



GOPS 2020
Shenzhen

GOPS 全球运维大会

2020 - AIOps 风向标

深圳站

中国 · 深圳

指导单位:



主办单位:



GreatOPS Community



OOPSA Open OPS Alliance

基于TARS的云原生Devops技术架构演进

利开园 腾讯高级工程师

工作经历

腾讯Docker容器平台开发

TARS服务的基础运行环境和运营平台

腾讯TARSGo框架开发

TARS的GO语言实现

腾讯云开发业务后台开发

使用TARSGo，实现TARS+K8S的方案



CONTENTS

目录

- ① TARS的DevOps解决方案
- ② 云原生趋势与TARS架构演进
- ③ 基于TARS的应用案例

01

TARS的DevOps解决方案

DevOps的核心挑战

高质运维

支持海量微服务的维护，保障服务质量，减少运维工作和复杂度

节省成本

支持弹性调度，节省服务器、网络等基础设施成本



高效开发

快速实现产品能力，支持版本的高效升级

并发能力

支持海量用户并发访问，支撑复杂的业务逻辑

TARS微服务架构主要能力

高效率开发

IDL

代码自动生成

高性能RPC

API网关

多语言

CI/CD

名字服务

测试工具

高质量运维

服务可视化

无损变更

配置管理

日志

监控

调用链

Set管理

过载保护

