构建高效私有云平台

— 今日头条私有云平台架构设计

夏绪宏

关于我

- 夏绪宏 @reeze
- 今日头条研发架构负责人
- 基础设施平台
- PHP Committer\LAMP

大纲

01. 私有云、公有云在头条

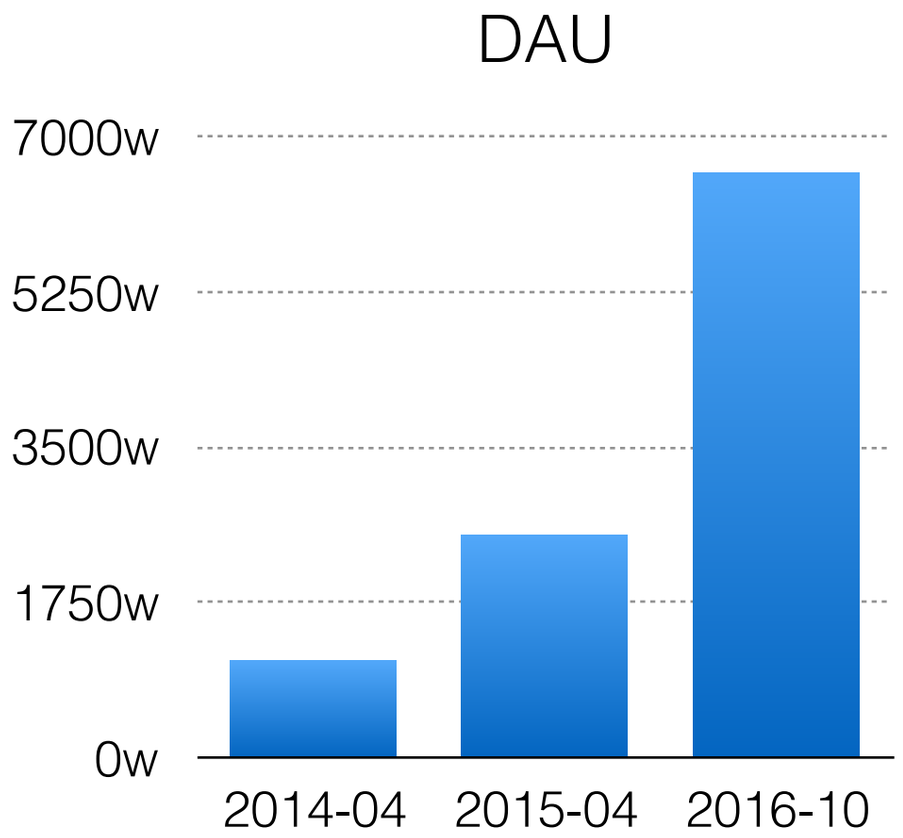
02. 头条私有云平台架构设计

03. 遇到问题以及未来的规划

01. 私有云、公有云在头条

关于今日头条

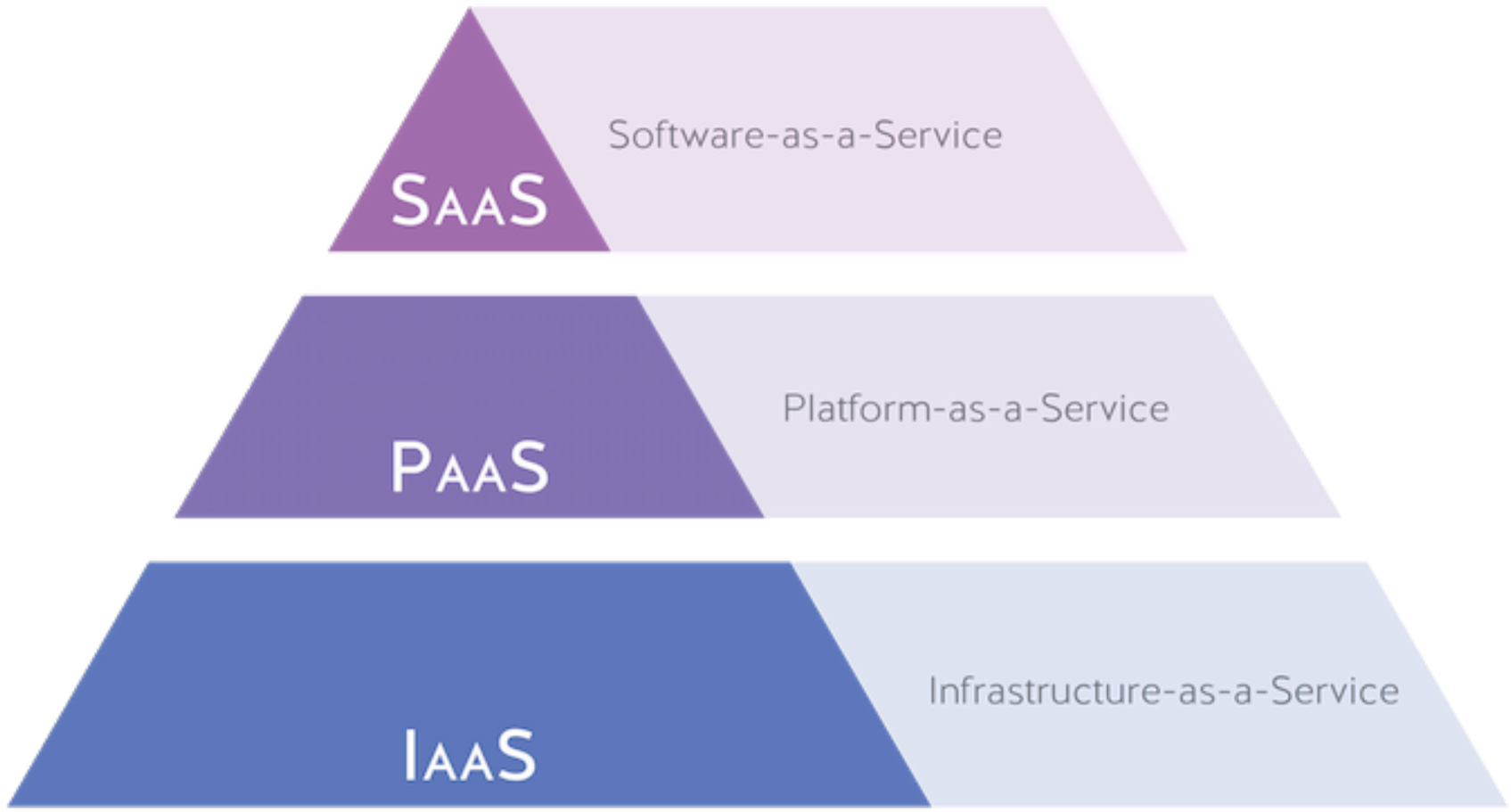
- 6亿用户
- 6600W DAU
- 76分钟日使用时长
- 迭代部署： **670**+次/天



今日头条&云

- IaaS公有云：应对突发和计算资源需求
  - 推送场景，峰值效应，带宽占用大
  - 国际化服务
- SaaS服务
  - 服务质量监控：云监控服务
  - 第三方统计服务
  - CDN网络
  - etc…

云计算设施



http://nordicapis.com/living-in-the-cloud-stack-understanding-saas-paas-and-iaas-apis/

公有云，私有云

公司规模	规模小		规模大	
	公有云	私有云	公有云	私有云
弹性	**好**	差	**好**	**好**
可控性	弱	**强**	弱	**好**
成本	**低**	高	高	**低**

02. 头条私有云平台架构设计

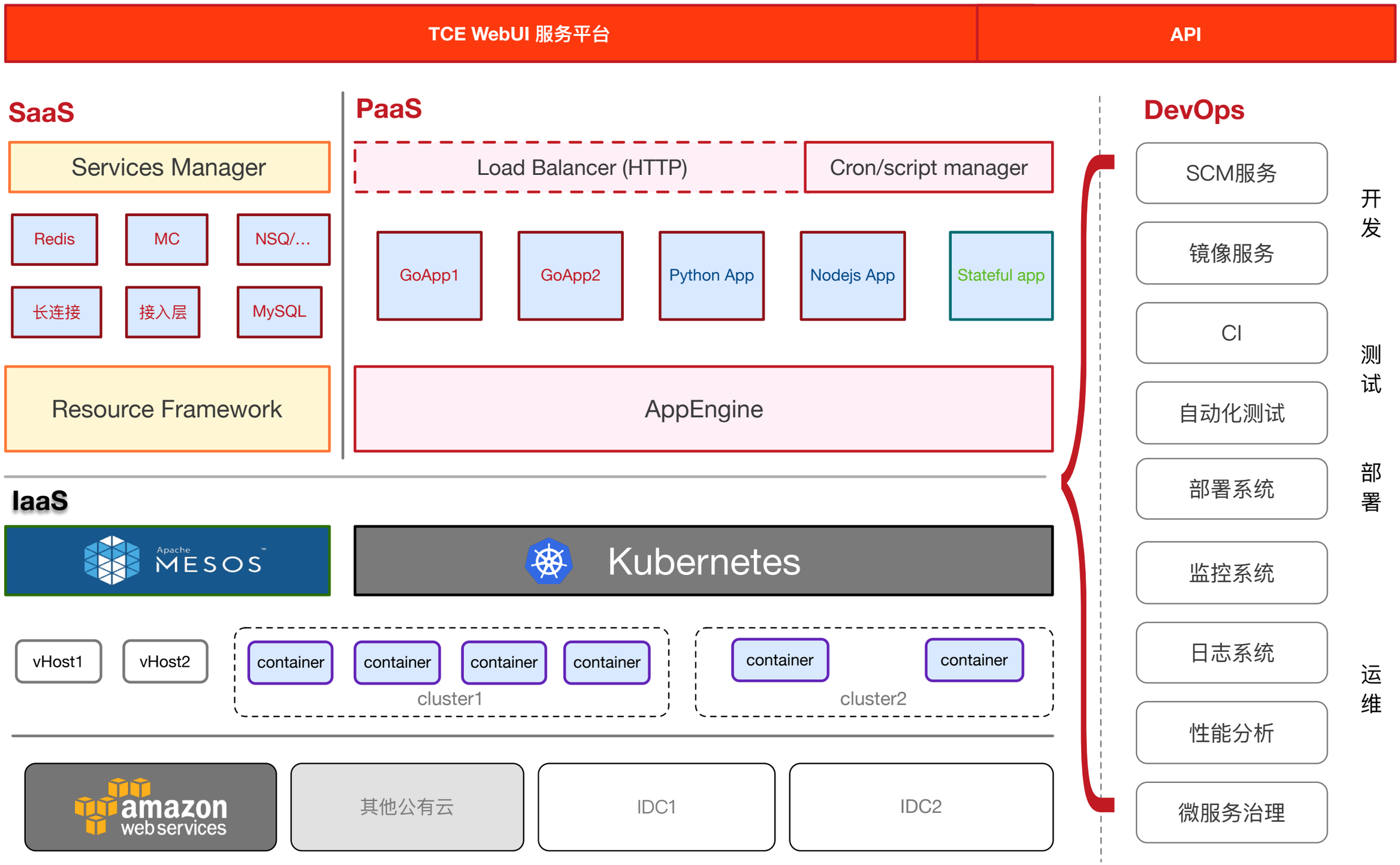
平台设计目标

- **目标**：高效的在线服务研发PaaS平台
- **思路**：
  - 松耦合，支持不同类型业务
  - 理解微服务
  - 构建完善周边SaaS服务
  - 足够的弹性，混合云支持

