SACC 2014中国系统架构师大会
SYSTEM ARCHITECT CONFERENCE CHINA 2014
发现架构之美

高性能实时应用架构

网易 谢骋超
xiechengchao@corp.netease.com
@圈圈套圈圈

Who am I

- 网易资深架构师
  - 2006年加入网易

消息推送平台

目录

- 挑战与难点
- 架构篇
- 性能篇
- 总结

实时应用简介

- 易信 -- java
  - 网易的流行移动聊天工具
  - 千万级别在线

- 消息推送平台 -- node.js + java
  - 千万级别在线

    新闻客户端 有道云笔记 云音乐 云阅读 ...
  - 移动消息推送的平台

消息推送平台

- 推送消息到移动端、web端的通用平台，接入了网易几乎所有主要的移动产品

挑战与难点—消息推送平台

- 平台需高可用，并支持水平扩展
- 支持千万级高并发下的实时推送
- 网络不稳定情况下的消息可靠性（QoS1）
- 系统需具备高度稳定性和可靠性
- 针对不同消息推送需求制定不同解决方案
- 移动终端的4S要求：Slim、Save power、Save traffic、Stable

消息推送平台--挑战与难点

- 网络不稳定状况下消息仍可达，私信到达率99.9%以上
- 99.9%的消息1S以内到达
- Android SDK月流量消耗<500K，低于典型同类APP
- 电量消耗低于典型同类APP
- 私信TPS>1000，全域广播TPS>10w

挑战与难点—易信

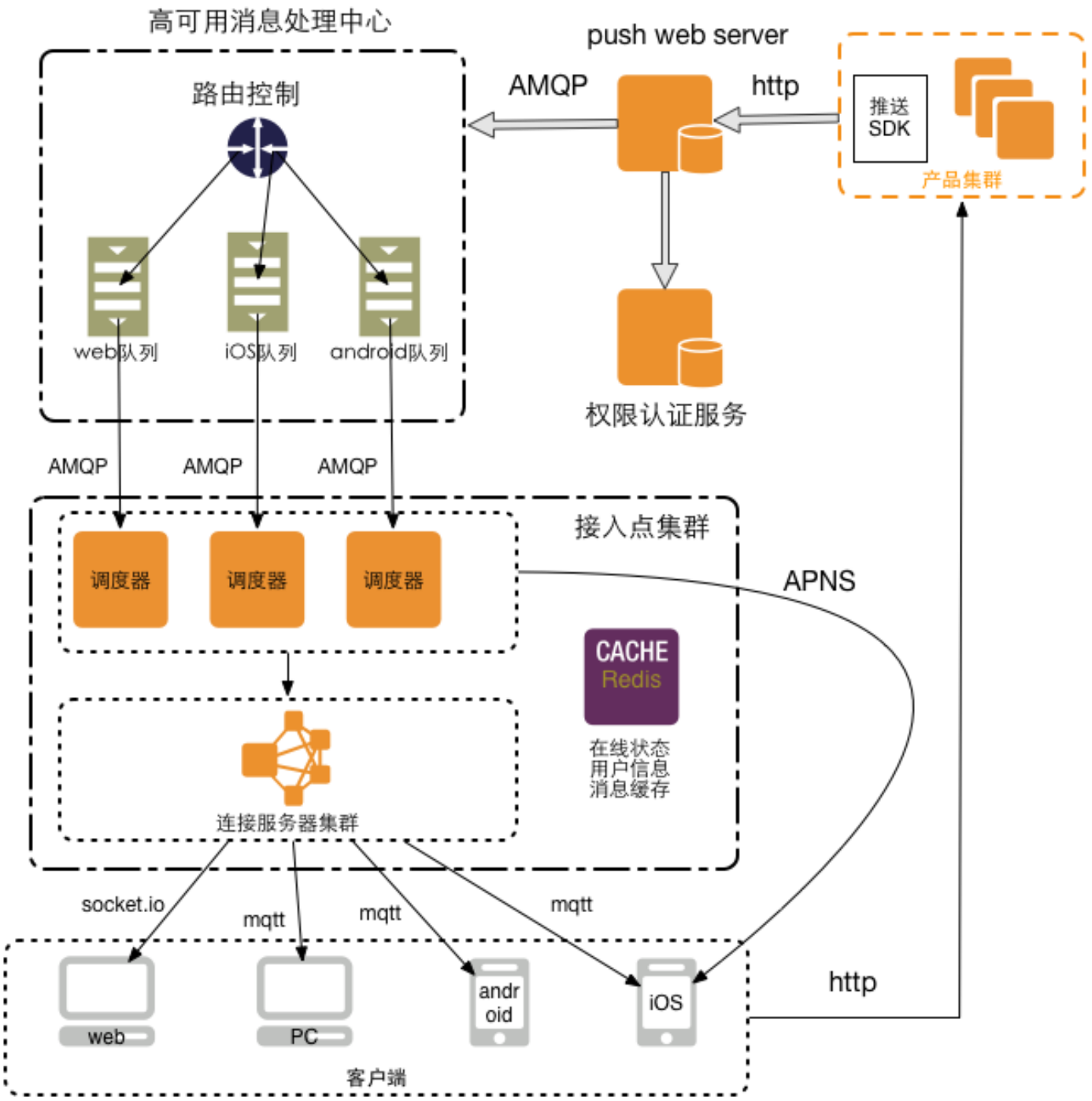
- 支持上亿的同时在线规模
- 消息发送的密度与频率高于消息推送平台，私信的TPS达10万级，群聊的TPS达1万级
- 网络不稳定状况下消息仍可达，私信到达率99.99%以上
- 99.99%的消息1S以内到达
- 高质量语音通讯

