CoreDNS

```
logs # cat /etc/resolv.conf
nameserver 172.21.0.10
search psd.svc.cluster.local svc.cluster.local cluster.local gz.cvte.cn
options ndots:5
logs #
```

rds.cvte.com 的查询过程

1. rds.cvte.com.psd.svc.cluster.local
2. rds.cvte.com.svc.cluster.local
3. rds.cvte.com.cluster.local
4. rds.cvte.com.gz.cvte.cn
5. rds.cvte.com

- Deployment的dns配置(spec.template.spec.dnsConfig)

```
dnsConfig:
  options:
  - name: ndots
    value: "2"
```

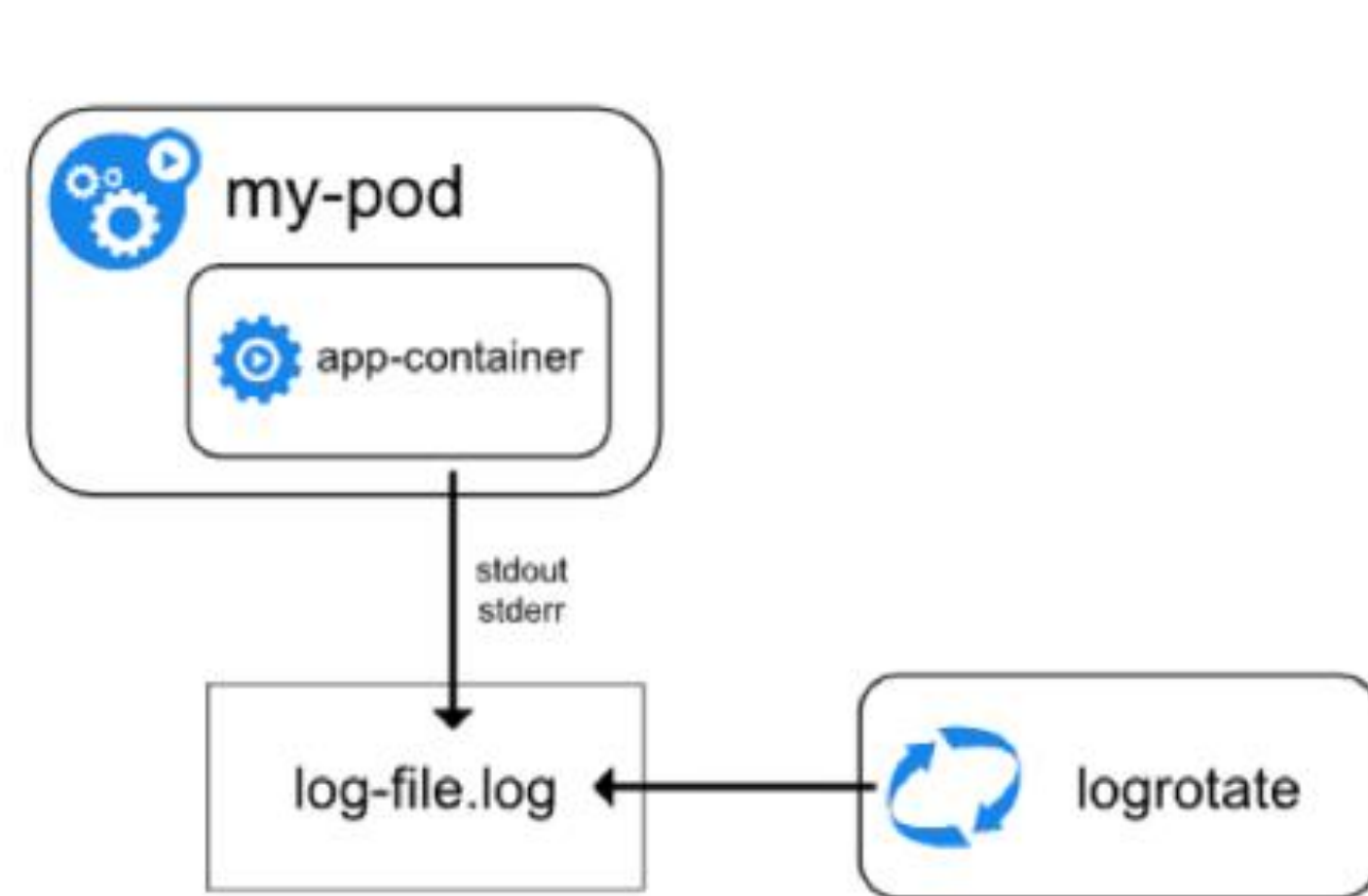
kubelet 参数 : `--feature-gates=CustomPodDNS=true`