目前的进展

- **TCE：** Toutiao Compute Engine
- **进展：**
  - 2016–05 启动 2016–10 上线
  - 120+服务迁移，继续迁移中
  - 扩容效率：10倍提升，< 1分钟

总体架构


TCE WebUI 服务平台
API
SaaS
Services Manager
Redis
MC
NSQ/...
长连接
接入层
MySQL
Resource Framework
PaaS
Load Balancer (HTTP)
Cron/script manager
GoApp1
GoApp2
Python App
Nodejs App
Stateful app
AppEngine
IaaS
Apache MESOS
Kubernetes
vHost1
vHost2
container
container
container
container
cluster1
container
container
cluster2
amazon web services
其他公有云
IDC1
IDC2
DevOps
SCM服务
镜像服务
CI
自动化测试
部署系统
监控系统
日志系统
性能分析
微服务治理
开发
测试
部署
运维

TCE平台

- **技术方案**
  - 技术选型、网络模型、服务发现
  - 日志收集、容器的使用、弹性调度、etc...
- **DevOps研发基础设施**
  - 开发测试、部署上线
  - 微服务的支持

技术选型

- PaaS
  - 定制需求多
  - 没有合适的开源方案，自研
- IaaS
  - Kubernetes
  - Mesos

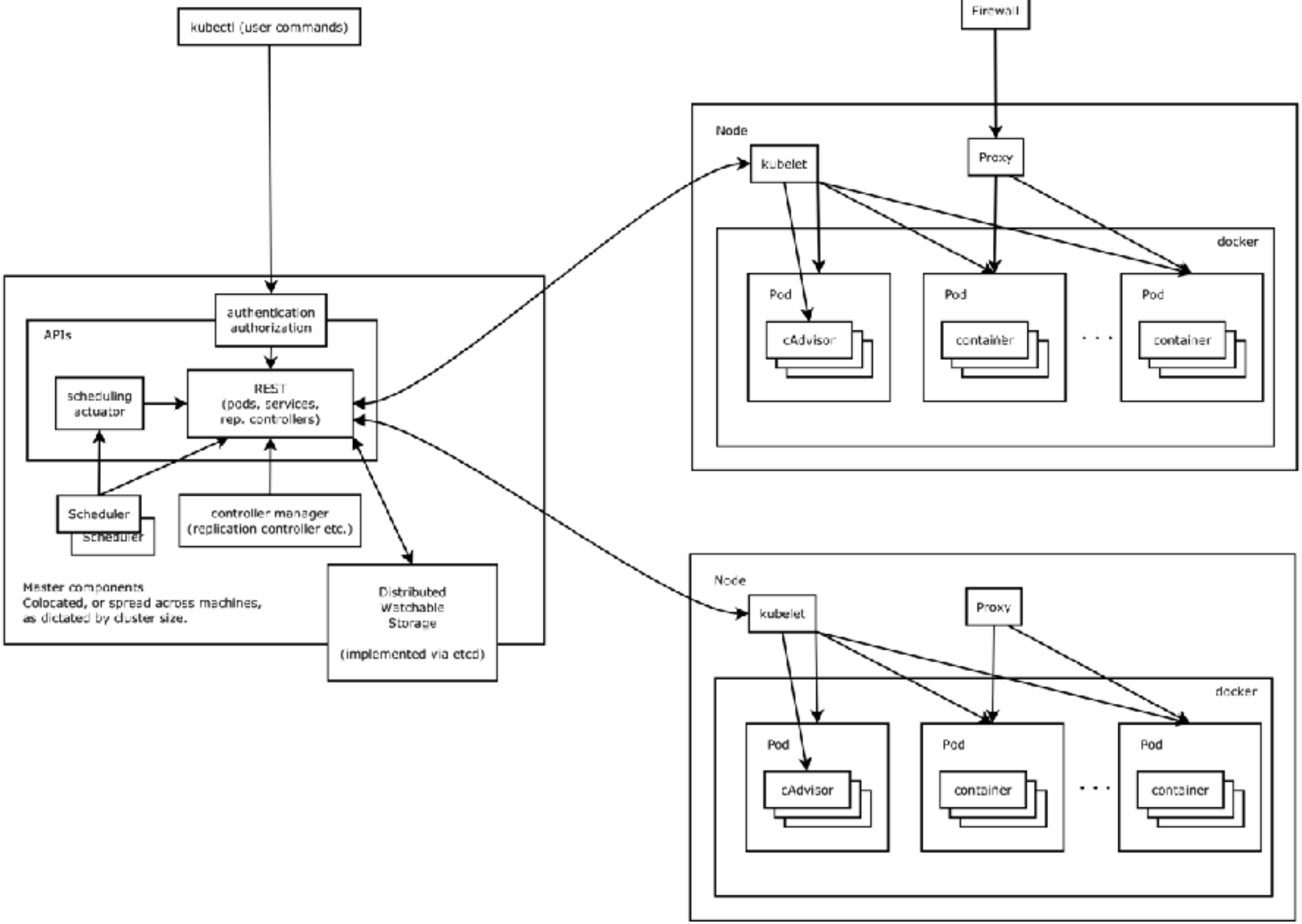
技术选型

	Kubernetes	**Mesos**
编排能力	强	一般
集群规模	千级别	massive
有状态服务	不太好	比较好
弹性	好	好
自愈能力	支持	支持

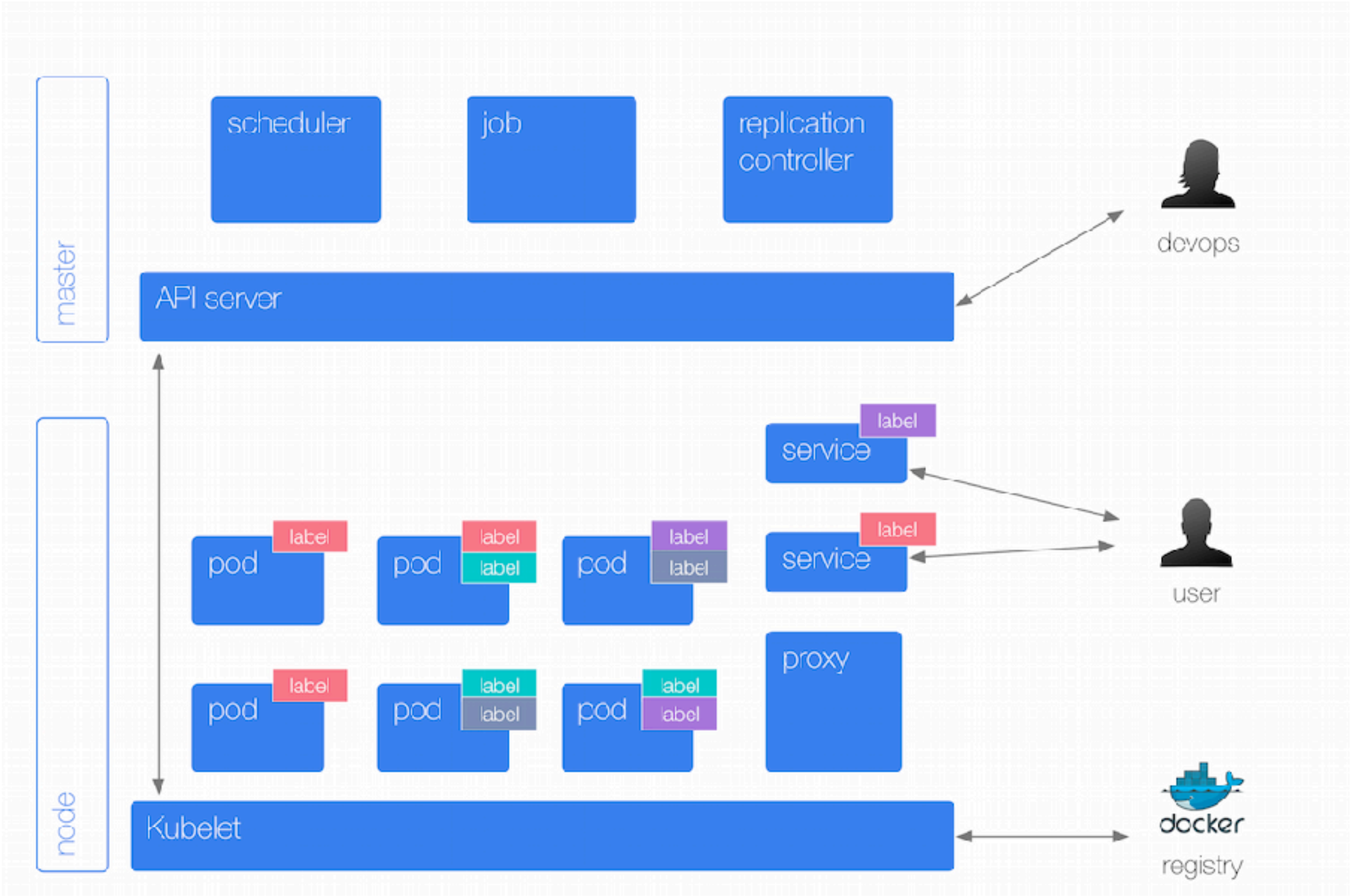
技术选型

- IaaS层设计
  - 基础设施中立
  - 不强绑定底层IaaS设施
  - 通用计算服务

Kubernetes


kubectl (user commands)
Internet
Firewall
Node
kubelet
Proxy
docker
Pod
cAdvisor
Pod
container
Pod
container
APIs
authentication
authorization
scheduling
actuator
REST
(pods, services,
rep. controllers)
Scheduler
controller manager
(replication controller etc.)
Master components
Colocated, or spread across machines,
as dictated by cluster size.
Distributed
Watchable
Storage
(implemented via etcd)
Node
kubelet
Proxy
docker
Pod
cAdvisor
Pod
container
Pod
container