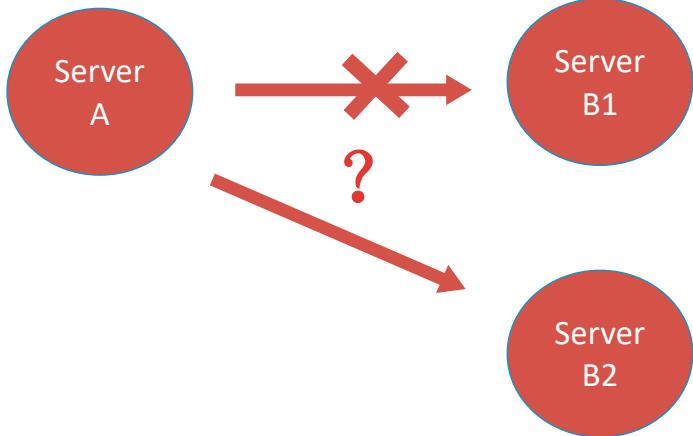
容灾容错

弹性伸缩

TARS与主流微服务方案比较

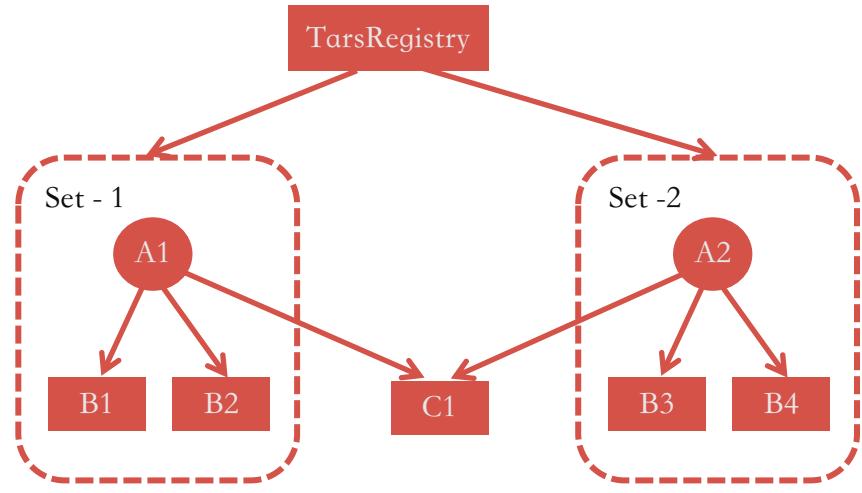
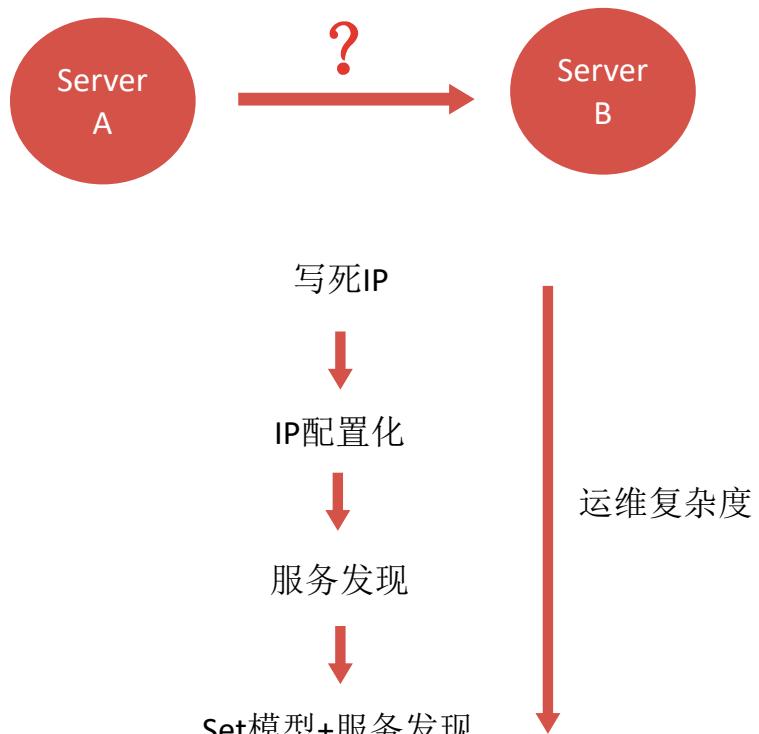
对比项	Tars	GRPC	Dubbo	备注
网络协议	udp/tcp/http2 /自定义	http2	tcp/http/自定 义	Tars默认提供udp/tcp支持, dubbo支持tcp/http等, grpc只支持http2
IDL	tars/pb/自定 义	pb	无	IDL本身思路差不多, tars可以设置字段默认值
多语言	支持	支持	不支持	Grpc/tars都支持多语言开发, dubbo最主要支持 java
RPC性能	非常高	一般	高	经过测试, tars性能高grpc 5倍, 且高于dubbo
服务治理	提供整套体系	无	提供服务治理	Tars与dubbo提供整个服务治理生态体系, grpc 更多的是提供思路, 由其它社区或者自己构建 生态。
应用规模	已大规模应用 11年	社区广泛应用	社区广泛应用	Tars已经大规模应用11年, grpc和dubbo社区用 得更多, 时间不长。

TARS负载均衡与容灾容错



- 负载均衡
 - 轮询
 - 取模HASH
 - 一致性HASH
- 熔断
 - 屏蔽: 超时/无法连接
 - 恢复: 尝试连接与请求
- 过载保护
 - 限制并发数