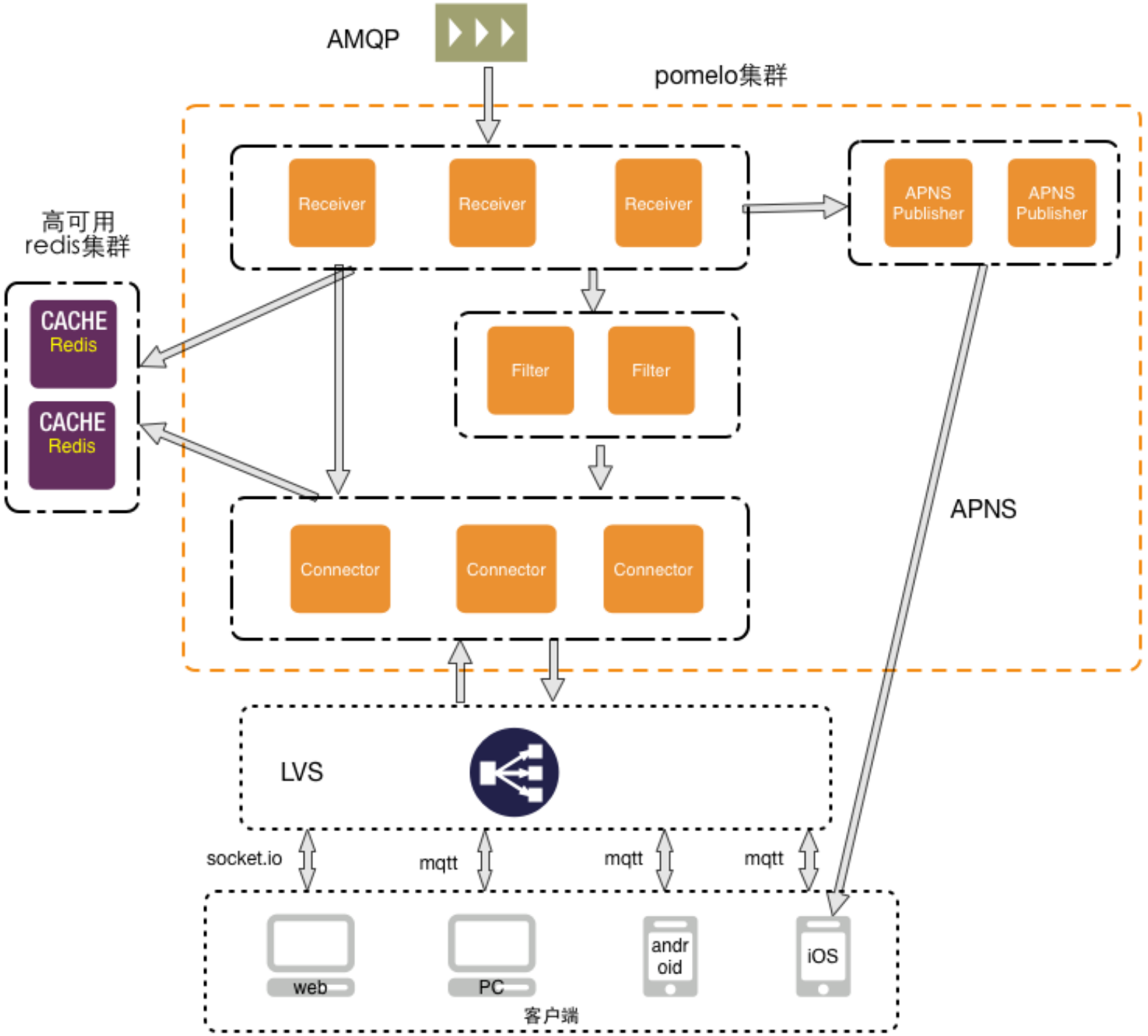
目录

- 挑战与难点
- 架构篇
- 性能篇
- 总结

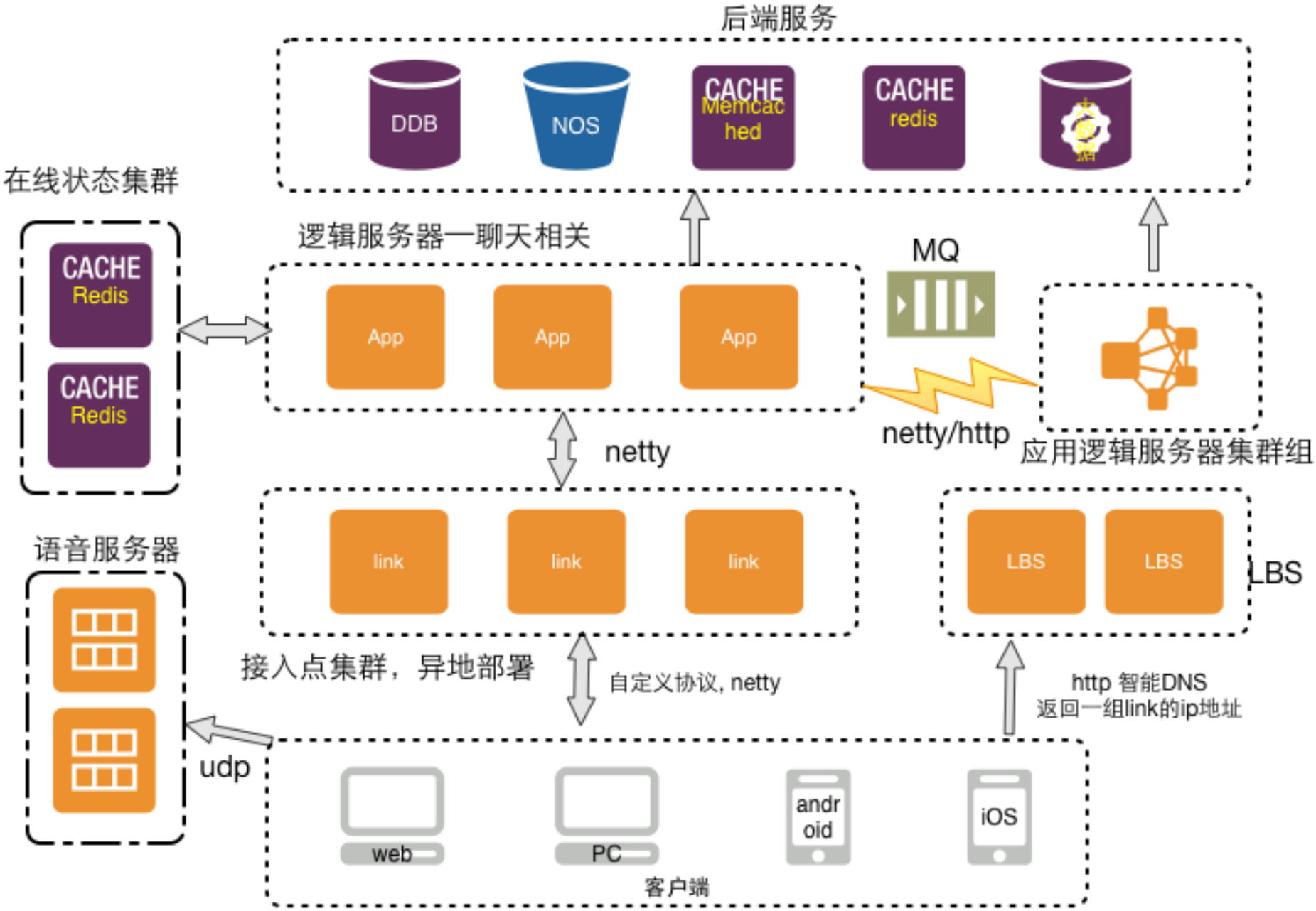
架构讨论

- 基础架构
- 消息实时性
- 消息到达率
- 高可用


高可用消息处理中心
push web server
路由控制
AMQP
http
推送
SDK
产品集群
web队列
iOS队列
android队列
权限认证服务
AMQP
AMQP
AMQP
接入点集群
调度器
调度器
调度器
APNS
CACHE
Redis
在线状态
用户信息
消息缓存
连接服务器集群
socket.io
mqtt
mqtt
mqtt
http
web
PC
andr
oid
iOS
客户端