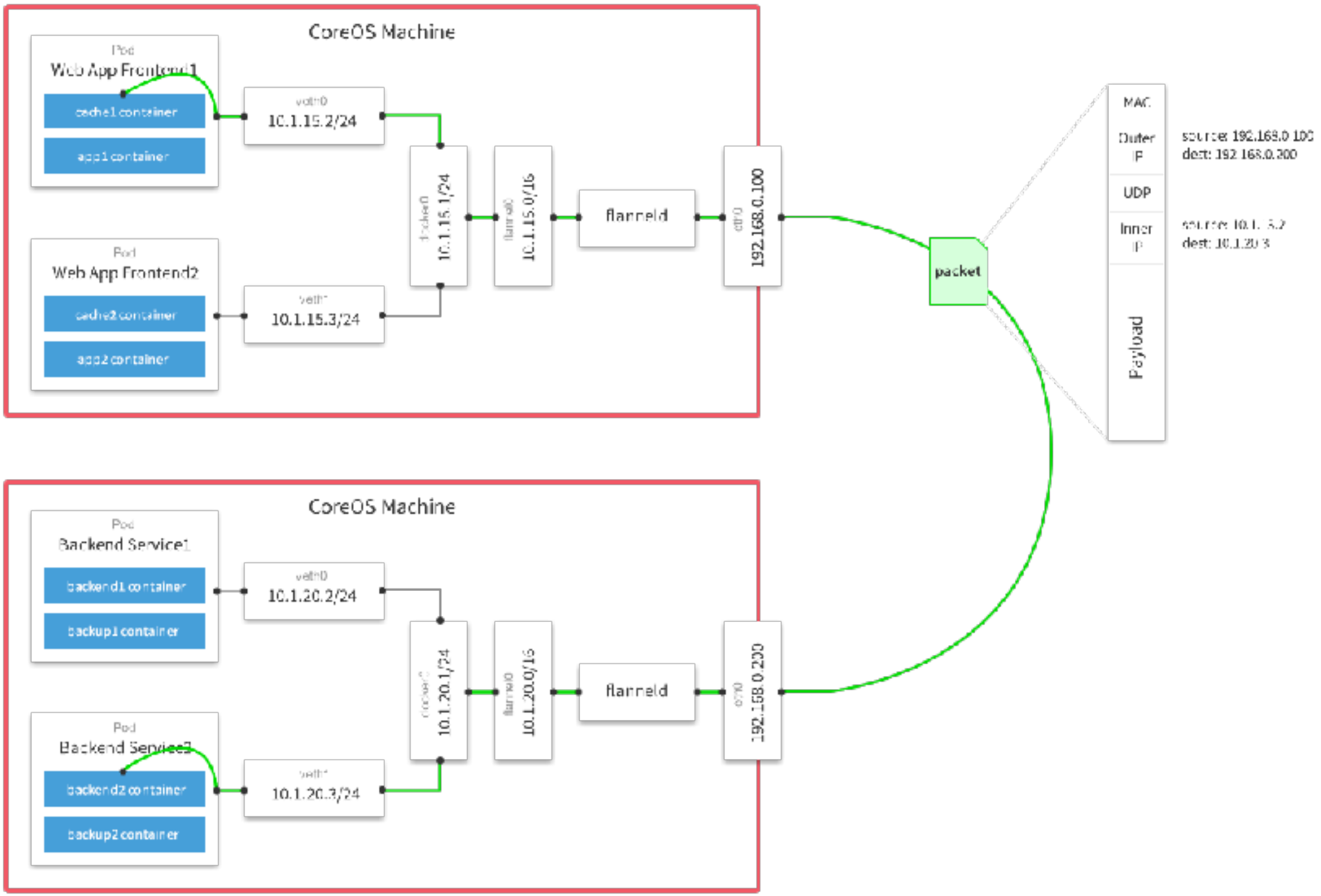
Kubernetes


master
scheduler
job
replication controller
devops
API server
label
service
label
service
user
label
pod
label
pod
label
pod
label
pod
label
pod
label
pod
proxy
node
Kubelet
docker
registry

Kubernetes网络模型

- **Container reach container**
  - all containers can communicate with all other containers without NAT
- **Node reach container**
  - all nodes can communicate with all containers (and vice-versa) without NAT
- **IP addressing**
  - Pod in cluster can be addressed by its IP

Flannel


CoreOS Machine
Pod
Web App Frontend1
cache1 container
app1 container
veth0
10.1.15.2/24
Pod
Web App Frontend2
cache2 container
app2 container
veth1
10.1.15.3/24
cbr0
10.1.15.1/24
flannel0
10.1.15.0/16
flanneld
eth0
192.168.0.100
packet
MAC
Outer IP
UDP
Inner IP
Payload
source: 192.168.0.100
dest: 192.168.0.200
CoreOS Machine
Pod
Backend Service1
backend1 container
backup1 container
veth0
10.1.20.2/24
Pod
Backend Service2
backend2 container
backup2 container
veth1
10.1.20.3/24
cbr0
10.1.20.1/24
flannel0
10.1.20.0/16
flanneld
eth0
192.168.0.200