日志的难题

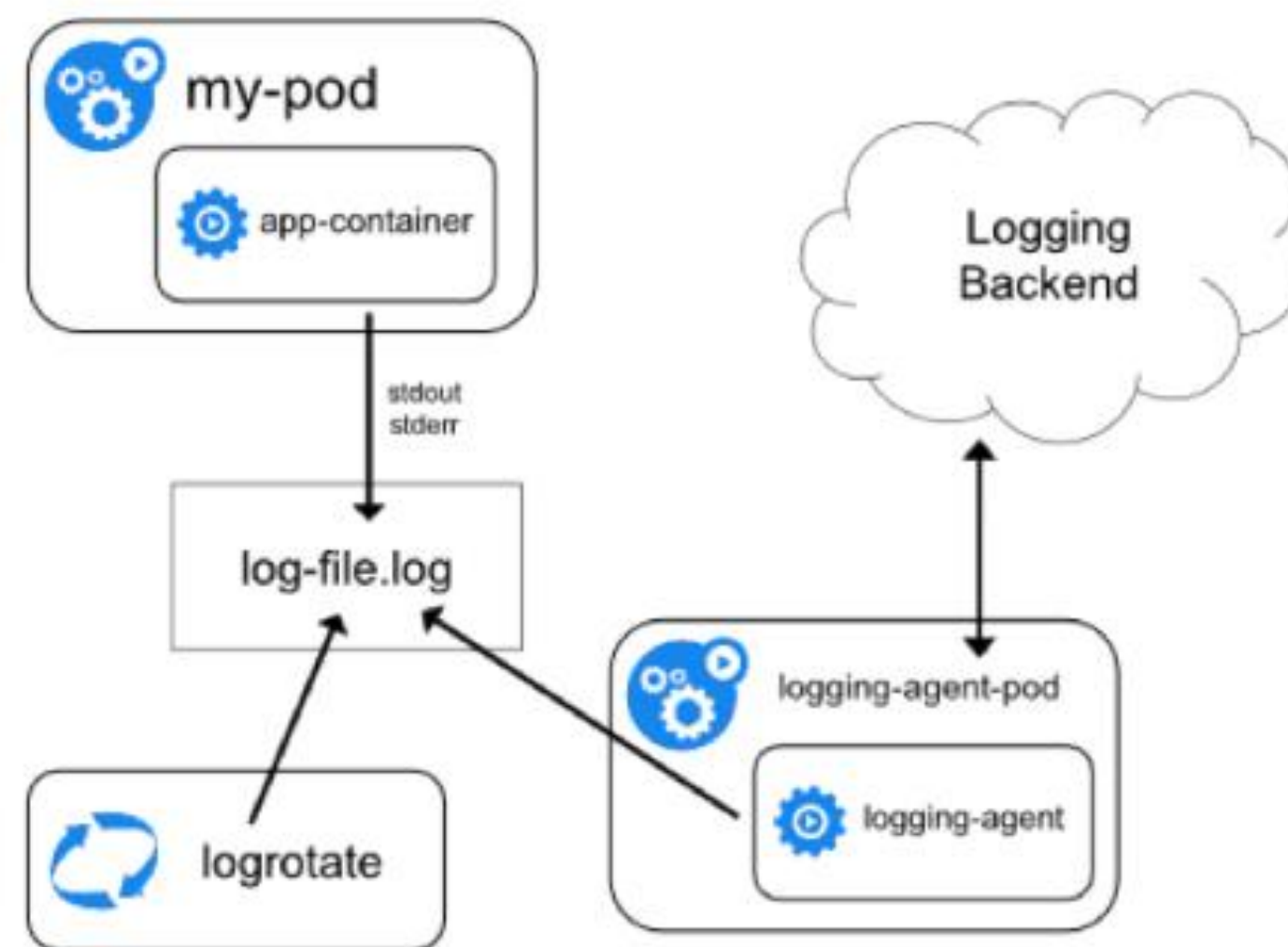
- 日志的采集
- 日志的展示

日志的收集模式

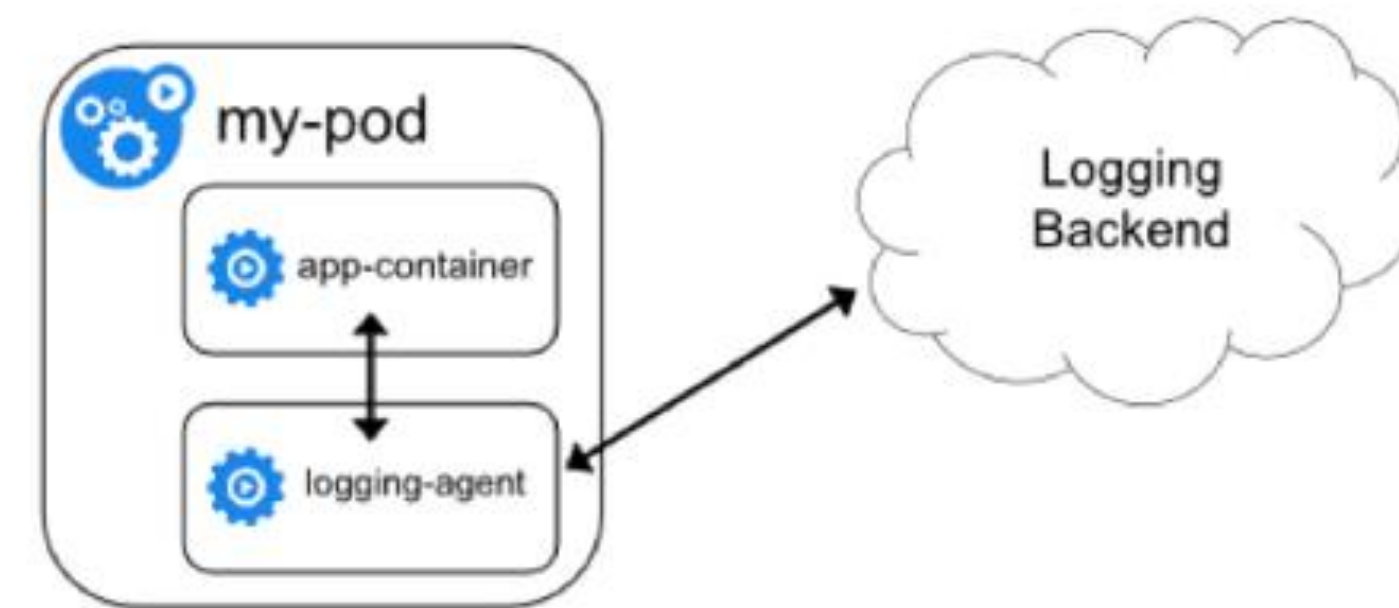
阿里云的日志收集方案： sidecar 容器收集日志 + CRD 声明配置



原生方式



DaemonSet方式



Sidecar方式

日志的收集模式

配置错误会导致日志服务无法正常采集当前应用日志
后续版本会提供验证日志是否配置正常的功能
如果日志配置正确, 5分钟内, 可以在日志搜索界面看到采集到的日志
日志使用通用格式打印之后, 可以获得链路日志/用户关联日志等功能。
接入参考：[日志标准格式](#)