AMQP
pomelo集群
Receiver
Receiver
Receiver
APNS Publisher
APNS Publisher
高可用
redis集群
CACHE
Redis
CACHE
Redis
Filter
Filter
APNS
Connector
Connector
Connector
LVS
socket.io
mqtt
mqtt
mqtt
web
PC
andr oid
iOS
客户端

易信—聊天部分


后端服务
DDB
NOS
CACHE
Memcac
hed
CACHE
redis
在线状态集群
CACHE
Redis
CACHE
Redis
逻辑服务器—聊天相关
App
App
App
MQ
netty/http
应用逻辑服务器集群组
netty
语音服务器
link
link
link
LBS
LBS
LBS
接入点集群，异地部署
自定义协议, netty
http 智能DNS
返回一组link的ip地址
udp
web
PC
andr
oid
iOS
客户端

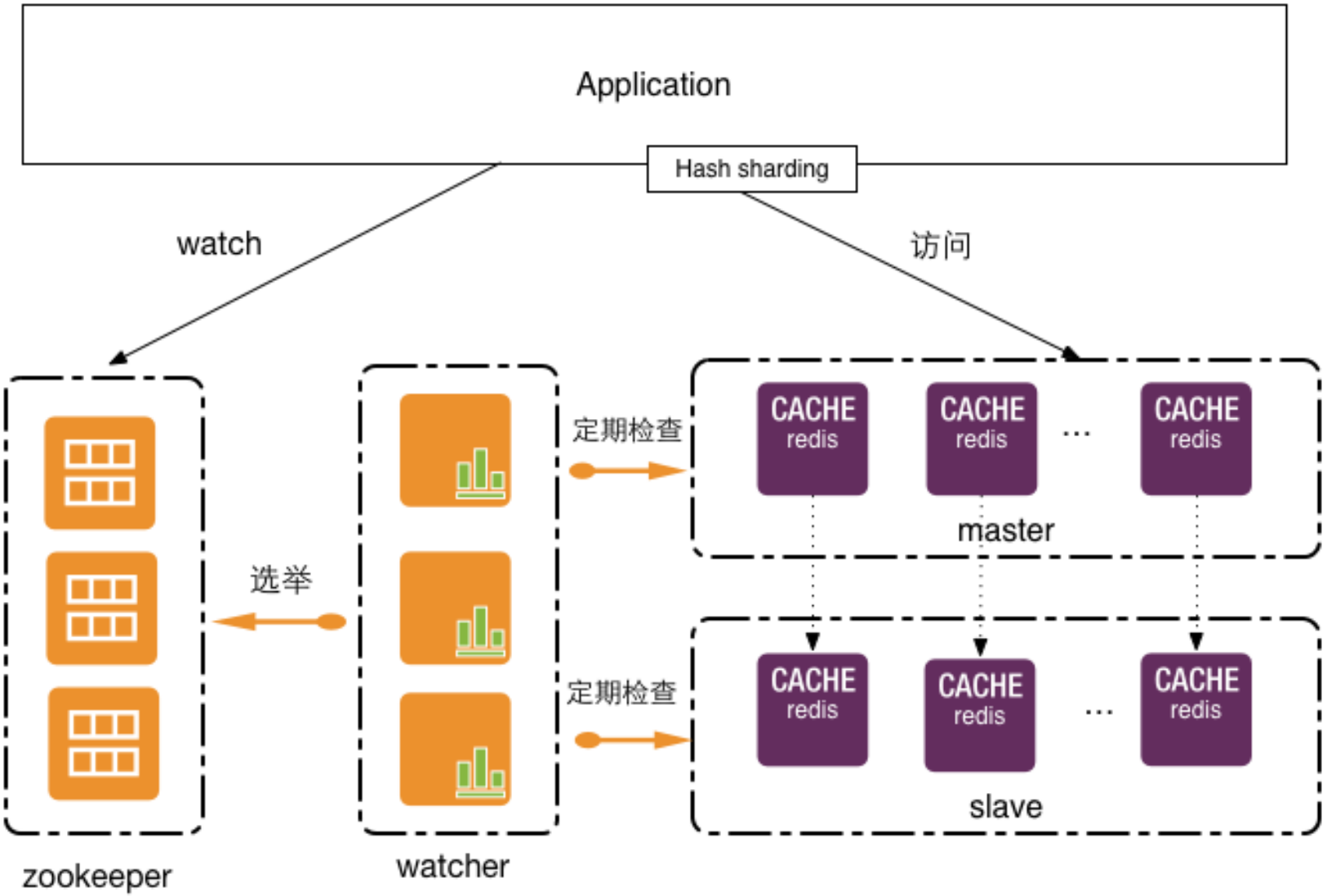
高可用

- 消息推送经过的所有功能服务进程无单点故障
- 高可用消息队列服务确保消息可靠性
- 多种流控和退避机制确保系统可靠性

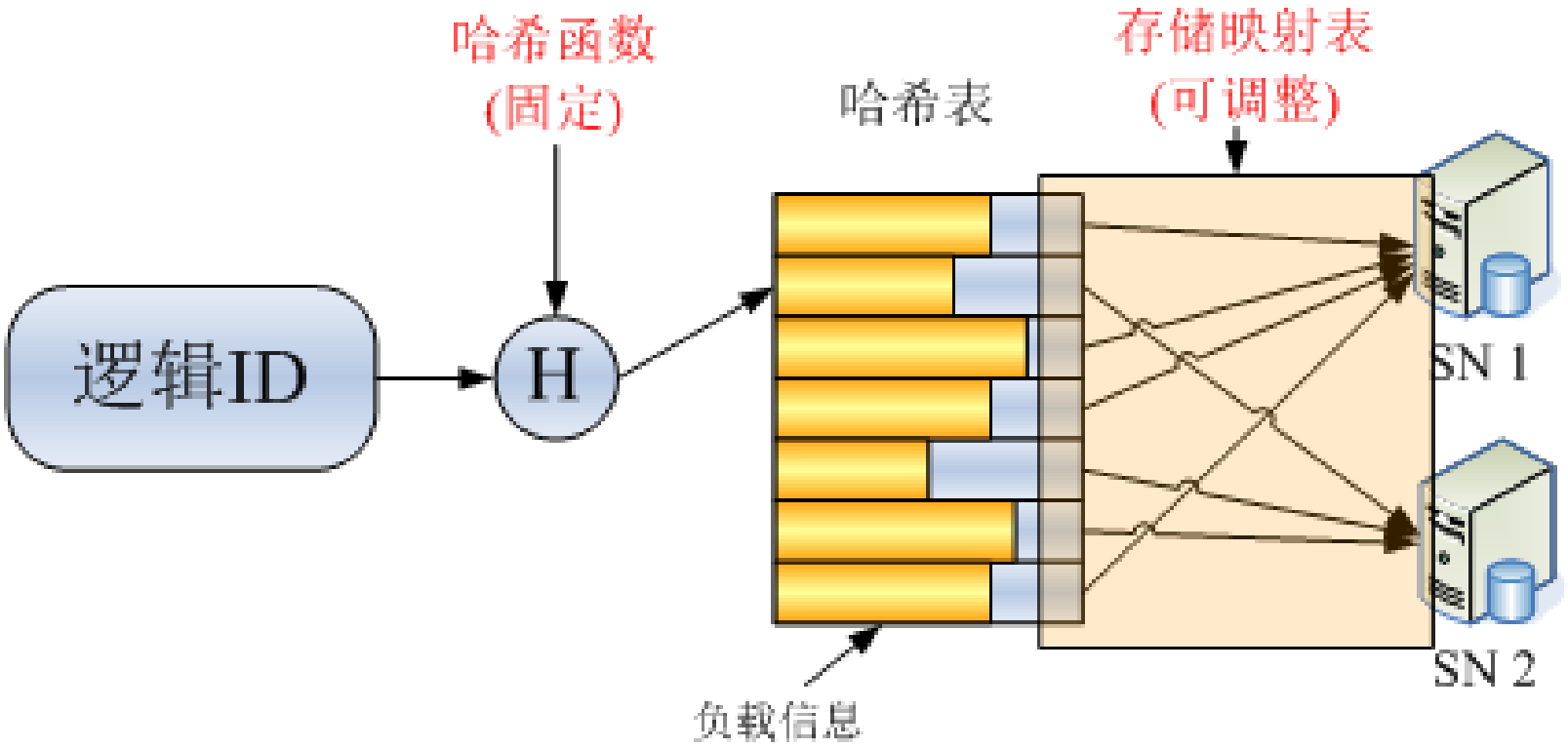