TARS名字服务与Set模型



```

func main(){
    comm := tars.NewCommunicator()
    client := &Test.Server{}
    comm.StringToProxy(servant: "Test.Server.MainObj" client)

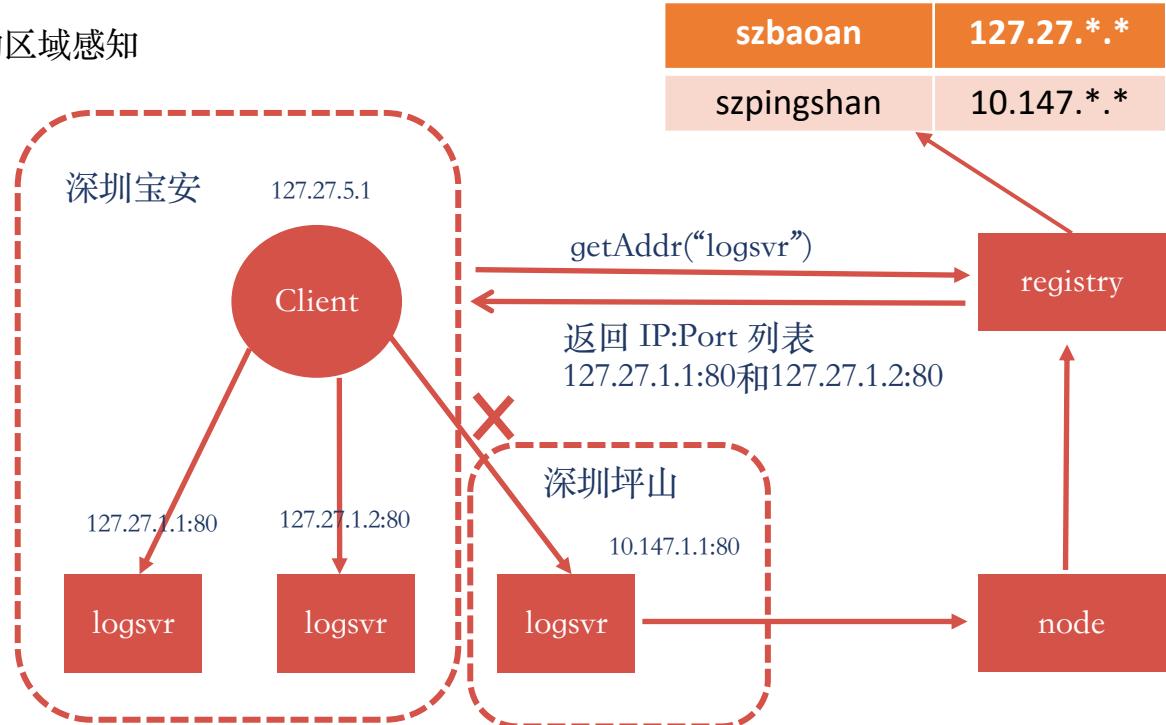
    ret, _ := client.Sum( A: 1, B: 1 )
    _ = ret
}
  