日志目录	<div>/app/logs</div> <div>系统会采集目录下的*.log*文件 填写绝对路径 范例： 程序配置打印日志路径到相对目录 logs 填写 /app/logs 可登录终端跳转到日志目录之后执行 pwd 命令查询</div>
行首时间格式	<div>yyyy-MM-dd HH:mm:ss.sss</div> <div>用于识别多行日志，非常重要 仅填写日志中涉及的时间相关的格式内容, 即使行首包含了[] ANSI颜色 等特殊字符, 也仅填写日期时间格式。 日志样例： [2018-10-15 11:12:00.844][DubboServerHandler-10.46.231.132:65012-thread-193][DEBUG][jdbc.sqltiming:365] org.apache.ibatis.executor.statement.PreparedStatementHandler.quer (PreparedStatementHandler.java:63) 填写</div>

```
apiVersion: log.alibabacloud.com/v1alpha1
kind: AliyunLogConfig
metadata:
  # your config name, must be unique in you k8s cluster
  name: simple-file-example
spec:
  # logstore name to upload log
  logstore: k8s-file
  # logtail config detail
  logtailConfig:
    # log file's input type is 'file'
    inputType: file
    # logtail config name, should be same with [metadata.name]
    configName: simple-file-example
    inputDetail:
      # 极简模式日志, logType设置为"common_reg_log"
      logType: common_reg_log
      # 日志文件夹
      logPath: /data/logs/app_1
      # 文件名, 支持通配符, 例如log_*.log
      filePattern: simple.LOG
      # 采集容器内的文件, dockerFile flag设置为true
      dockerFile: true
      # only collect container with "ALIYUN_LOGTAIL_USER_DEFINED_ID" in docker env con
      dockerIncludeEnv:
        ALIYUN_LOGTAIL_USER_DEFINED_ID: ""
```

日志的展示模式

链路日志

时间

2019.01.20 10:41:45.0

请输入关键字，用英文逗号隔开

倒序显示

选择时间：2019-01-20 10:41:45 ~ 2019-01-20 10:56:45

日志源：终端, /app/logs/all-logs...

LogLevel: TRACE, DEBUG, INFO,...

时间	AppId	实例ID	级别	线程名称	日志内容	代码文件：行数
2019.01.20 10:41:45.001	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- --> GET http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events http/1.1	HttpLoggingInterceptor:160
2019.01.20 10:41:45.001	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- --> GET http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events http/1.1	HttpLoggingInterceptor:160
2019.01.20 10:41:44.997	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- <-- 200 OK http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/pods (8ms, unknown-length body)	HttpLoggingInterceptor:222
2019.01.20 10:41:45.000	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- 应用列表接口，查询k8s pod信息总耗时 11 ms	AppListService:103
2019.01.20 10:41:45.000	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- 应用列表接口，查询k8s pod信息总耗时 11 ms	AppListService:103
2019.01.20 10:41:45.001	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- --> GET http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events http/1.1	HttpLoggingInterceptor:160
2019.01.20 10:41:45.001	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- --> GET http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events http/1.1	HttpLoggingInterceptor:160
2019.01.20 10:41:45.003	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- <-- 200 OK http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events (2ms, 140-byte body)	HttpLoggingInterceptor:222
2019.01.20 10:41:45.004	鲸云后端	172.30.1.1	INFO	http-nio-8080-exec-6	- cpu负载 0.16, cpu核数 8.0	RuntimeInfoLogInterceptor:33
2019.01.20 10:41:45.004	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- 应用列表接口，查询k8s event信息总耗时 4 ms	AppVersionEventService:131
2019.01.20 10:41:45.004	鲸云后端	172.30.1.1	INFO	http-nio-8080-exec-6	- jvm heap init 956 M, used 585 M, committed 998 M, max 998 M	RuntimeInfoLogInterceptor:34
2019.01.20 10:41:45.004	鲸云后端	172.30.1.1	DEBUG	http-nio-8080-exec-6	- <-- 200 OK http://k8s-api.gz.cvte.cn/api/v1/namespaces/mis-dev/events (2ms, 140-byte body)	HttpLoggingInterceptor:222
2019.01.20 10:41:45.005	鲸云后端	172.30.1.1	INFO	http-nio-8080-exec-6	- jvm heap init 956 M, used 585 M, committed 998 M, max 998 M	RuntimeInfoLogInterceptor:34
2019.01.20 10:41:45.005	鲸云后端	172.30.1.1	INFO	http-nio-8080-exec-6	- cpu负载 0.16, cpu核数 8.0	RuntimeInfoLogInterceptor:33