- 无状态 VS 有状态
  - 在线状态
  - 连接服务器 - 维护长连接

- Redis高可用集群方案 - zookeeper + presharding

高可用redis集群


Application
Hash sharding
watch
访问
选举
定期检查
定期检查
CACHE
redis
CACHE
redis
CACHE
redis
master
CACHE
redis
CACHE
redis
CACHE
redis
slave
zookeeper
watcher

高可用redis集群—presharding + 二级Hash



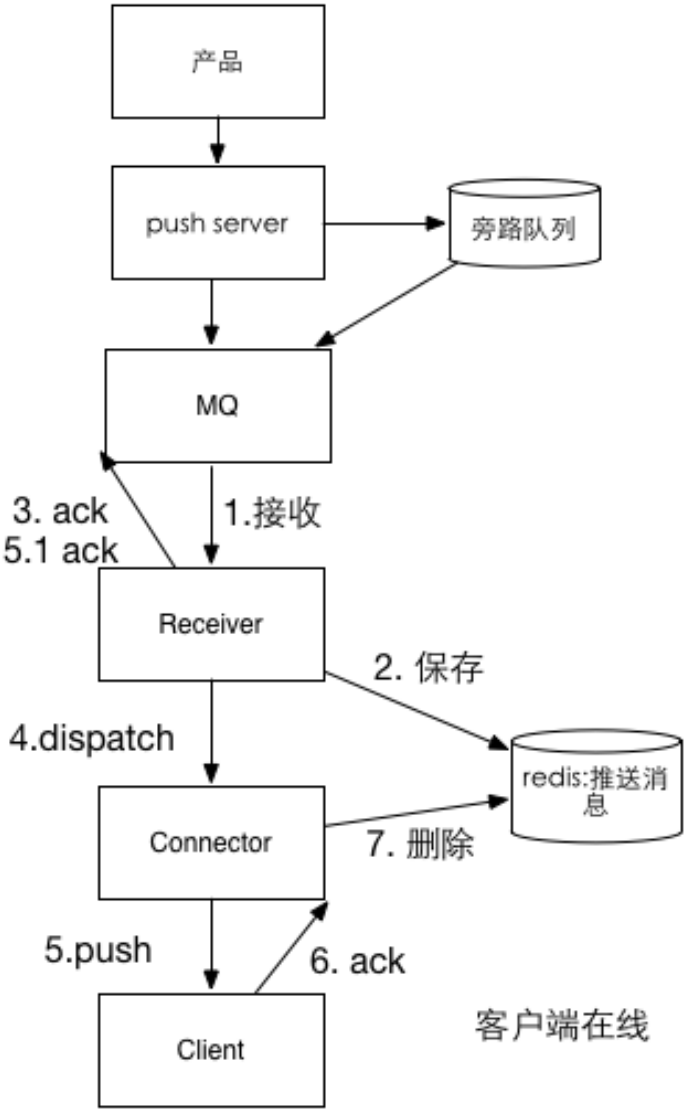
负载均衡

- 消息推送平台的负载均衡策略 – LVS
  - DR，RR模式
  - socket.io采用source hash
- 易信的负载均衡策略 – LBS
  - 使用智能DNS
  - 支持异地部署
  - 一组IP，多个端口重连

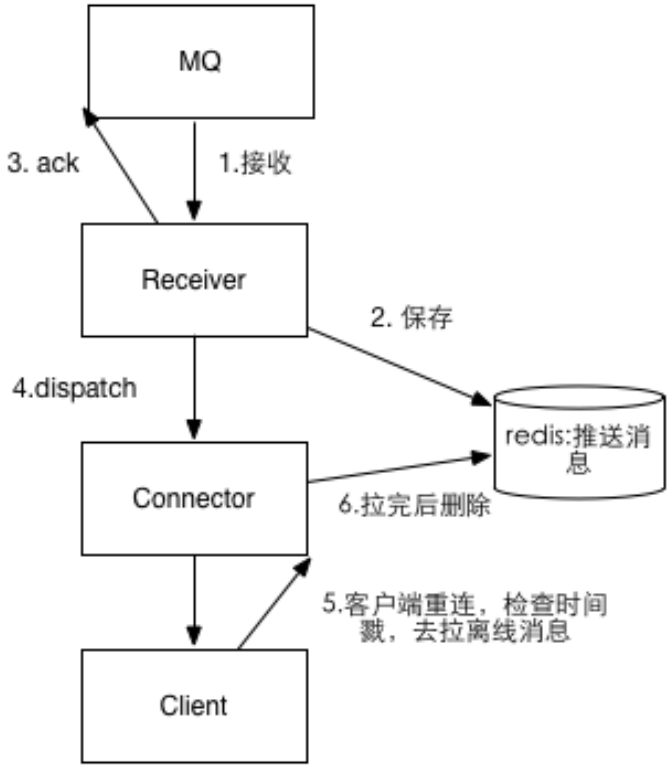
保证消息到达率

- 高可用队列qos=1
- 消息备份
  - Redis高可用，slave存盘
- 离线消息
  - Redis备份消息，Ack后才删除
  - 客户端保存接收最后一条消息的时间戳
  - 客户端重连拉未消费消息