```

机房容灾方案-IDC分组

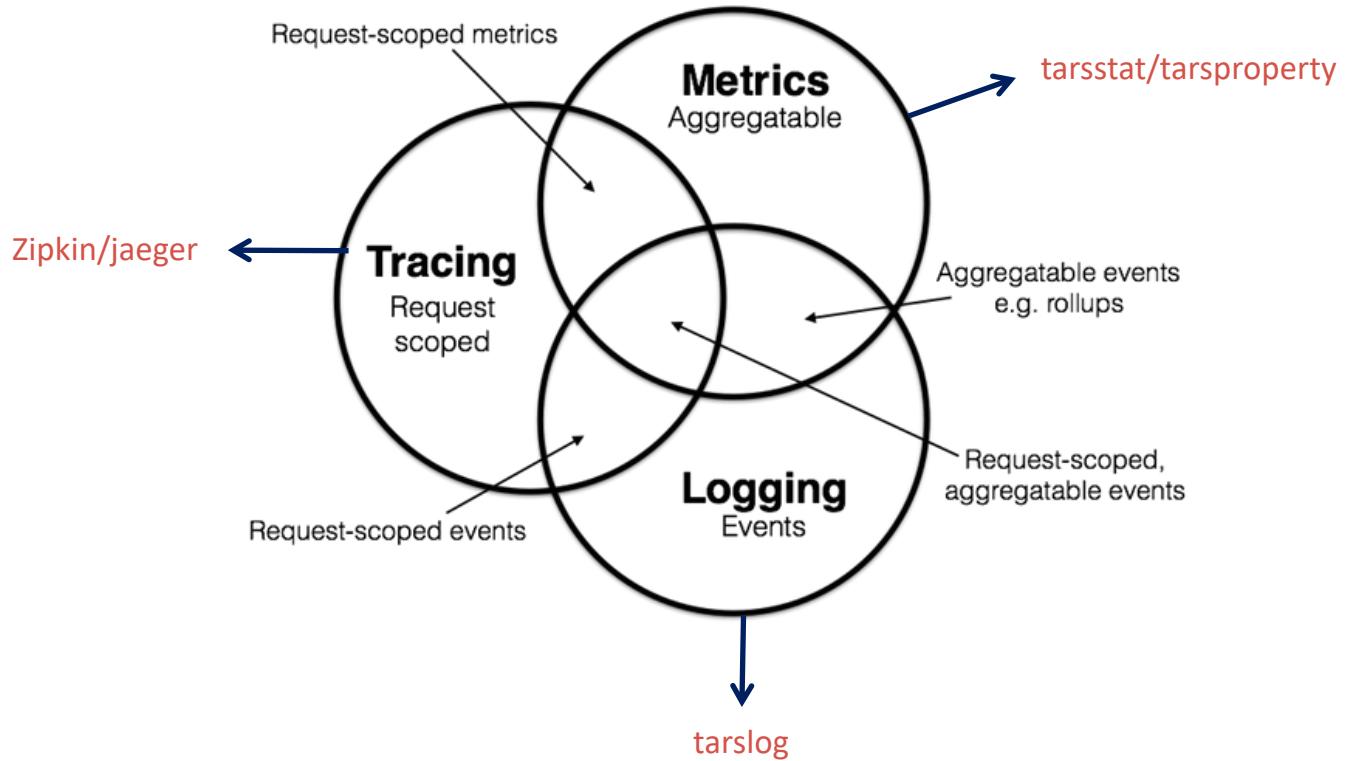
- 常规方案：网络延时高 OR 配置复杂
- TARS方案：名字服务实现自动区域感知