发布流量控制

采用阿里云的Ingress Controller实现预发功能

应用版本预发 / 卜下线

下线

请选择发布类型

预发布

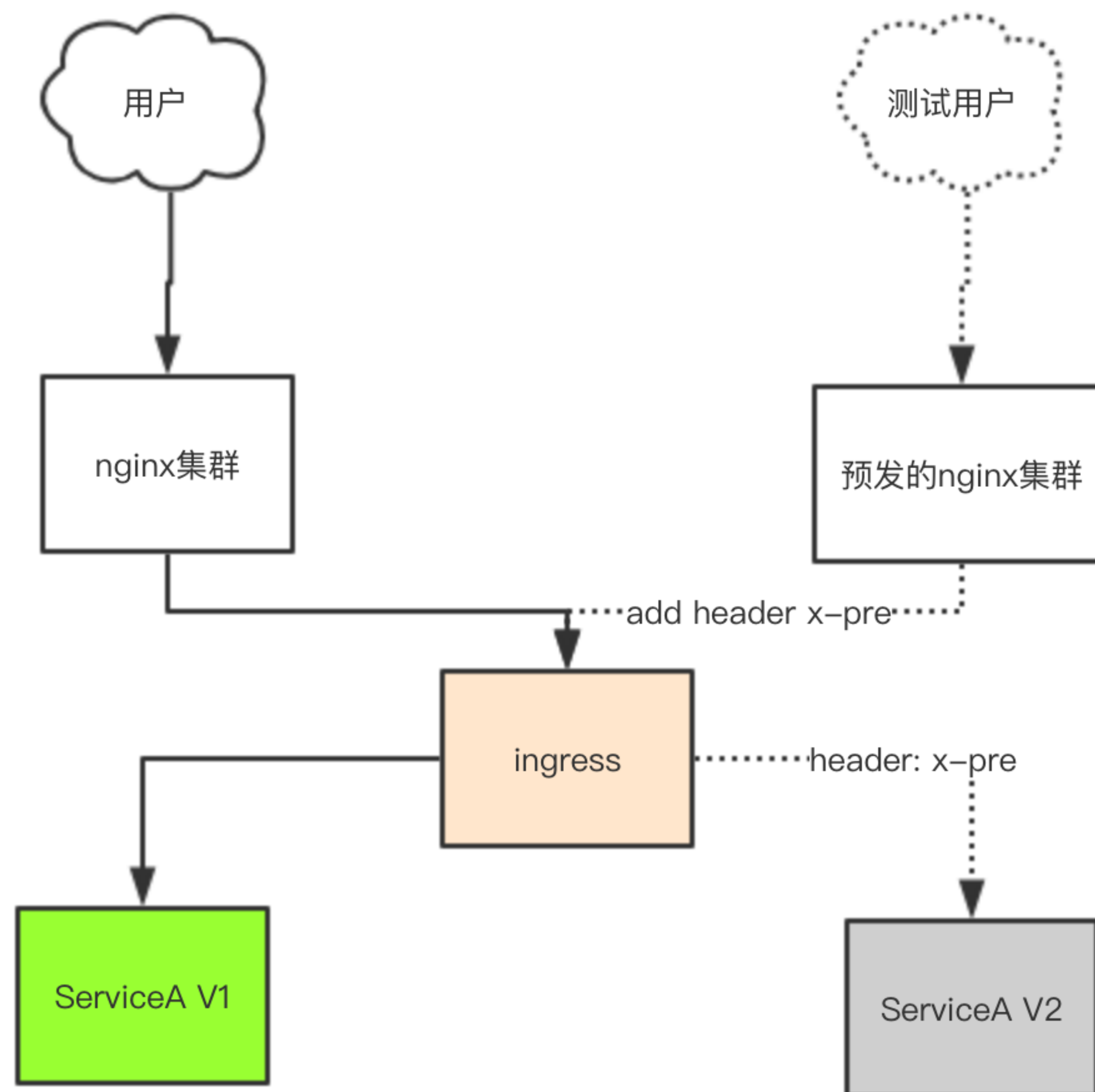
上线

说明：

- 1、上线：用户能访问
- 2、预发布：用户访问不到，测试人员可通过设置域名定向访问
- 3、下线：无论如何都访问不到

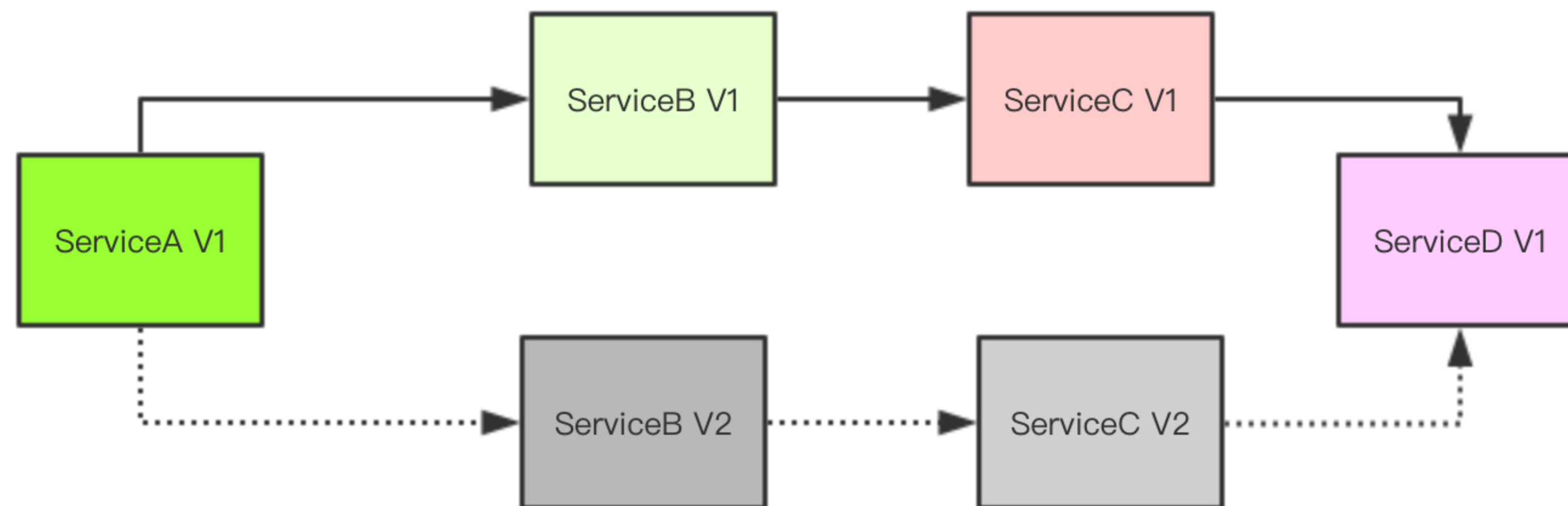
取消

确定



发布流量控制

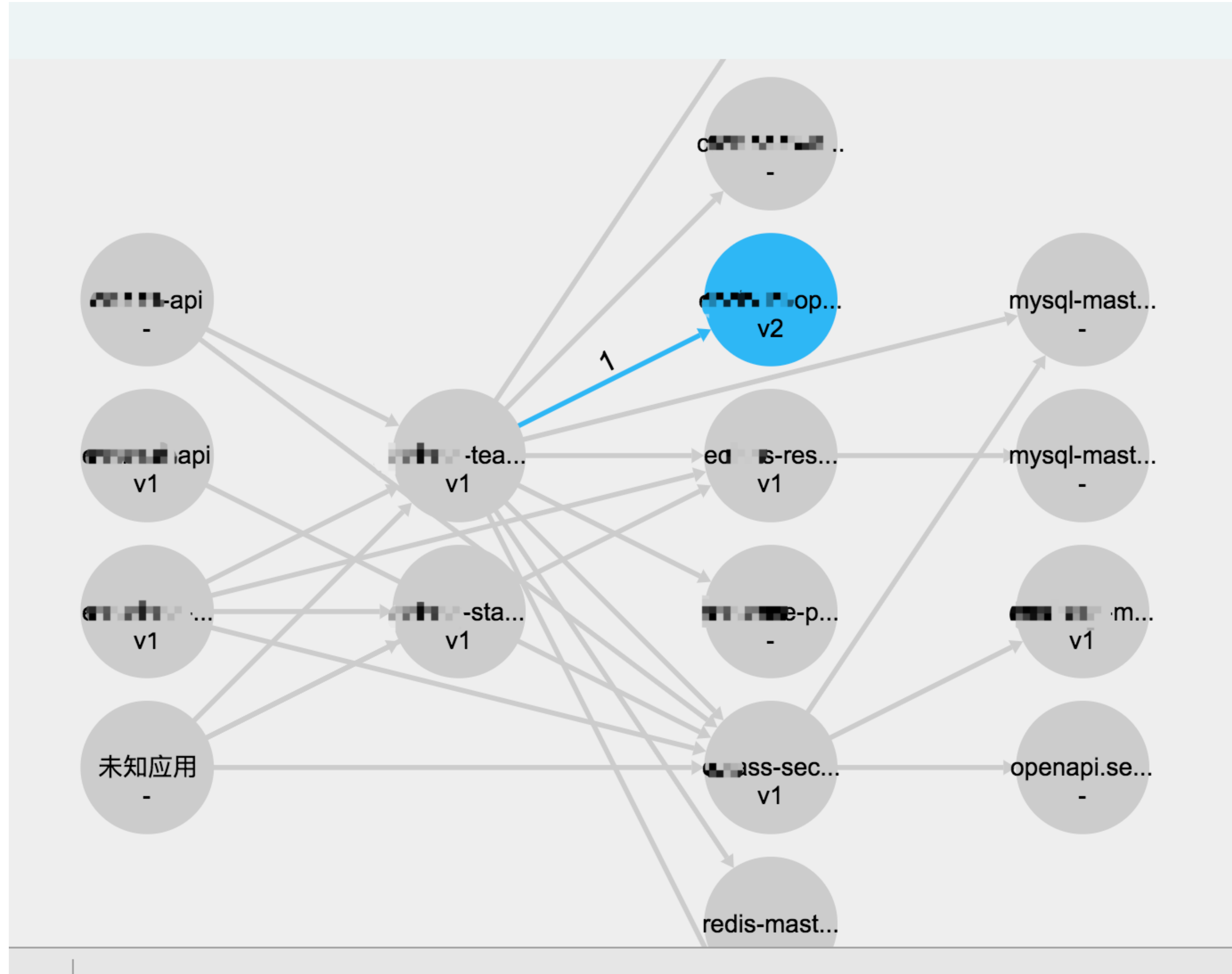
- 微服务间的调用
- dubbo 非http协议的调用



解决方案

- 使用APM 的代码注入，自动传递header
- 使用环境变量+ Dubbo Group机制分组

发布流量控制



监控

- 主机监控采用zabbix
- 容器采用node-exporter
- 应用内性能监控采用APM



k8s 事件的收集和告警

- 监听event 变化，并收集持久化