消息到达流程



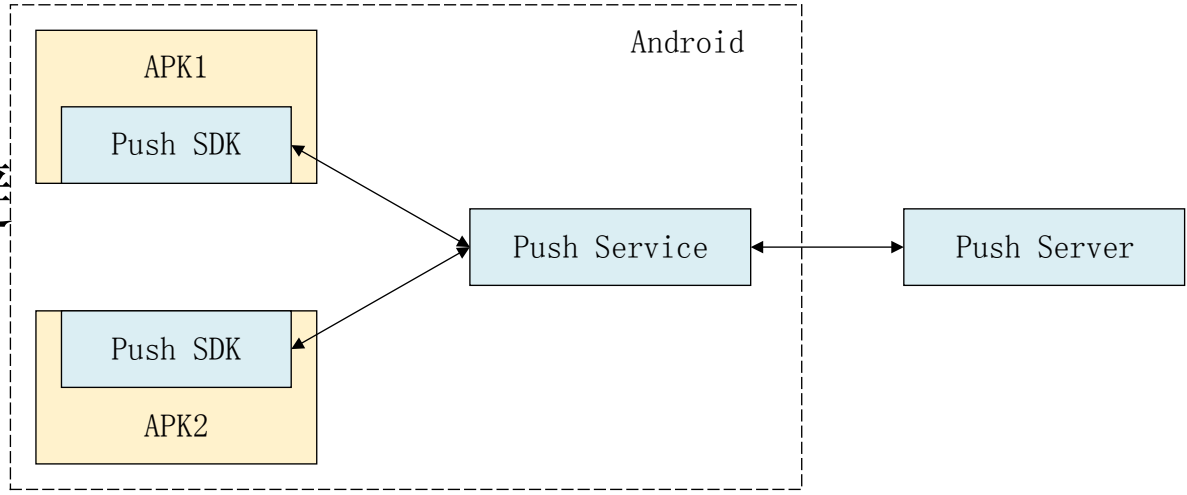
客户端在线



客户端不在线，拉离线消息

移动端 -- android客户端

- 链路复用
- 基于MQTT协议
- 心跳周期退避
- 服务端拥塞退避
- 断线重连机制
- 签名认证机制
- 支持消息去重
- 支持自动/手动Ack
- SDK升级策略（兼容+并存）



目录

- 挑战与难点
- 架构篇
- 性能篇
- 总结

性能

- 性能指标
- 消息实时性
- 广播的性能优化
- GC -- node.js 与 java

性能

- 支持的连接数
  - 易信（netty），25W连接，12G内存(前提： 不出现频繁Full GC），内存占用约40K/连接
  - web易信(netty)， 如果使用long polling，1.5W连接，CPU 90%
  - 消息推送平台（node.js mqtt协议），3W连接，600M内存，内存占用约 18K/连接
  - 消息推送平台web端（node.js socket.io）， 如果是long polling, 只能支撑3000个连接

性能

- QPS
  - 易信
    - 单进程6.8w并发用户时，发送点对点消息，14000/S
    - 响应时间 4ms以下
    - CPU 70% 内存8G
  - 消息推送平台
    - 单进程2w并发用户时，点对点消息， 3000/s
    - 数千万条广播，5秒内发完