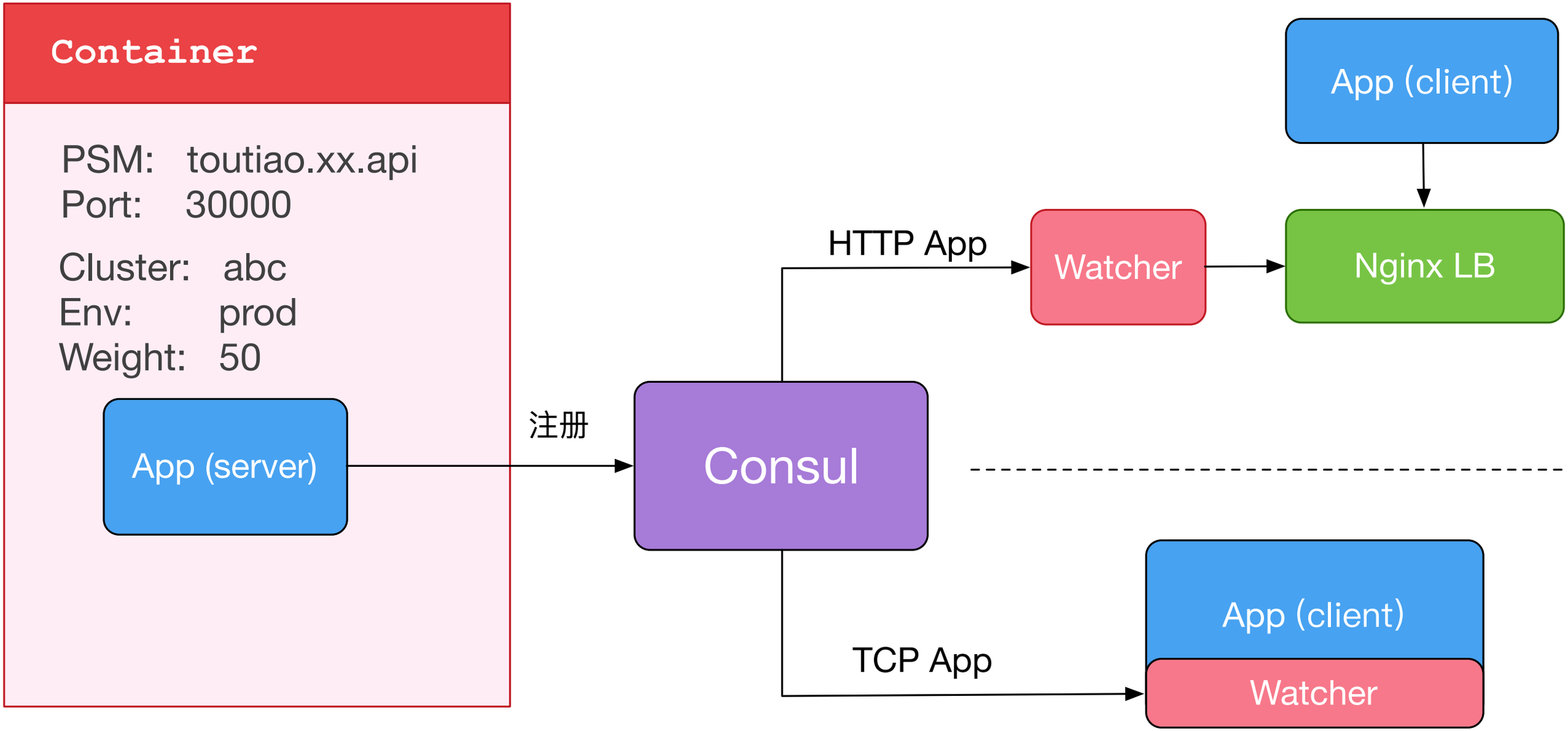
我们的方案

- 使用Flannel
- 不直接使用虚拟子网
- 为每个实例expose随机端口

服务发现

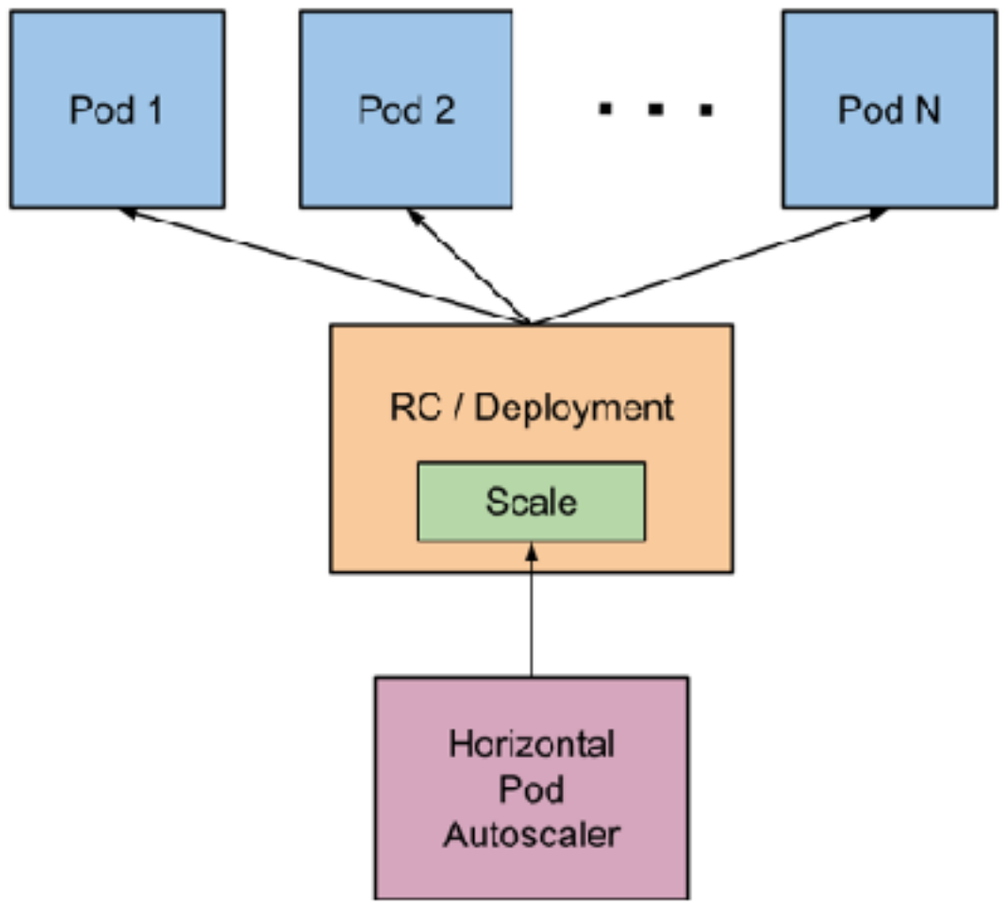
- Kubernetes
  - Service: Cluster IP
  - DNS
- 使用Consul自己做服务发现
  - 跨集群问题，虚拟网络和已有网络的互通问题
  - 性能问题
  - 减少层次，问题定位效率
  - 我们一直在用Consul

服务发现


Container
PSM: toutiao.xx.api
Port: 30000
Cluster: abc
Env: prod
Weight: 50
App (server)
注册
Consul
App (client)
HTTP App
Watcher
Nginx LB
TCP App
App (client)
Watcher

弹性调度

- 提升应对突发流量的能力
- 资源利用率提升
- 基于CPU metrics
- 后续根据业务指标扩容



容器的使用方式

- 使用init进程守护(systemd)
- 好处:
    - panic之类的问题的可以更快的重启
    - 和物理机的守护方式一致
    - 支持容器内应用重启或临时更新操作
- 坏处:
    - 服务一直异常可能会影响服务（LB和RPC框架会屏蔽）

容器的使用方式

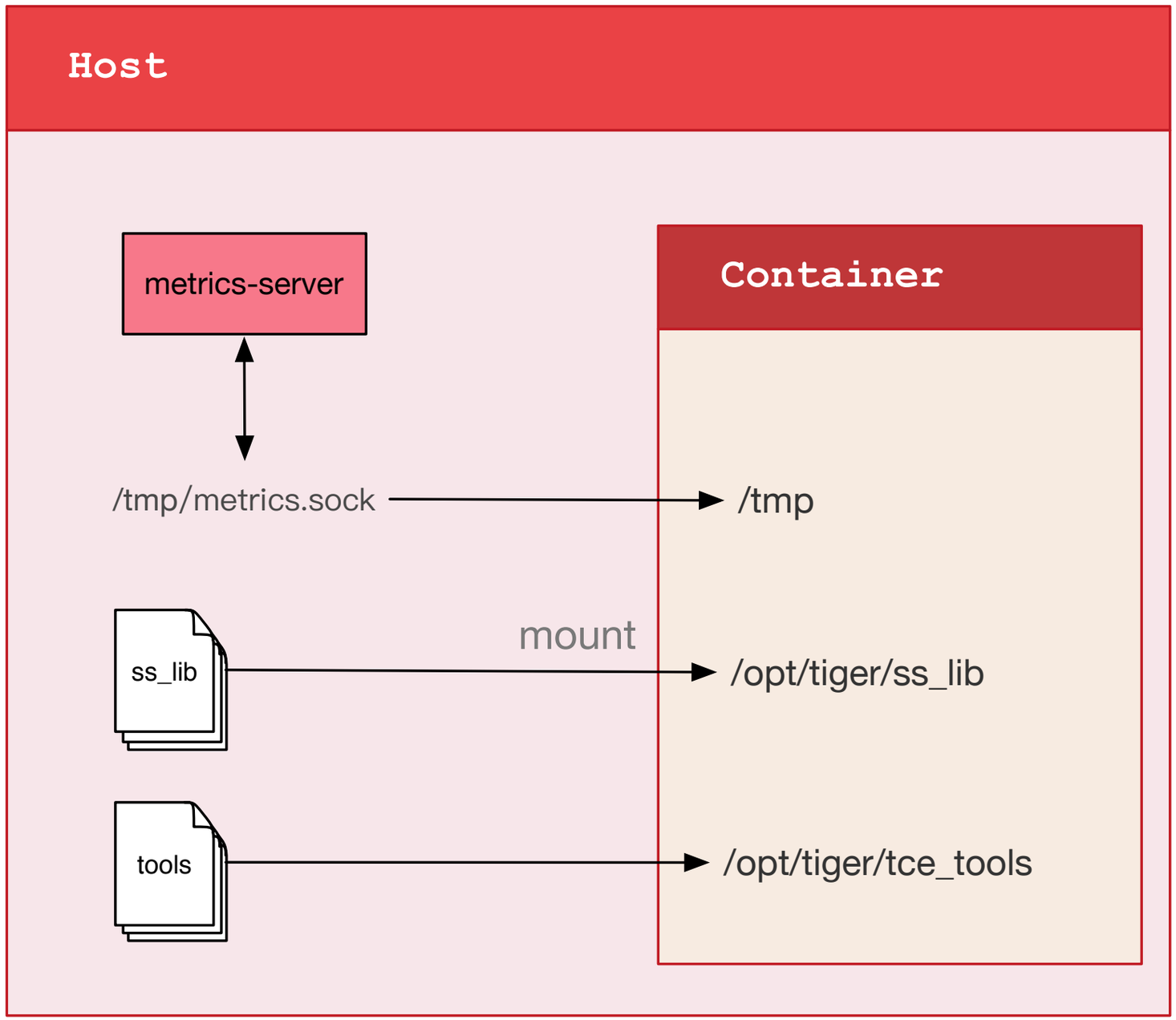
- 不完全自包含
    - 基础服务在容器外运行：各种agent
    - 日志持久化
    - 公共Library共享
- 一种折衷和过渡

