奇艺的极致网络

×

王伟峰

爱奇艺
工程师

目 录

01.网络治理概论

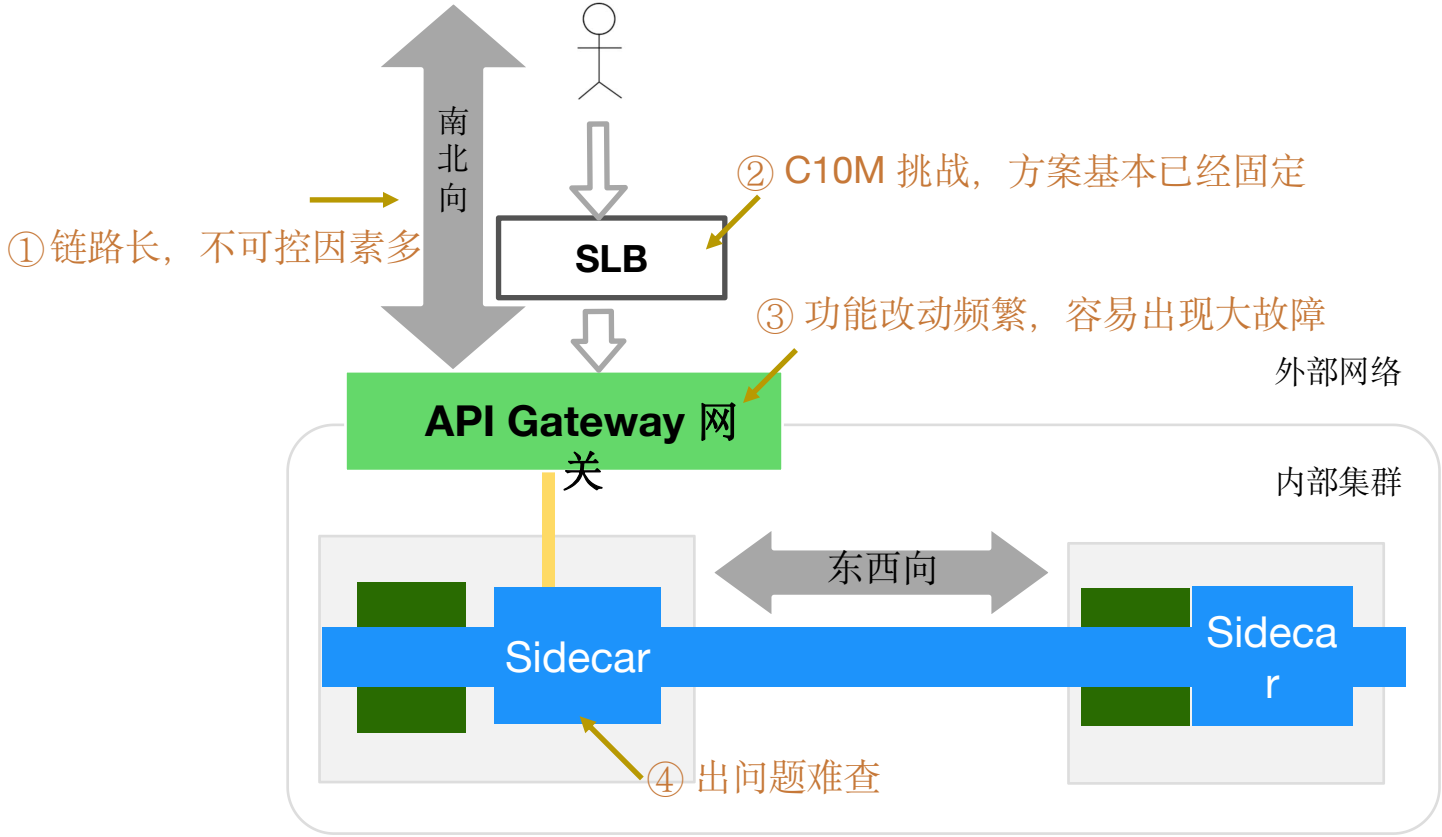
- 服务/网络治理的约束理论：**效能最低的阶段**决定着整个系统的生产能力。
- 网络治理的两个方向：**东西向**和**南北向**。