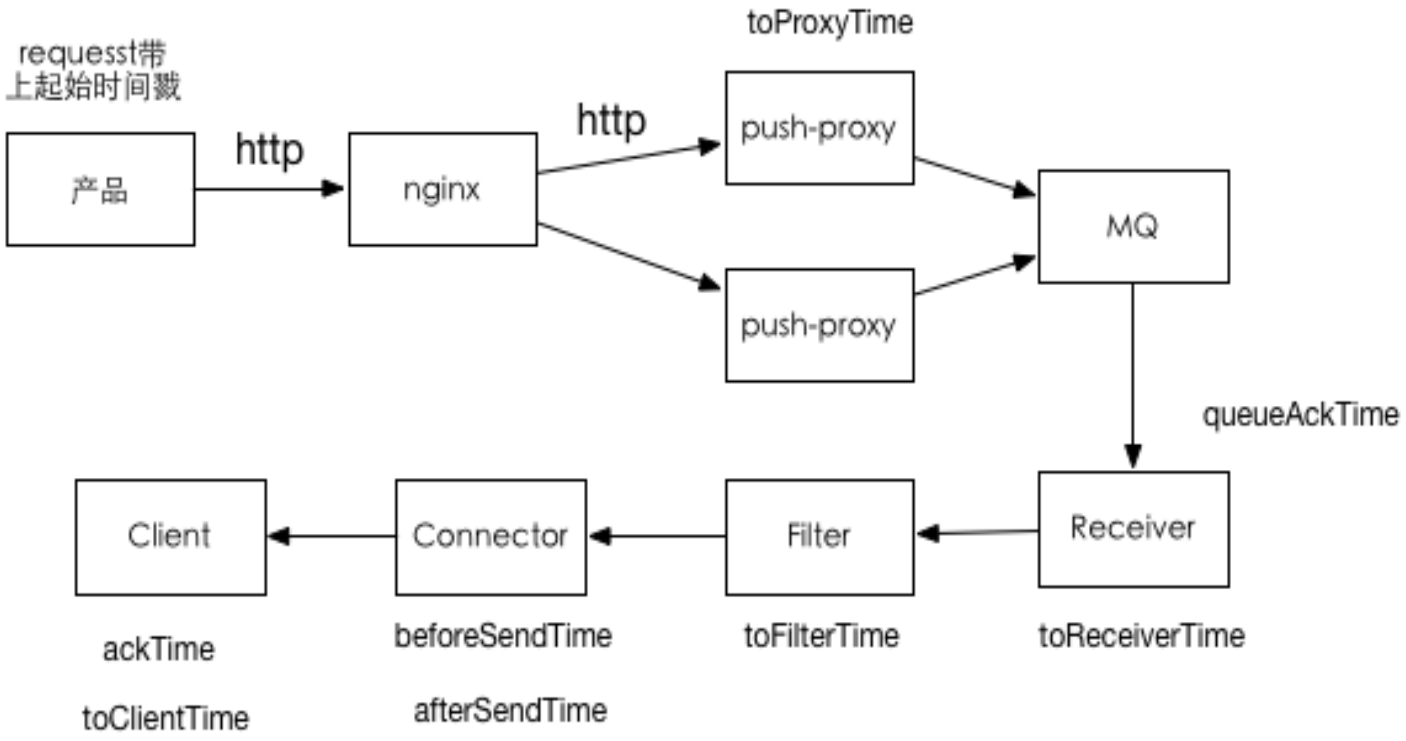
消息的实时性

实时性要求高
需要不停的测试与优化

- 消息路径划分与报警
- 一次故障排查
- GC

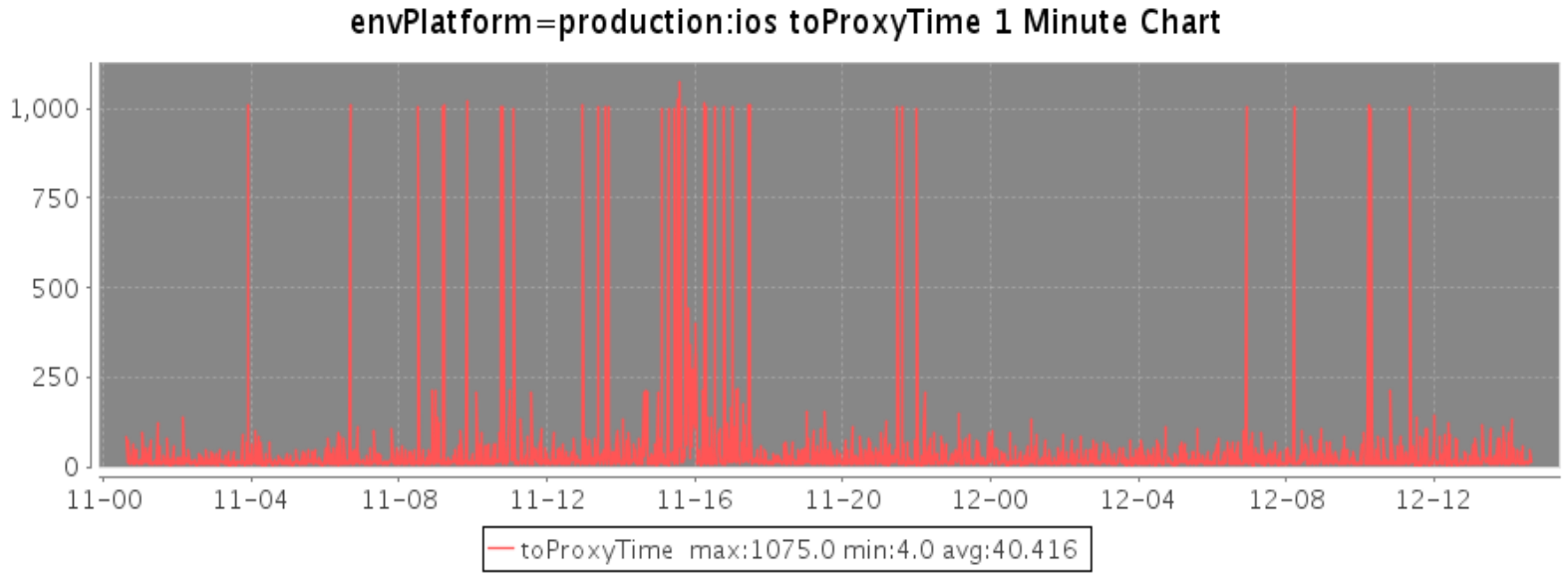
消息实时性 -- 消息路径划分

- 实时性要求
  - 按消息路径划分处理时间， 对各级响应时间打点



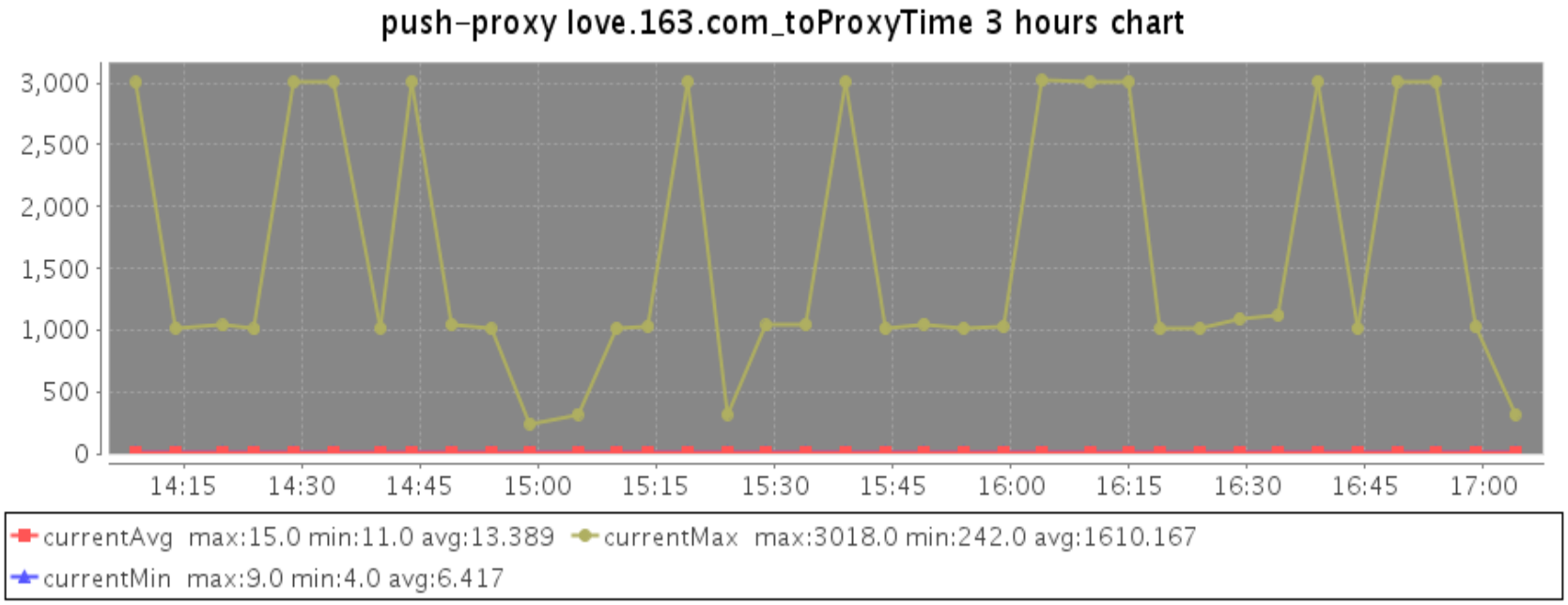
消息实时性 -- 一次故障排查实例

- 现象：产品反映聊天时响应较慢
- toProxyTime经常飙升到1秒



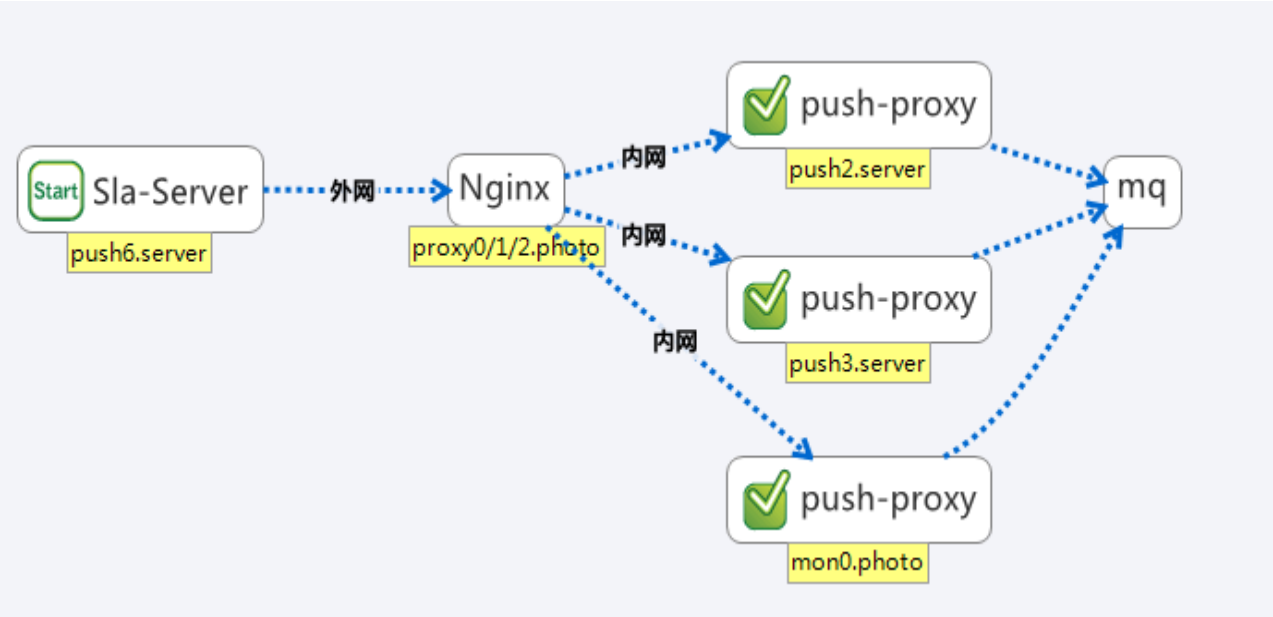
另一个产品

- toProxyTime升到3秒



消息实时性消息路径

- toProxyTime = endTime – startTime
- 排除法定位出nginx问题



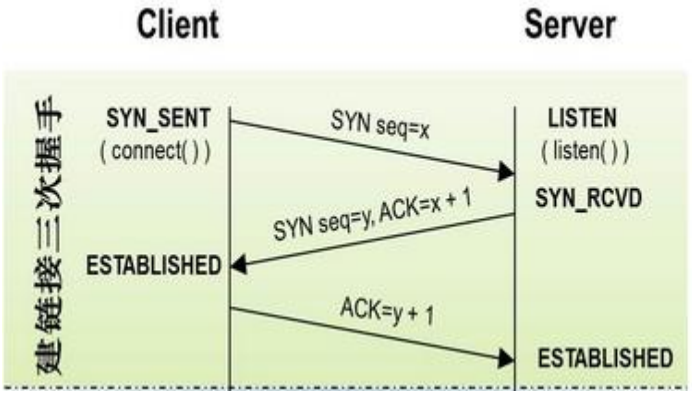
抓包

Filter: http.set_cookie contains "msgId" Expression... Clear Apply Save msgId msgOne msgTwo retransmissi

No.	Time	Source	Destination	Protocol	Length	Info
118754	2014-07-08 22:41:13.553552	123.58.180.95	123.58.180.180	HTTP	71	HTTP/1.1 200 OK (text/html)
128330	2014-07-08 22:43:16.903953	123.58.180.94	123.58.180.180	HTTP	71	HTTP/1.1 200 OK (text/html)

No.	Time	Source	Destination	Protocol	Length	Info
128047	1628.186361	123.58.180.180	123.58.180.94	TCP	74	45173 > http [SYN] Seq=0 Win=5840
128284	1631.183038	123.58.180.180	123.58.180.94	TCP	74	[TCP Retransmission] 45173 > http
128285	1631.183160	123.58.180.94	123.58.180.180	TCP	74	http > 45173 [SYN, ACK] Seq=0 Ack=
128286	1631.183174	123.58.180.180	123.58.180.94	TCP	66	45173 > http [ACK] Seq=1 Ack=1 Win