容器的使用方式

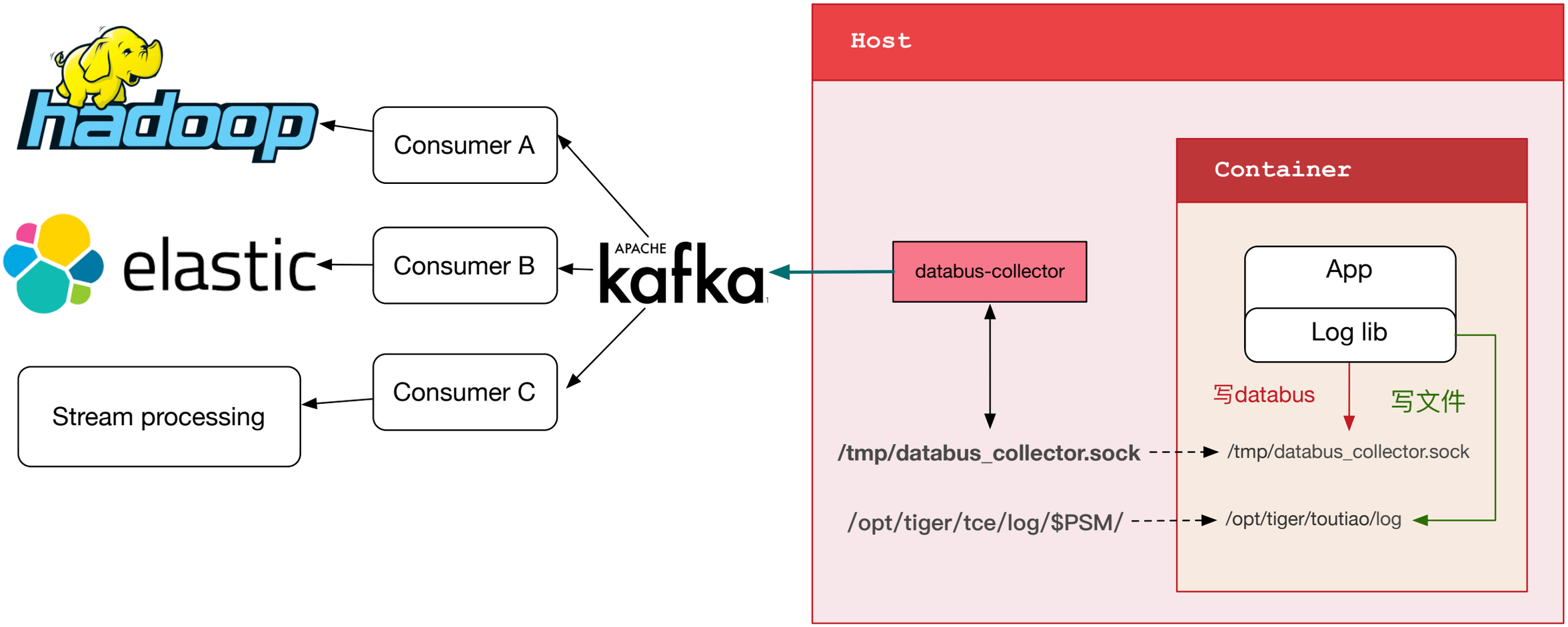
Shared Service

Shared Python Library

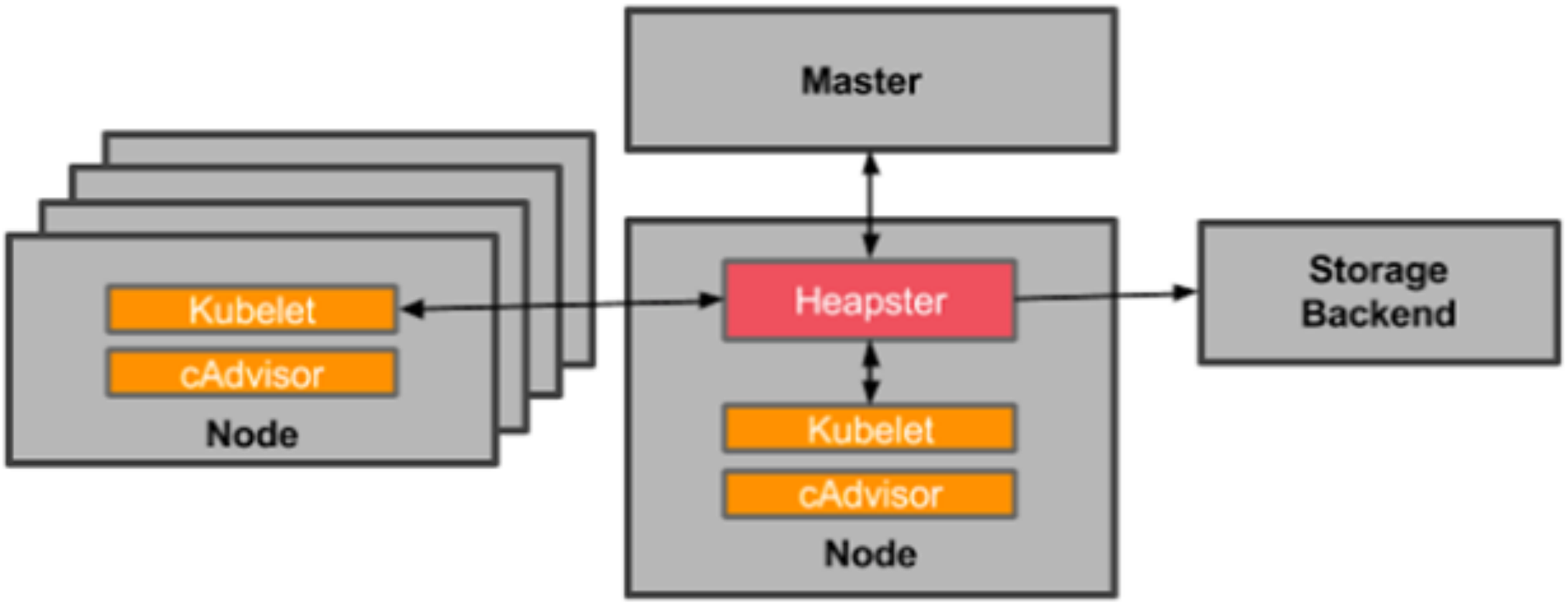
Platform tools



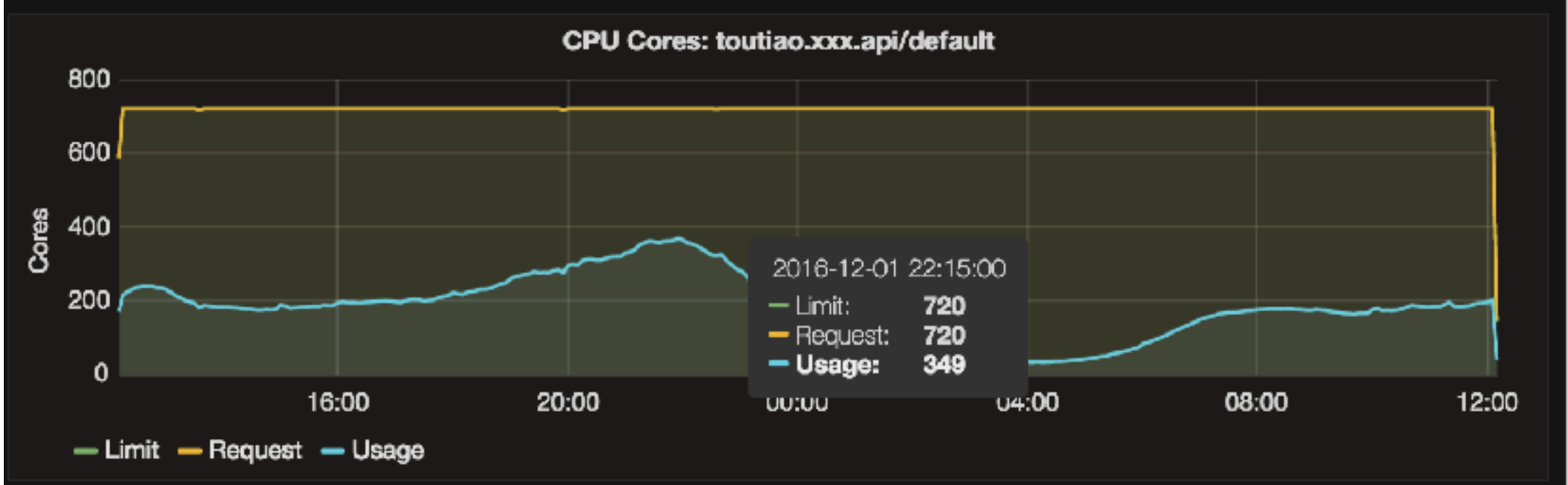
日志收集

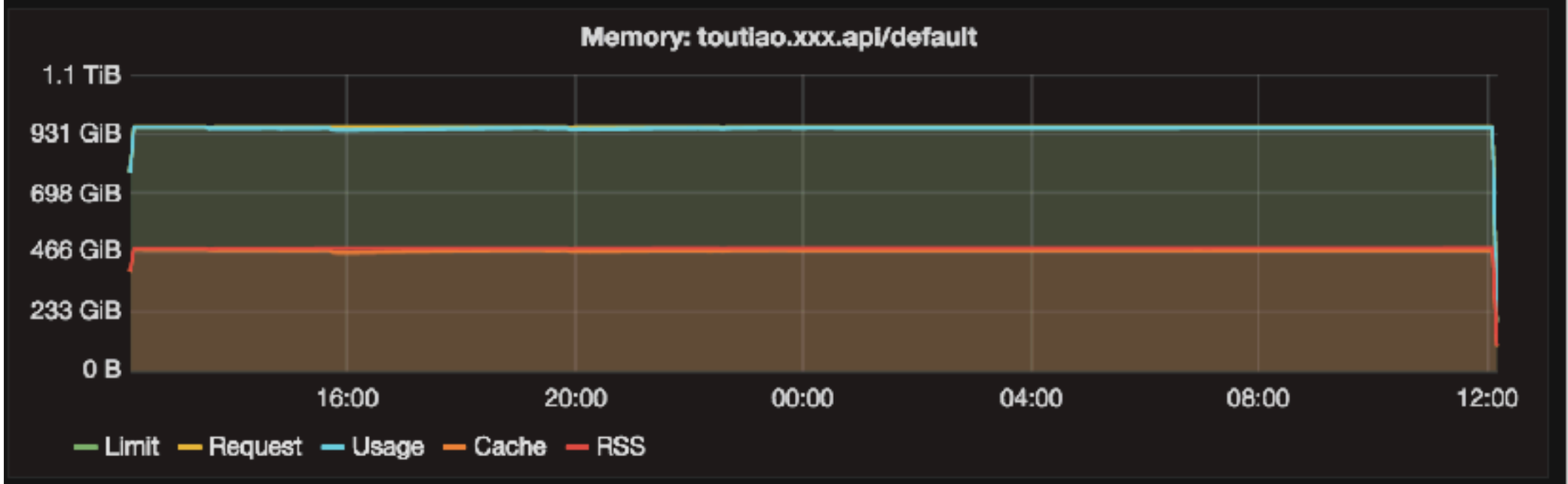

hadoop
elastic
Consumer A
Consumer B
Consumer C
Stream processing
APACHE kafka
Host
Container
databus-collector
App
Log lib
写databus
写文件
/tmp/databus_collector.sock
/tmp/databus_collector.sock
/opt/tiger/tce/log/$PSM/
/opt/tiger/toutiao/log