- 监听endpoint 的变化，并告警

节点管理

ingress-nginx管理

k8s事件

k8s 对象

设置

件级别

全部

▼

询时间

1 周

▼

2019-01-13 08:14:12 ~ 2019-01-20 08:14:12

关键字:

请输入关键字

查询

序号	事件级别	事件信息	事件原因	汇报者	来源机器	最后出现时间	操作
1	警告	Readiness probe failed: dial tcp 2.20.31.124:8080: getsockopt: connection refused	Unhealthy	kubelet	cn-hangzhou.i-bp1...	昨天 [01-19] 23:42:31	详情
2	正常	Started container	Started	kubelet	cn-hangzhou.i-bp1...	昨天 [01-19] 23:41:49	详情
3	正常	Created container	Created	kubelet	cn-hangzhou.i-bp1...	昨天 [01-19] 23:41:47	详情
4	正常	pulling image	Pulling	kubelet	cn-hangzhou.i-bp1...	昨天 [01-19] 23:41:46	详情
5	正常	Successfully pulled image	Pulled	kubelet	cn-hangzhou.i-bp1...	昨天 [01-19] 23:41:46	详情

【阿里云鲸云】应用不健康

应用名: **[REDACTED]**

所属项目: [MIS-PRO] / [MIS-BASIS]

不健康实例数: 1

总实例数: 2

节点 ip: **[REDACTED]**

容器退出: ExitCode 137, 原因

OOMKilled

时间: 01-19 23:41:46

总结

- k8s的引入，极大增加了运维规模
- 结合社区的力量，加速DevOps的落地
- 前路漫漫，我们还招人



坤丰 
广东 广州



扫一扫上面的二维码图案，加我微信