1秒问题



Filter: tcp.analysis.retransmission and tcp.flags.syn == 1 | Expression... | Clear | Apply | Save | msgId | msgOne | msgTwo | retransmission | syn

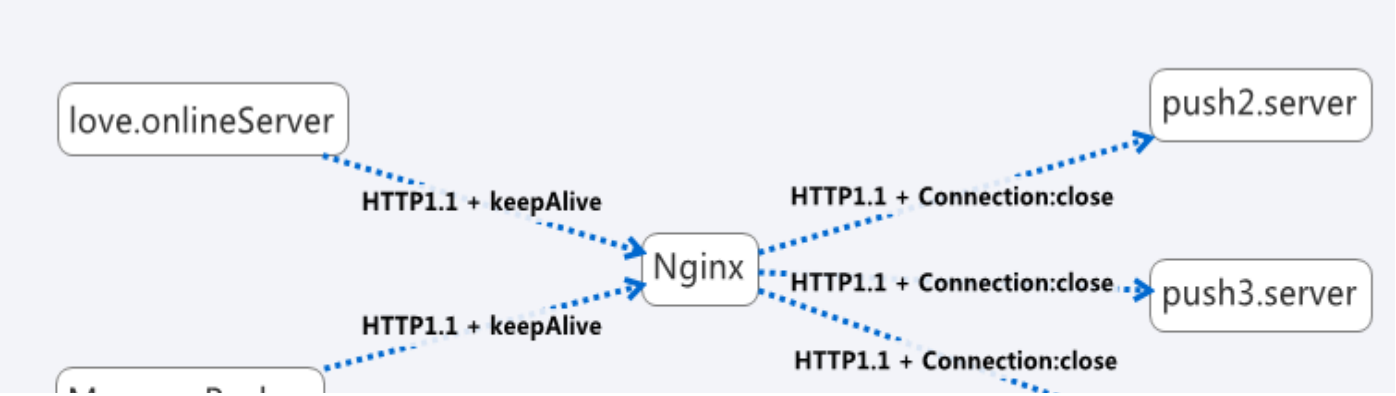
No.	Time	Source	Destination	Protocol	Length	Info
424	2014-07-18 11:02:32.457993	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 47048 > http
8707	2014-07-18 11:06:39.642550	114.113.201.21	123.58.180.94	TCP	70	[TCP Retransmission] 41422 > http
10880	2014-07-18 11:07:40.466990	114.113.201.21	123.58.180.94	TCP	70	[TCP Retransmission] 41456 > http
15945	2014-07-18 11:10:06.578521	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48266 > http
16239	2014-07-18 11:10:13.962979	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48268 > http
16773	2014-07-18 11:10:27.869561	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48278 > http
16774	2014-07-18 11:10:27.869597	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48282 > http
18193	2014-07-18 11:11:07.578495	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48312 > http
19742	2014-07-18 11:11:50.113996	114.113.201.21	123.58.180.95	TCP	70	[TCP Retransmission] 48375 > http