优势

- 运维简单
- 降低延时减少带宽消耗
- 更强的容灾能力



Logging/Metrics/Tracing



02

云原生趋势与TARS架构演进

云原生基础架构的演进趋势



TARS云原生方案的思考

TARS + K8S

完善服务治理能力

云原生服务部署能力

高效开发

低成本

高质运营

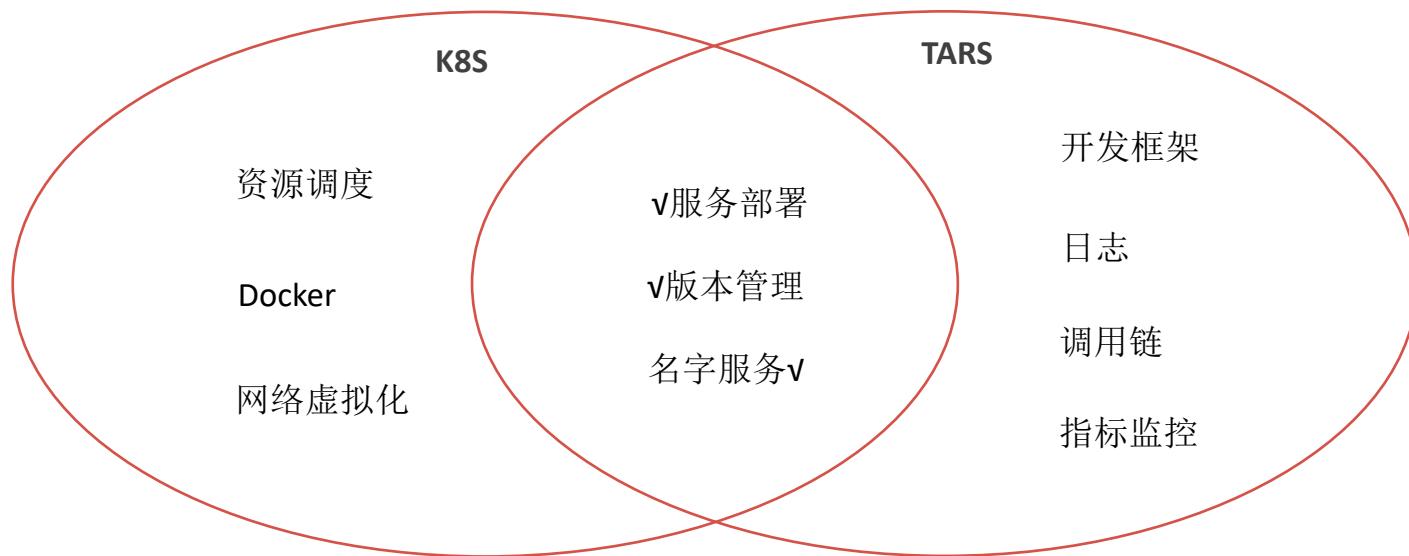
避免厂商绑定

没用使用Istio原因：

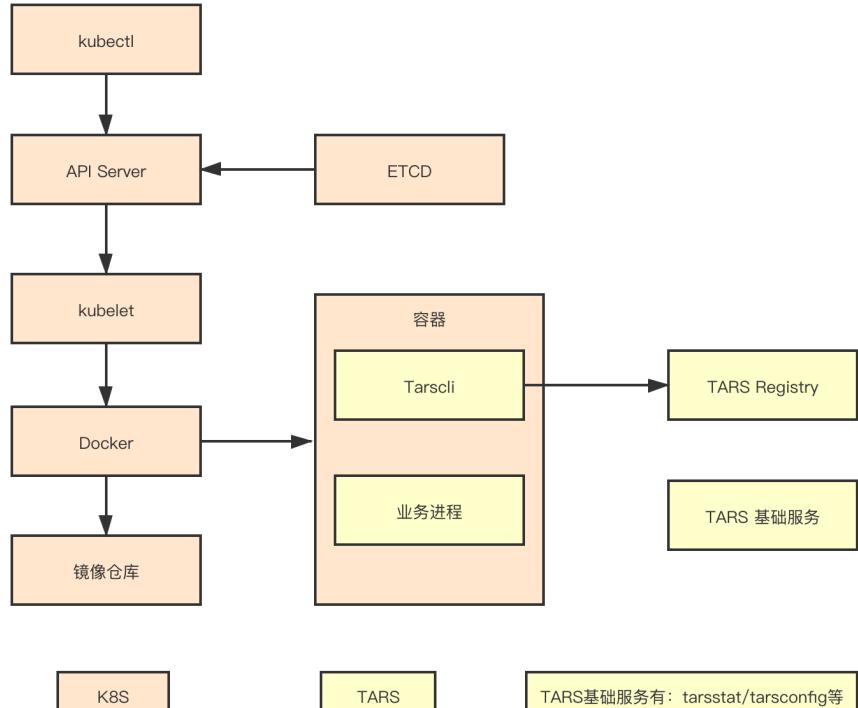
- 存量服务迁移成本
- 长连接（高性能）支持
- 客户端容错能力
- 支持set模型/IDC分组的名字服务
- 服务端口管理
- 版本稳定性

TARS+K8S整合方案

- K8S: 原生资源管理能力
- TARS: 保留开发架构及服务治理能力



TARS+K8S方案及名字服务整合



Pod生命周期

Tarscli: 保证TARS名字服务与容器列表的一致性

启动

Tarscli supervisor 启动TARS服务

就绪

Tarscli supervisor 将TARS服务设为active

运行

Tarscli supervisor 监控TARS服务和上报心跳

删除前

Tarscli prestop 下线TARS服务

删除

Tars Registry通过心跳保证最终一致

方案架构图

TARS服务部署示例及可视化管理

基础服务

```
kubectl apply -f baseserver/yaml/db_all_in_one.yaml
```

```
kubectl apply -f baseserver/yaml/registry.yaml
```