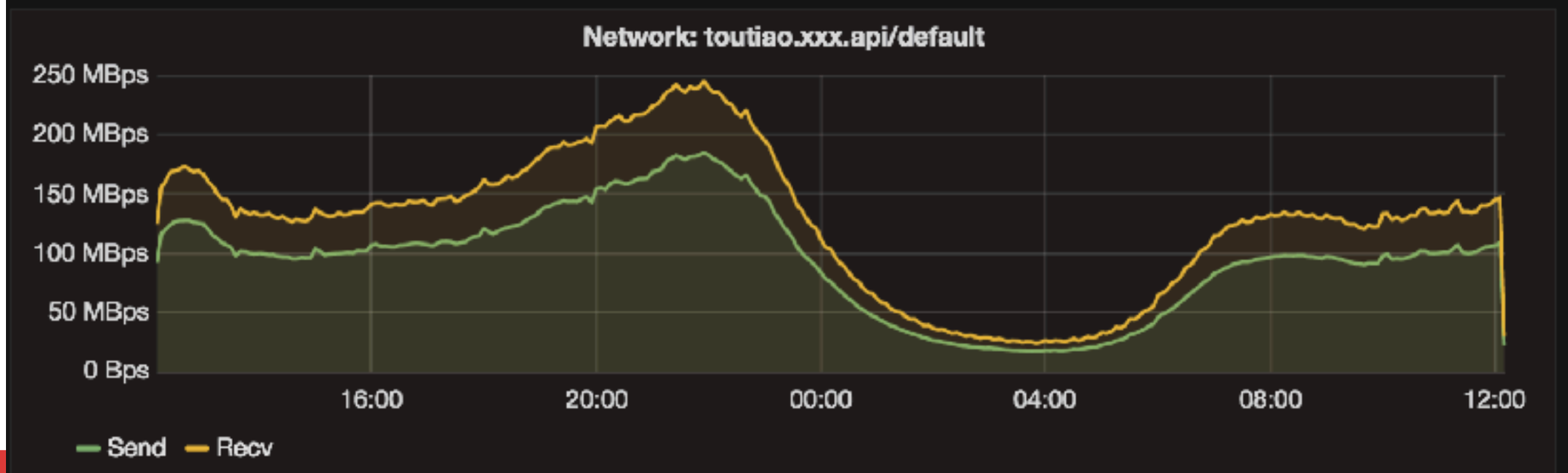
监控


Master
Kubelet
cAdvisor
Node
Heapster
Kubelet
cAdvisor
Node
Storage
Backend

监控

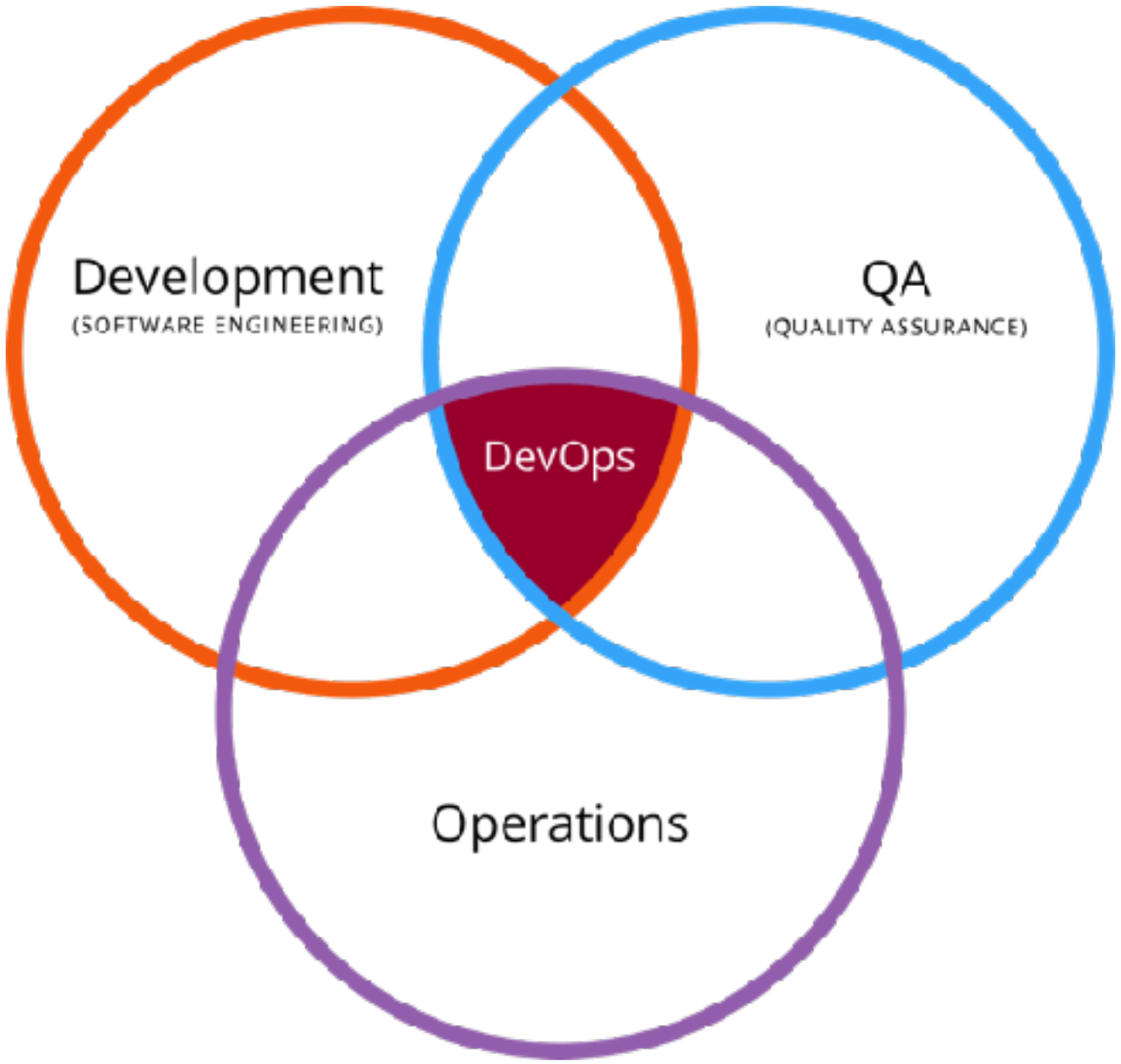

CPU Cores: toutiao.xxx.api/default
800
600
400
200
0
Cores
16:00
20:00
00:00
04:00
08:00
12:00
2016-12-01 22:15:00
Limit: 720
Request: 720
Usage: 349
Limit
Request
Usage


Memory: toutiao.xxx.api/default
1.1 TiB
931 GiB
698 GiB
466 GiB
233 GiB
0 B
16:00
20:00
00:00
04:00
08:00
12:00
Limit
Request
Usage
Cache
RSS


Network: toutiao.xxx.api/default
250 MBps
200 MBps
150 MBps
100 MBps
50 MBps
0 Bps
16:00
20:00
00:00
04:00
08:00
12:00
Send
Recv