1. **南北向**：端到服务器之间的链路

- 链路不可靠
- 链路非常长（例如，国际网络）
- 协议僵化问题

2. **东西向**：集群内部的流量

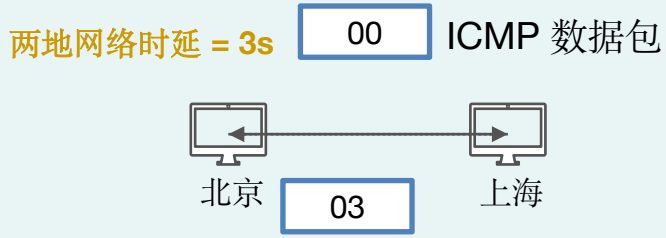
- 高并发挑战 （C10M）
- 高可用挑战（多活系统）
- 考验基础架构建设能力



01.部分网络指标说明

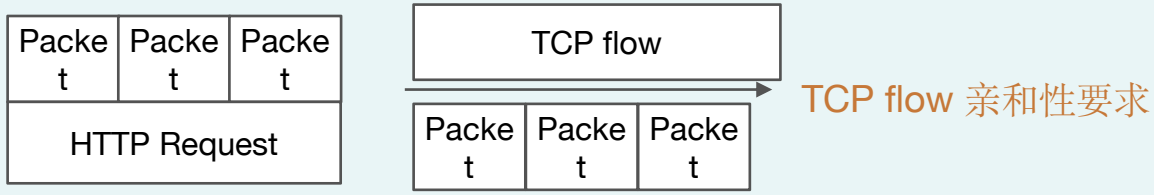
用 RTT 评估网络**链路质量**，用 TTFB 评估**服务质量**，用 PPS 评估服务**入口承受能力**。