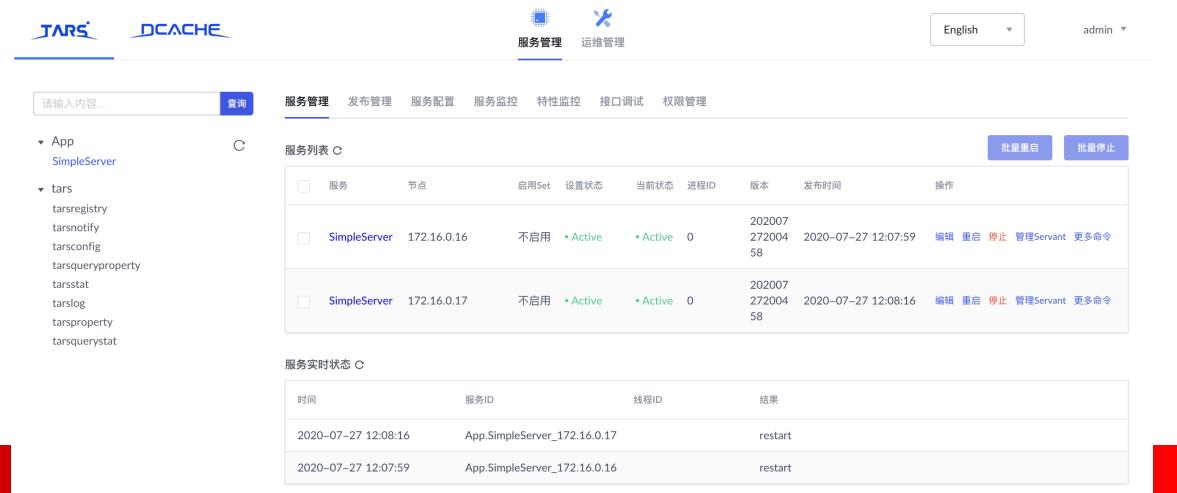
```
kubectl apply -f baseserver/yaml/tarsweb.yaml
```

```
kubectl apply -f baseserver/yaml/tarsnotify.yaml
```

业务服务示例

```
kubectl apply -f examples/simple/simpleserver.yaml
```

```
kubectl apply -f examples/TestApp>HelloGo/simpleserver.yaml
```



The screenshot shows the TARS Service Management Platform interface. At the top, there are tabs for 'TARS' and 'DCACHE'. Below the tabs, there are two main sections: '服务管理' (Service Management) and '运维管理' (Operations and Maintenance). The '服务管理' section contains a search bar, a sidebar with service categories like 'App' and 'tars', and a table for managing services. The table lists 'SimpleServer' instances with details such as IP (172.16.0.16, 172.16.0.17), status (不启用, Active), and actions (Edit, Restart, Stop, Manage Servant, More Commands). The '运维管理' section shows a table of real-time service status logs.

服务	节点	启用Set	设置状态	当前状态	进程ID	版本	发布时间	操作
SimpleServer	172.16.0.16	不启用	Active	Active	0	20200758	2020-07-27 12:07:59	编辑 重启 停止 管理Servant 更多命令
SimpleServer	172.16.0.17	不启用	Active	Active	0	20200758	2020-07-27 12:08:16	编辑 重启 停止 管理Servant 更多命令

时间	服务ID	线程ID	结果
2020-07-27 12:08:16	App.SimpleServer_172.16.0.17		restart
2020-07-27 12:07:59	App.SimpleServer_172.16.0.16		restart

监控页面（变更过程无损）

TARS DCACHE 服务管理 运维管理 English admin

请输入内容...