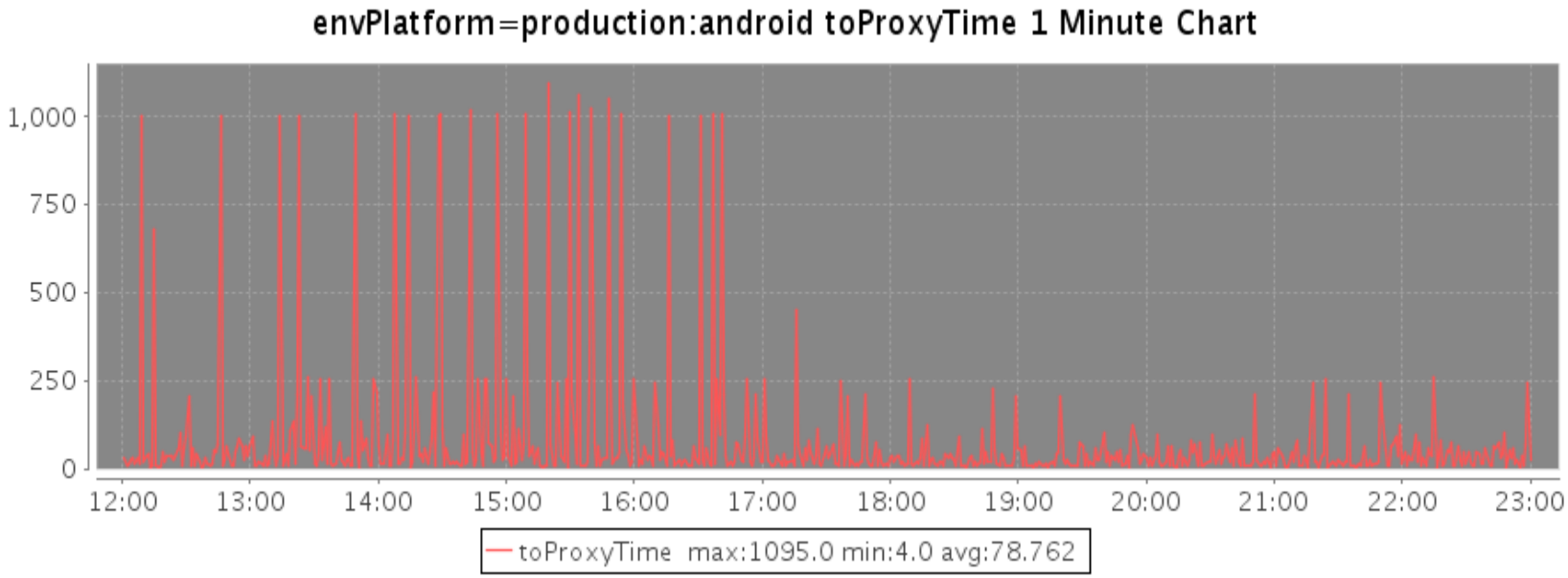
No.	Time	Source	Destination	Protocol	Length	Info
18465	2014-07-09 10:38:16.795470	114.113.201.21	123.58.180.94	TCP	70	40596 > http [SYN] Seq=0 Win=584
18535	2014-07-09 10:38:19.793986	114.113.201.21	123.58.180.94	TCP	70	[TCP Retransmission] 40596 > htt
18536	2014-07-09 10:38:19.794252	123.58.180.94	114.113.201.21	TCP	70	http > 40596 [SYN, ACK] Seq=0 Ac
18537	2014-07-09 10:38:19.794277	114.113.201.21	123.58.180.94	TCP	66	40596 > http [ACK] Seq=1 Ack=1 W

3秒问题

解决办法

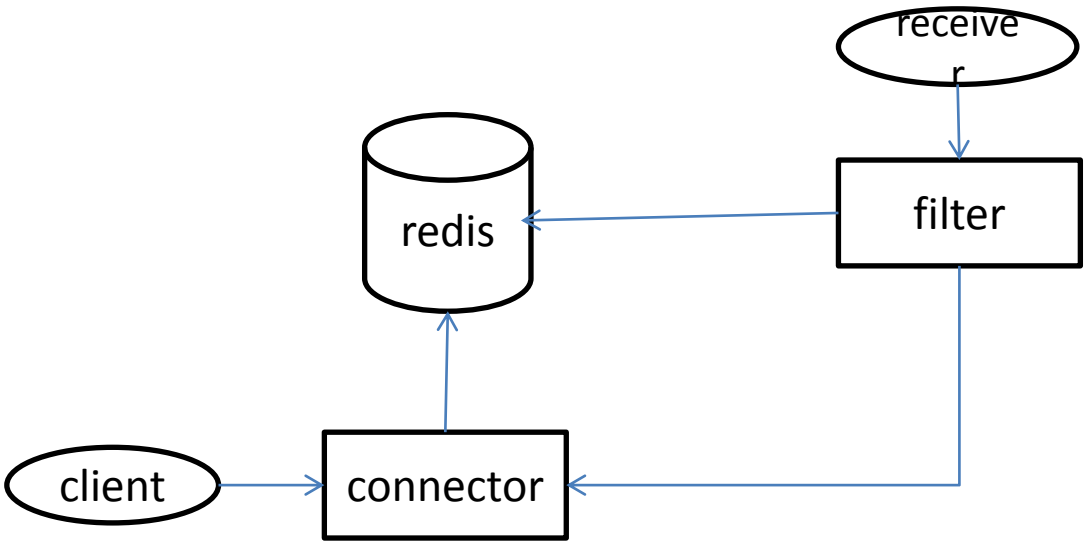
- 解决办法： Nginx到后端开启keepAlive





消息推送平台挑战 -- 新闻客户端的实时性要求

- 一分钟内广播消息给数千万个客户端
- 测试结果
  - 广播推送：15秒内推完
  - 按附件推送： 100万需要5分钟，瓶颈在filter



广播优化策略

- 批处理所有操作
  - 批处理查redis
  - 批量发消息到连接服务器
  - 优化后附件广播100万条在50秒之内推完

- 退避策略
  - 通过MQ自身的机制退避，广播消息在push完之后ack
  - 当广播的数量过大时， 适当退避，让出CPU

消息实时性 -- GC

- Node.js的GC
  - 基于V8，分代收集
  - 单进程吃不了太多内存，保持在600M以下
  - Young GC的时间很短，在内存低的情况下Full GC也能控制在30ms之内
  - --max-new-space-size=2048
- Java的GC
  - 分代收集
  - 文档多，很复杂的参数
  - CMS，G1对实时应用很重要
  - 一般建议java内存4G以下，但由于连接服务器吃内存的要求，配到8G或以上

易信GC

- 长连接服务器的内存管理
  - 吃内存
  - 实时性要求高，千万不能出现几秒的full gc ，需要采用CMS
  - 新生代的产生数据较多，不符合新产生的数据生命周期短的年青代规则
  - 使用CMS时千万不要promotion failure

易信GC

+UseConcMarkSweepGC

-XX:SurvivorRatio=3

-Xms12G -Xmx12G -Xmn2g

-XX:CMSInitiatingOccupancyFraction=80

-XX:+PrintPromotionFailure

-XX:+UseCMSCompactAtFullCollection

-XX:MaxTenuringThreshold=8

-XX:+CMSClassUnloadingEnabled

总结

- 实时应用架构
  - C/C++ 微信
  - Node.js & pomelo 消息推送平台
  - Java & netty 易信
  - Erlang whatsapp
  - Golang 360 七牛
- 有垃圾收集器的语言一样适合开发实时应用
- 条条大路通罗马，什么语言或工具不重要。理解架构的本质

Q&A

THANKS

SequeMedia 盛拓传媒 | IT168 www.it168.com | ChinaUnix | ITPUB