02-1. DevOps、微服务

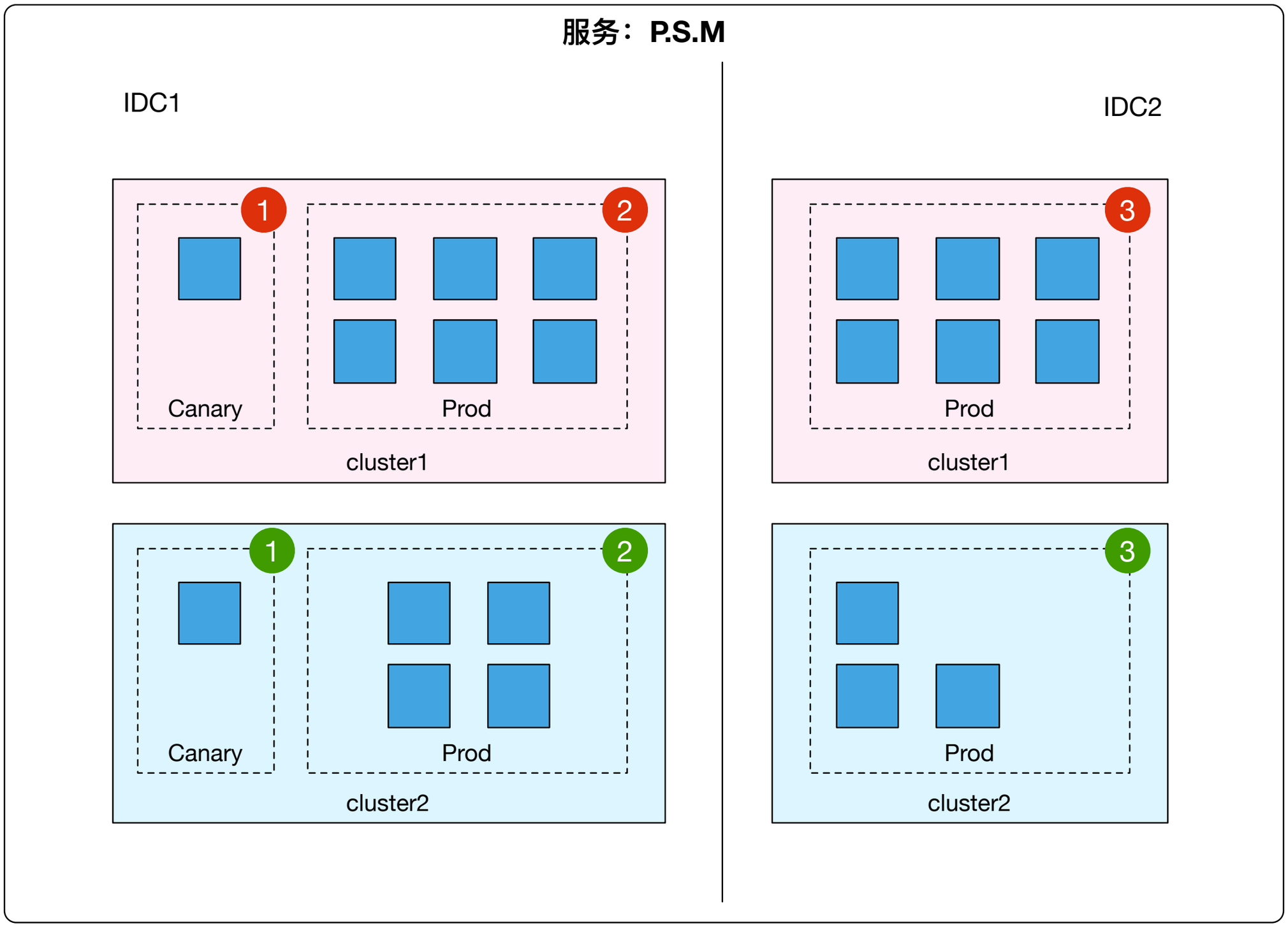
DevOps


Development
(SOFTWARE ENGINEERING)
QA
(QUALITY ASSURANCE)
DevOps
Operations

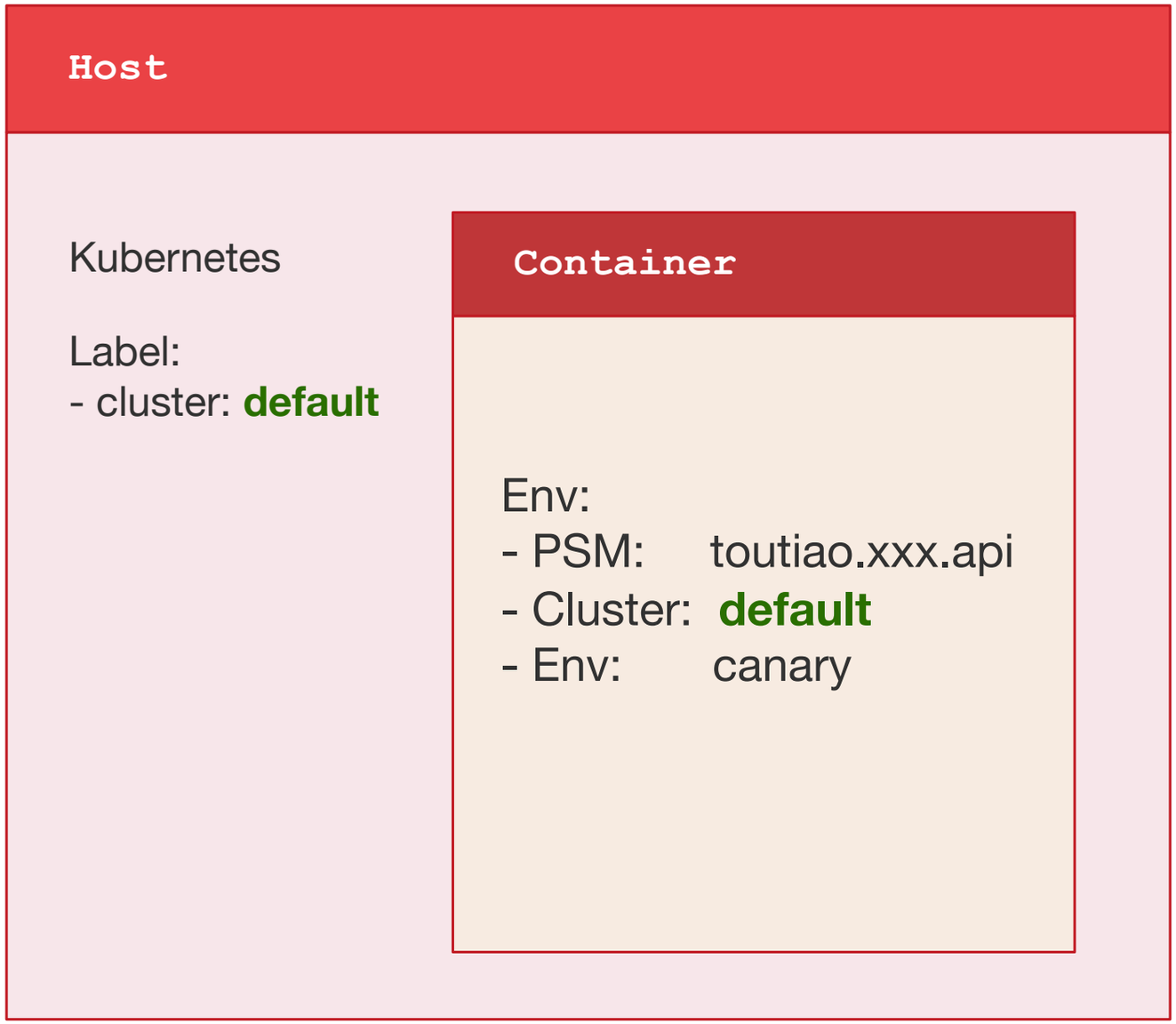
平台眼里的服务(App)

- 理解服务化
- 全局唯一标示：P.S.M
  - {$PRODUCT}.{$SUBSYS}.{$MODULE}
    - 贯穿自动化测试，服务授权，监控，日志等方面
- 分集群：区别对待不同的用户， cluster，env，框架一起理解
- 基本的元信息
- 依赖的程序/lib包信息

平台眼里的服务(App)


服务：P.S.M
IDC1
IDC2
1
2
3
Canary
Prod
Prod
cluster1
cluster1
1
2
3
Canary
Prod
Prod
cluster2
cluster2

平台眼里的服务(App)




```
vv271fa931-4mjhd(toutiao.mtest-subsys.mod @default:prod):/# echo $TCE_
$TCE_ADDR                $TCE_ENV                 $TCE_PRIMARY_PORT
$TCE_CLUSTER             $TCE_INSTANCE_WEIGHT     $TCE_PSM
```

服务信息

服务名称	PSM	所有人	创建时间	操作	报警联系人
backbone	toutiao.xx.xx	zheng	2016-9-29 17:51	请选择操作	设置

集群信息 | 服务详情 | 变更历史

集群	状态	单实例容量	内存使用(HY,LF)	CPU使用(HY,LF)	实例数量(HY,LF)	实例信息	上线单信息	报警设置
default	正在运行	4G | 2核	0.84% 0.24%	0.52% 1.40%	2 | 2	实例	上线单	设置
cluster1	正在运行	4G | 2核	0.38% 0.32%	1.82% 0.02%	2 | 2	实例	上线单	设置
cluster2	正在运行	4G | 2核	0.31% 0.32%	1.04% 0.43%	3 | 3	实例	上线单	设置

服务信息

集群信息 | 服务详情 | 变更历史

服务名称：stream_feed_v2　　PSM：toutiao.xx.xx

所有人：dev1　　所在组：toutiao

基础镜像：toutiao.debian:v1.2　　服务端口：4608(http)

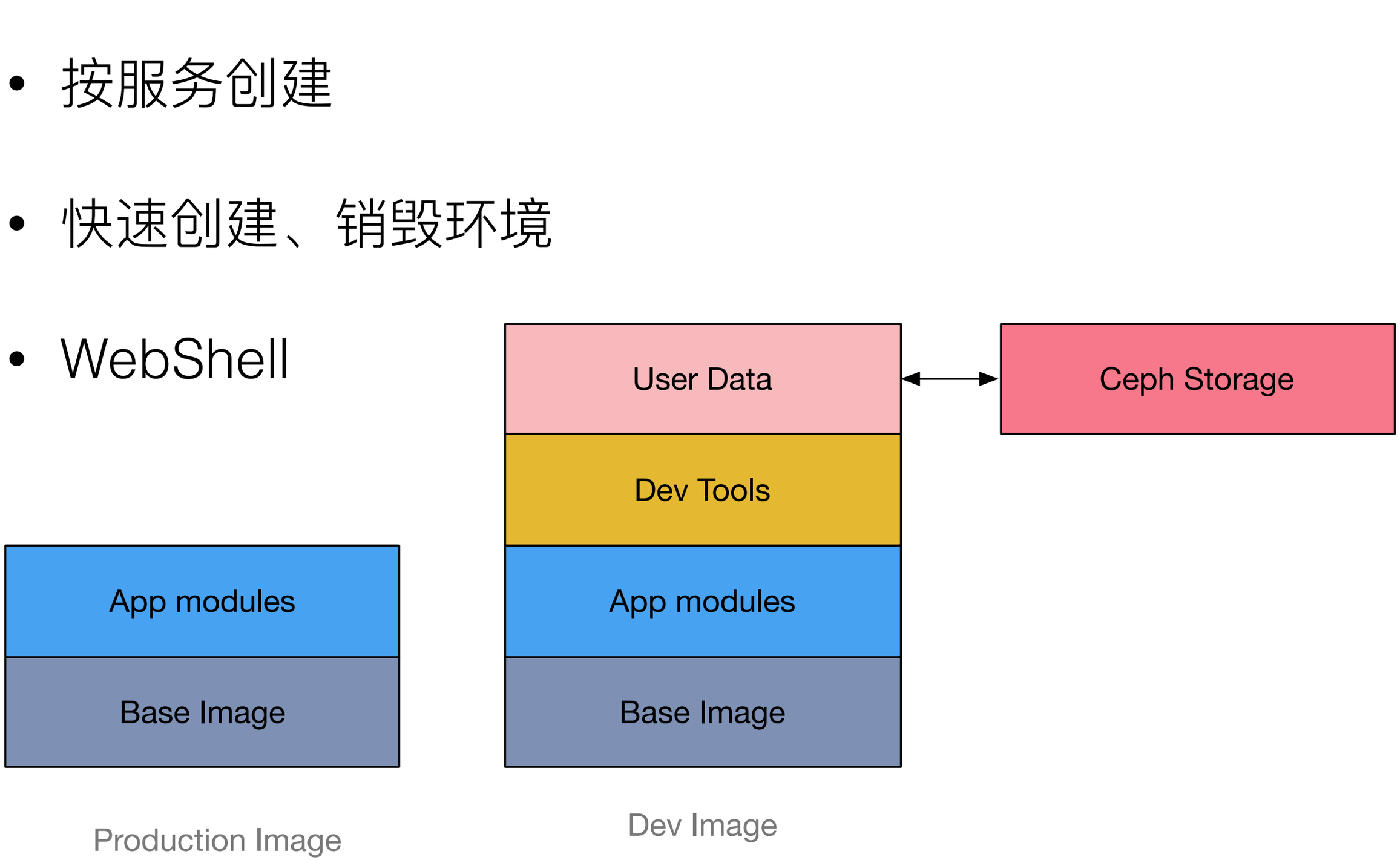
语言类型：python　　服务安装脚本：toutiao/app/stream/api