评估两地的网络质量
RTT：Round-Trip Time



首字节时间：评估服务质量
TTFB：Time To First Byte

每秒包处理数量
PPS：Packet Per Second



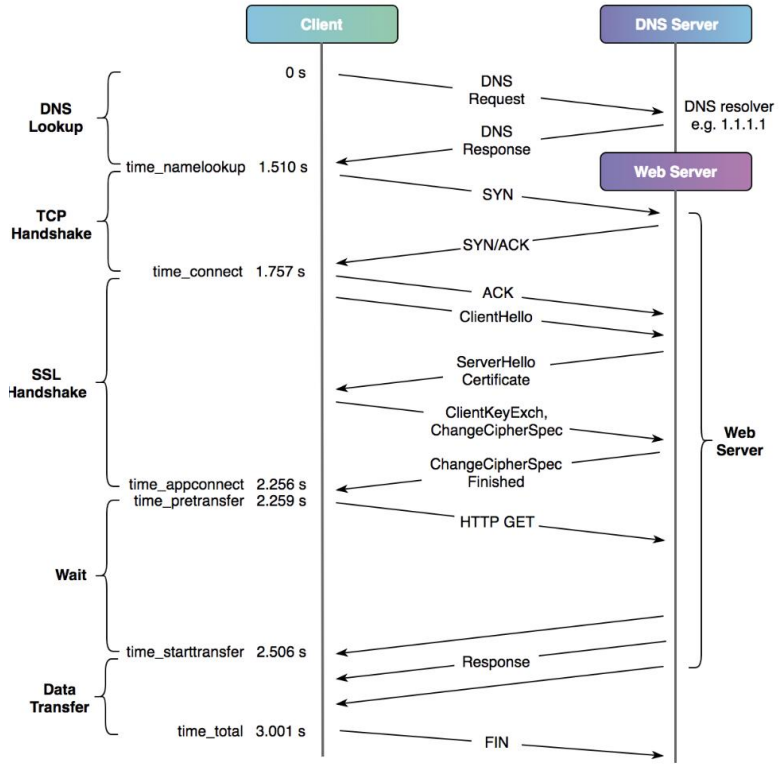
02

南北网络治理

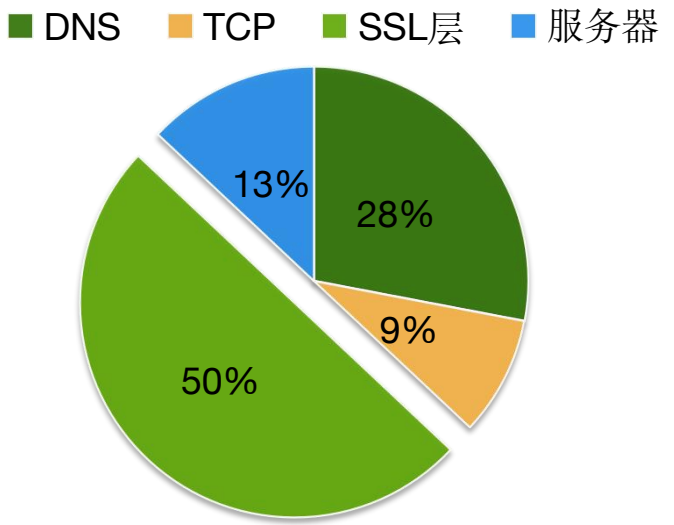
2.1 南北网络链路分析

一个完整、无缓存、未复用连接的 HTTPS 请求阶段：**DNS 域名解析、TCP 握手、SSL 握手、服务器处理、内容传输。**

一个 HTTPS 请求流程分析

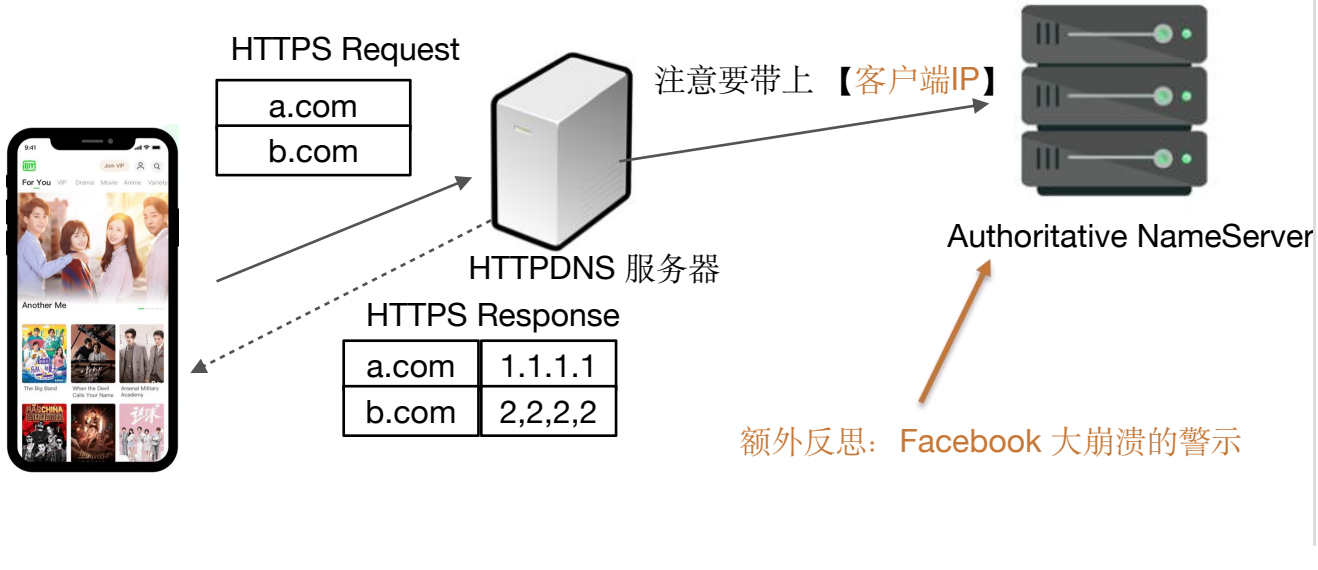


HTTPS 请求阶段消耗分析



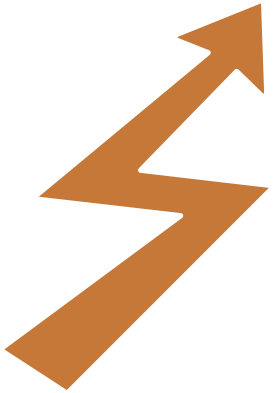
2.2 DNS服务治理

问题： DNS 问题比想象中严重 **延迟高**（最高 1000 ~ 2000 ms）、解析不准、海外存在一定量的劫持。
解决： 客户端使用 HTTPDNS 预解析