App SimpleServer

显示日期 对比日期 开始时间 结束时间

20200727 20200727 0000 2360

接口名 主调 被调 主调IP

tarsregistry tarsnotify tarsconfig tarsqueryproperty tarsstat tarslog tarsproperty tarsquerystat

被调IP

镜像变更

总流量 -○ 当日 -○ 对比日

0830
● 当日: 212
● 对比日: 212

平均耗时 -○ 当日 -○ 对比日

异常率 -○ 当日 -○ 对比日

超时率 -○ 当日 -○ 对比日



TARS未来规划

- 日志、监控、调用链与开源方案的整合
 - ES/Grafana/Jaeger
- TARS+K8S的深度整合
 - TarsWeb对接K8S，屏蔽k8s底层实现
- TARS + Service Mesh的探索
 - 将TARS核心能力整合到Sidecar中

03

基于TARS的应用实践

CI/CD应用实践

- CI/CD流程

1. 提MR请求
2. 单元测试
3. 人工Review
4. 合并代码
5. 发布正式环境



- 演进方案

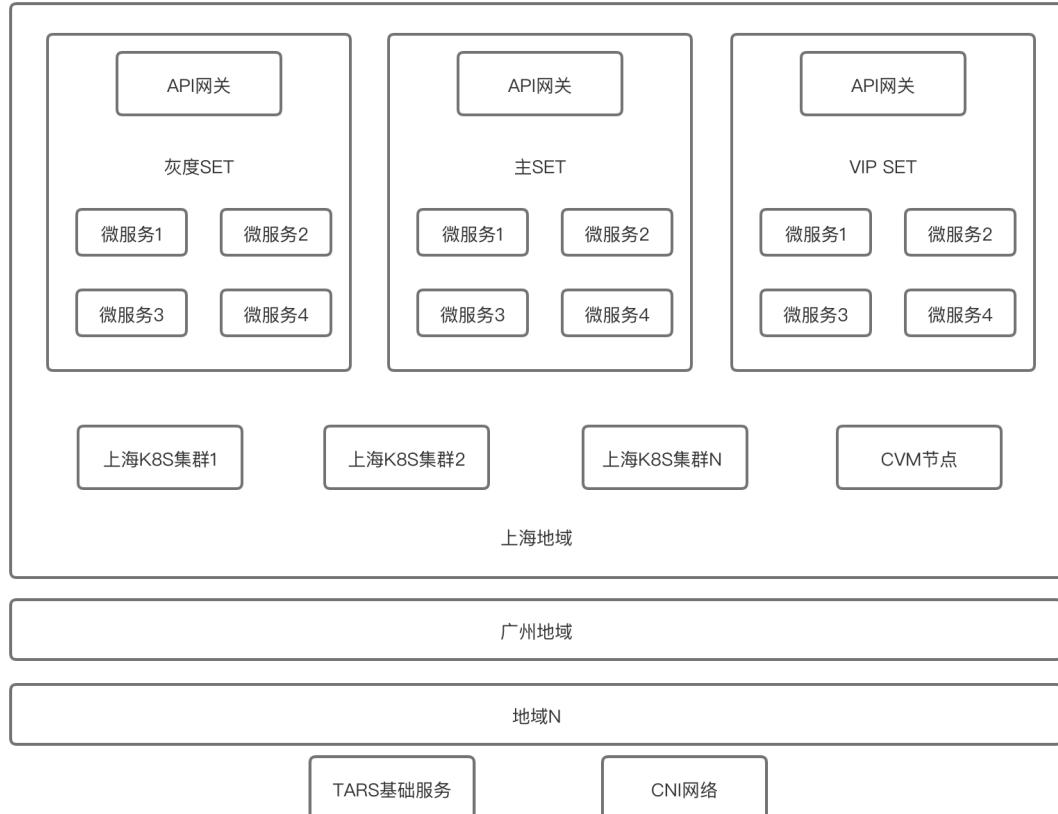
1. 企业微信机器人发布
2. 自动提MR
3. 单元测试
4. 模拟合并
5. 自动发布体验、预发环境
6. 自动化测试
7. 人工Review
8. 合并代码—自动生成版本
9. 发布正式环境