报警列表：

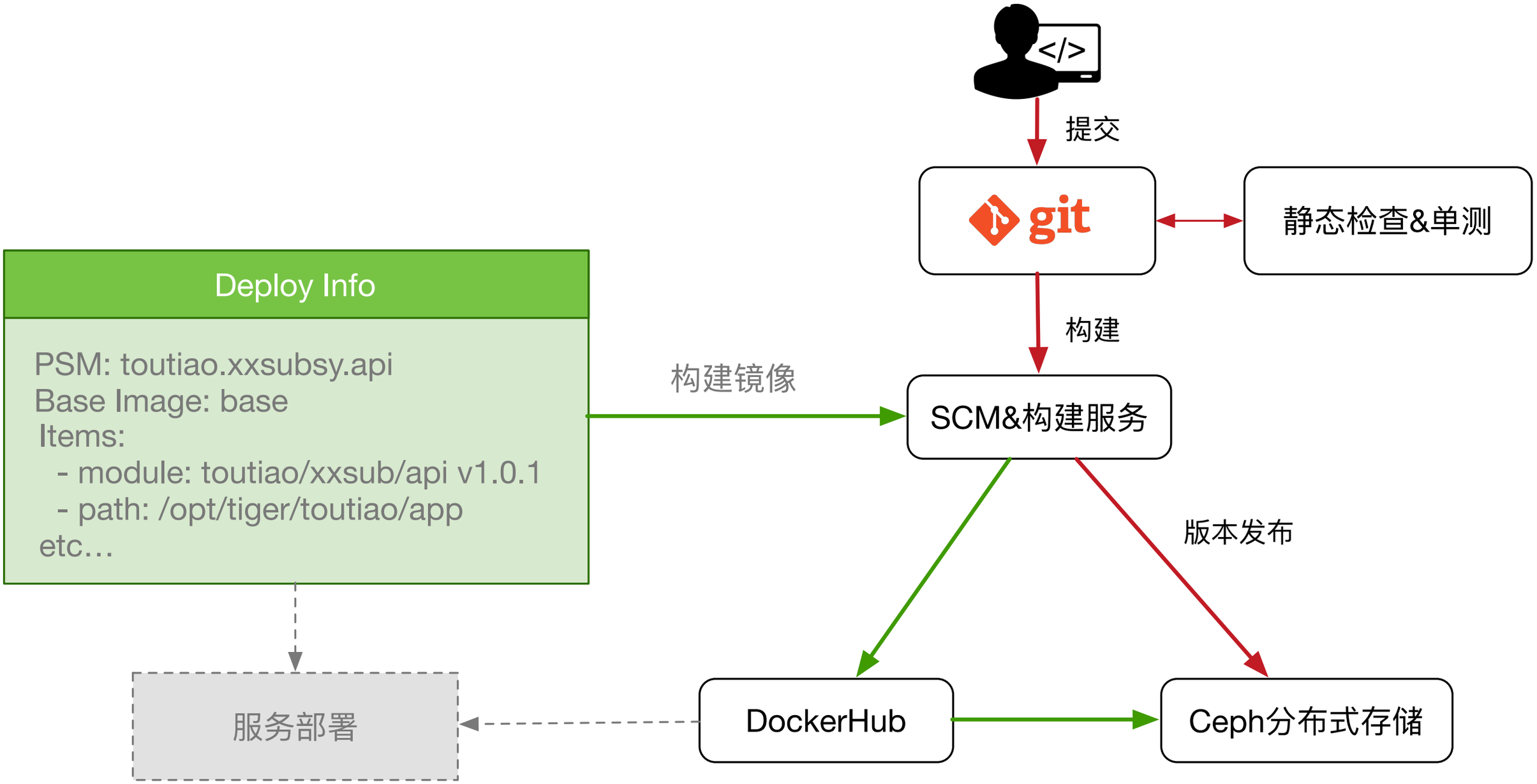
依赖仓库	部署路径	版本	更新
toutiao/storage/python/storage	toutiao/lib/storage	1.0.0.2	
toutiao/lib/toutiao	toutiao/lib/toutiao	1.0.0.10	
toutiao/frame	toutiao/lib/frame	1.0.0.15	
toutiao/conf	toutiao/conf	1.0.0.87	

开发镜像(WIP)

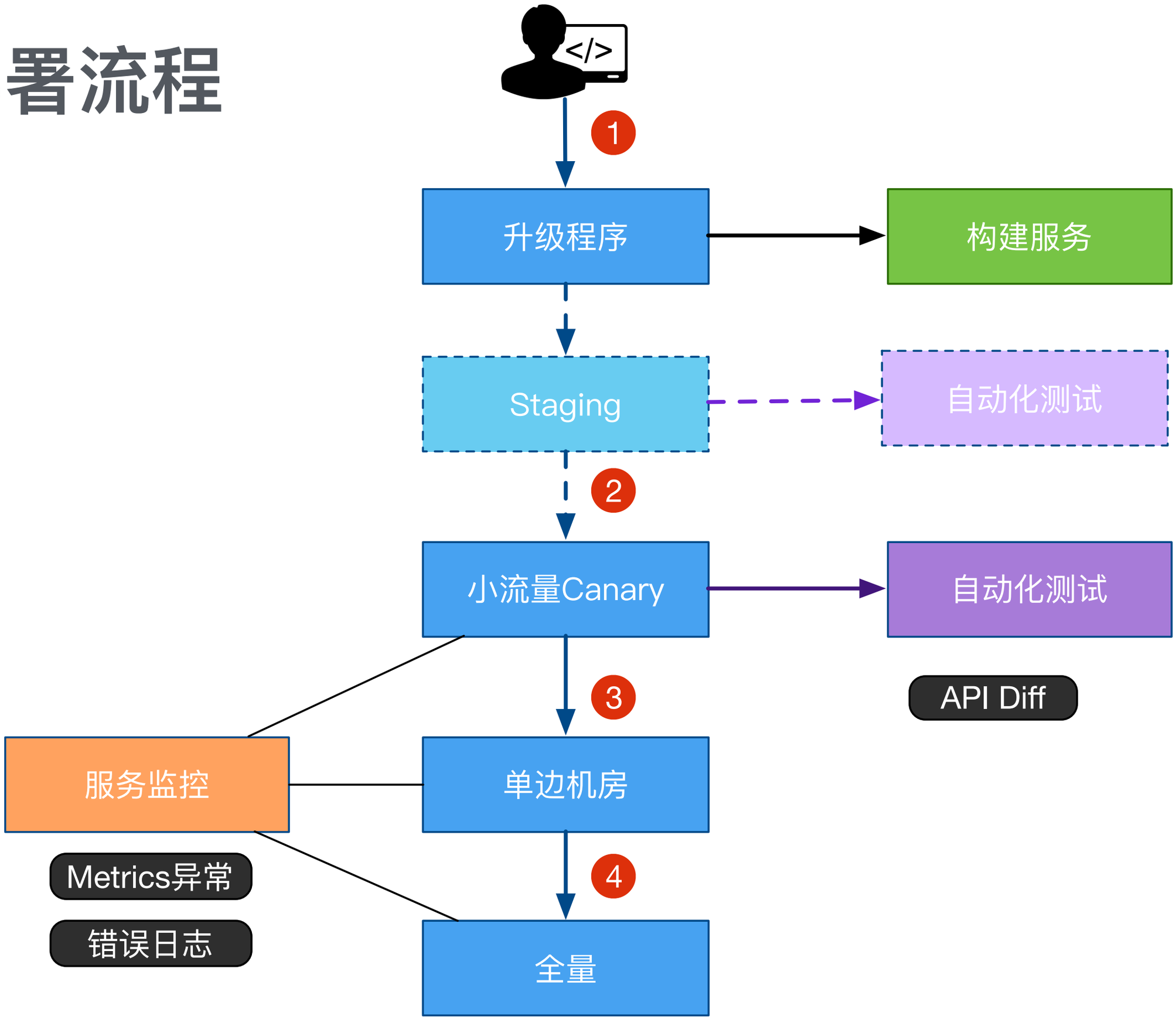
- 按服务创建
- 快速创建、销毁环境
- WebShell



构建流程


Deploy Info
PSM: toutiao.xxsubsy.api
Base Image: base
Items:
- module: toutiao/xxsub/api v1.0.1
- path: /opt/tiger/toutiao/app
etc…
</>
提交
git
静态检查&单测
构建
构建镜像
SCM&构建服务
版本发布
服务部署
DockerHub
Ceph分布式存储

部署流程


1
升级程序
构建服务
Staging
自动化测试
2
小流量Canary
自动化测试
3
API Diff
服务监控
单边机房
Metrics异常
4
错误日志
全量

03. 遇到的问题以及未来的规划

遇到的问题

- Kubernetes集群规模问题：千级别的节点
- IaaS层封装，底层分集群
- 多集群，快速弹性调度只能在集群内

遇到的问题

- 网络相关问题：
  - 端口分配问题：可能和临时端口冲突，修改range
- systemd
  - 守护的进程只能使用root账号，导致日志权限为root
  - 环境变量无法从docker继承的问题

遇到的问题

- 小容器资源变小
- Python类服务多进程模型服务启动CPU过高的问题
- 开发框架适配Docker环境，worker数的适配等

遇到的问题

- 数据库访问授权
    - 容器化后，所在的物理机IP会经常变动，连接信息隔离
    - 利用应用层MySQL新功能。或者在SQL层次带上更多的认证信息，改造成本大
    - 重要服务物理级别隔离

遇到的问题：历史包袱

- 脚本类服务的基础库统一更新问题
  - 统一更新，容易出事故
  - 不统一更新，基础库版本不一致
- 选择和微服务及容器化一致的理念：自包含
  - 版本发布进行标注，强制应用下次升级更新解决一致性问题

未来的规划

- IaaS层抽象和改造
  - 调度器的优化：不同类型的业务，提升利用率
  - 支持有状态服务
- 多地域的支持：国际化，周边设施的完善
- 混合云： 接入IaaS公有云提供更好的伸缩能力
- PaaS服务和SaaS服务的深度整合

Thanks