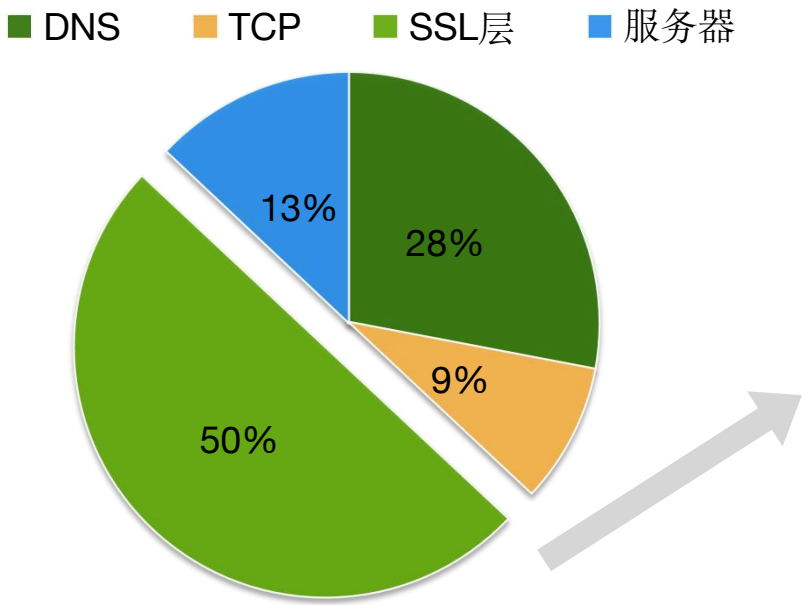
实践效果

延迟改善 12%



2.3 SSL层优化

问题： 一个 HTTPS 请求 SSL 层（TLS 握手 + 加解密 + 证书校验）延迟占比 50%！
解决： 升级 TLS 协议、弃用 RSA 证书改成 ECC 证书



TLS 协议 密钥交换
校验证书链
内容加解密
…

优化方向

- 协议升级
- 升级 ECC 证书

内置 ECDSA 公钥的证书一般被称之为 ECC 证书，内置 RSA 公钥的证书就是 RSA 证书.
相比 RSA，ECC 证书具有安全性高，处理速度更快的优点

2.3 SSL层优化效果

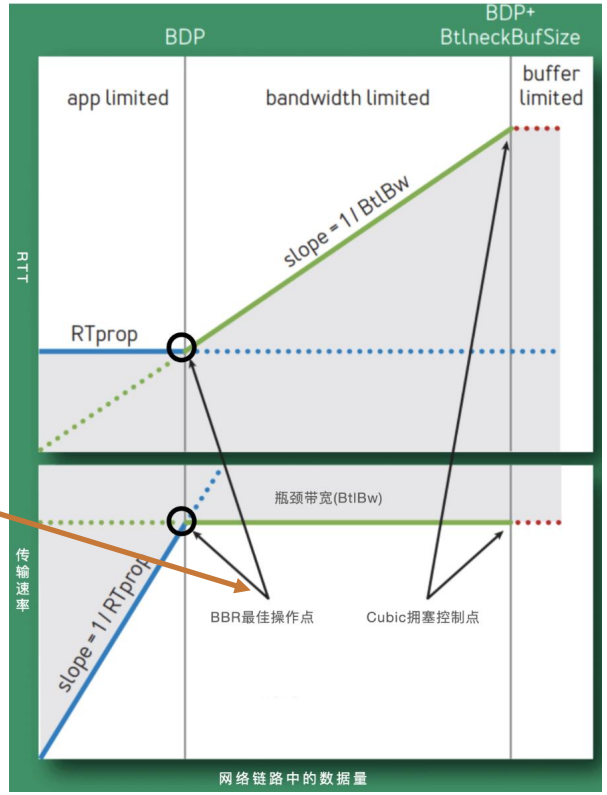
从 SSL 加速的结果上看，使用 ECC 证书较 RSA 证书性能提升很多，即使 RSA 使用了 QAT 加速比起 ECC 还是存在差距。
另外 QAT 方案的硬件成本、维护成本较高，综合考虑建议使用 TLS1.3 + ECC 证书方式。

推荐 TLS 1.3 + ECC 证书			
场景	QPS	Time	单次发出请求数
RSA 证书 + TLS1.2	316.20	316.254ms	100
RSA 证书 + TLS1.2 + QAT	530.48	188.507ms	100
RSA 证书 + TLS1.3	303.01	330.017ms	100
RSA 证书 + TLS1.3 + QAT	499.29	200.285ms	100
ECC 证书 + TLS1.2	639.39	203.319ms	100
ECC 证书 + TLS1.3	627.39	159.390ms	100