TCB持续集成 BOT

```
=====
任务564 CI结果通知
=====
FAIL: 有单元测试未通过
单元测试总数:138 成功:118 通过率:85.5%, 阈值:80% 结果:通过
当前构建分支|feature/dianshi_for_wx_master|,对比分支|master|,增量代码覆盖率:0.0%,阈值:0% 结果:通过

prerelease环境自动化测试任务ID=106205
用例总数:1545 成功:1545 失败:0 通过率:100% 阈值:100% 结果:通过
报告链接:http://qta.ao.com/#/bbk/autotest/report/17361638
✓ QCI流水线任务成功
请继续完成后续检查工作
1 观察各服务日志是否正常
```

基于TARS + K8S多集群混合部署架构



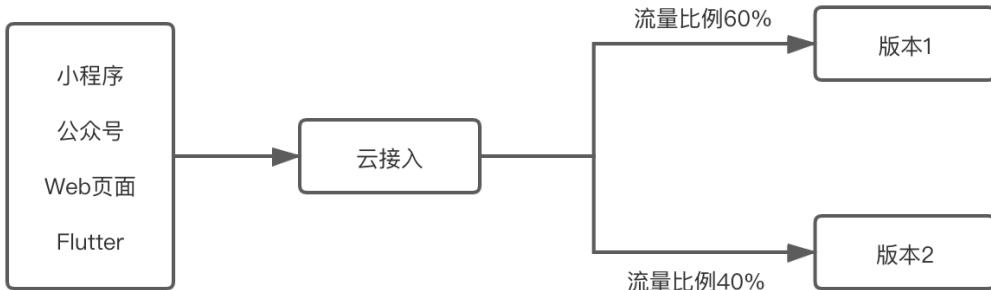
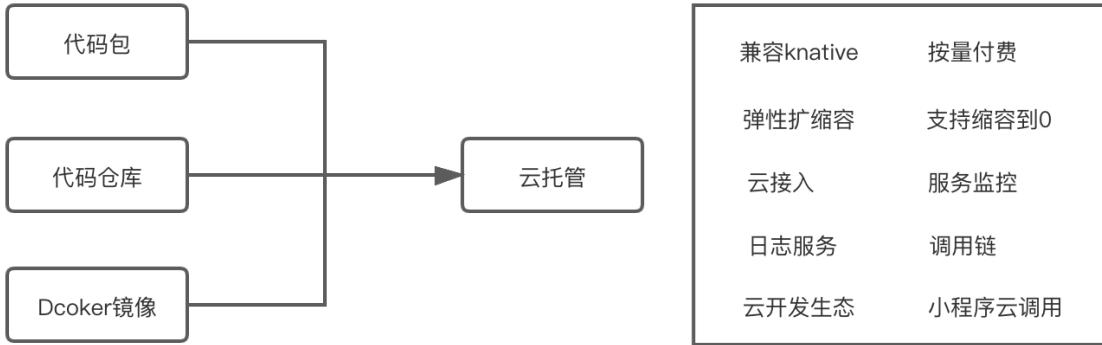
- 多Set功能
 - 隔离故障域
 - 容灾备份切换
- 部署架构
 - 多K8S集群容量水平扩容
 - 容灾
 - 支持CVM手工部署
- TARS基础服务
 - 复用同一套TARS系统
 - 统一的管理视图
- CNI网络模式
 - 实现多地域网络互通

TARS丰富的应用案例



- Tars在腾讯内部使用超过十年，并于2017年4月10日开源，开源后与业界众多企业交流，同时也得到了广泛应用。
- 开源地址：<https://github.com/TarsCloud>
- 2018年6月加入Linux基金会

云托管：新一代云原生应用引擎



<https://cloud.tencent.com/product/tcbr>



Thanks

高效运维社区
开放运维联盟

荣誉出品