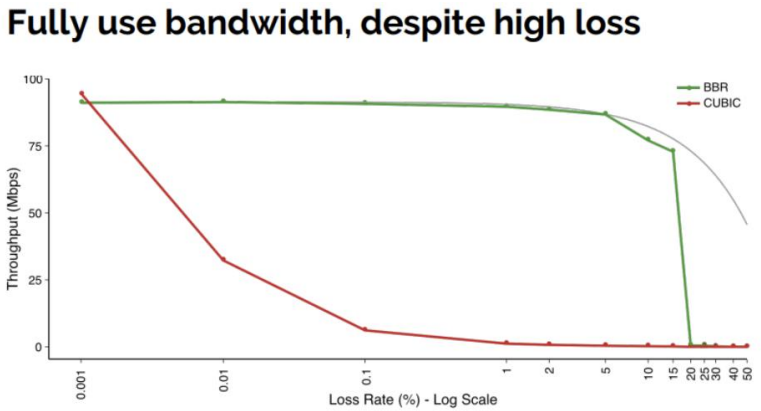
2.4 链路层优化

问题：网络传输没想象的好！路由节点越多、物理链路越长，BDP 就越高，**基于丢包的拥塞控制，无法充分利用现代网络链路。**
解决：使用 BBR，网络吞出率提升 45 %，弱网环境下优势明显（轻微丢包率）。

BBR 原理：寻找网络效能最大化
（延迟极小值 + 带宽极大值）操作点



BBR vs Cubic

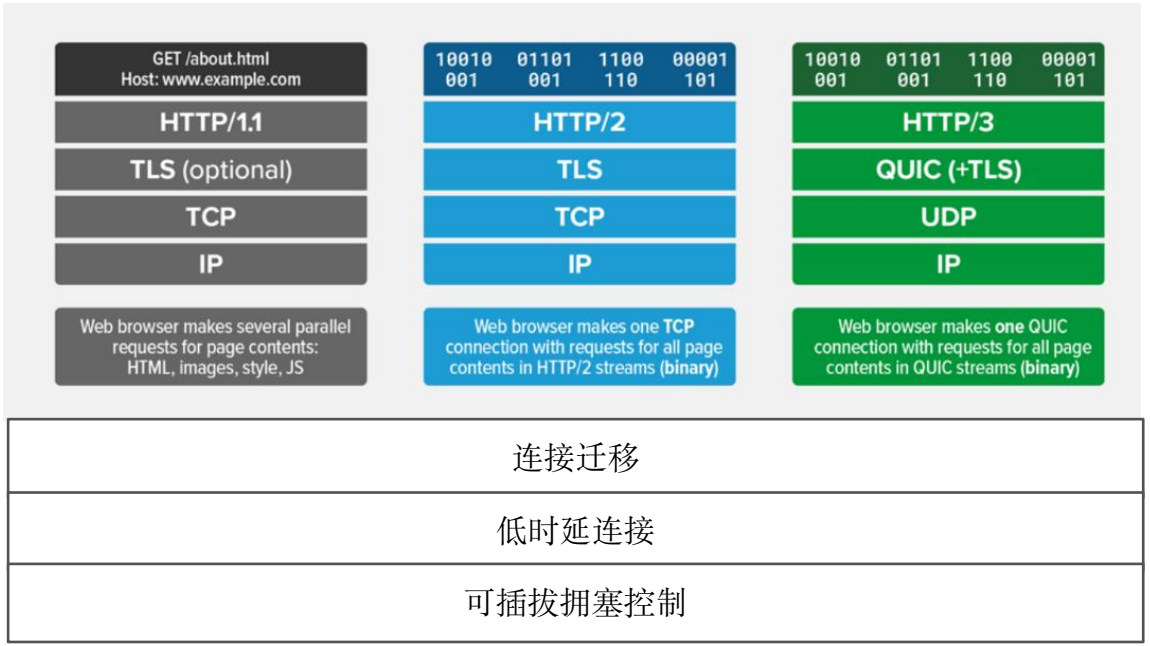


BBR vs CUBIC: synthetic bulk TCP test with 1 flow, bottleneck_bw 100Mbps, RTT 100ms

BDP (带宽时延积) = BtlBw(bottleneck bandwidth,瓶颈带宽) * RTprop(round-trip propagation time, 两地最小时延)

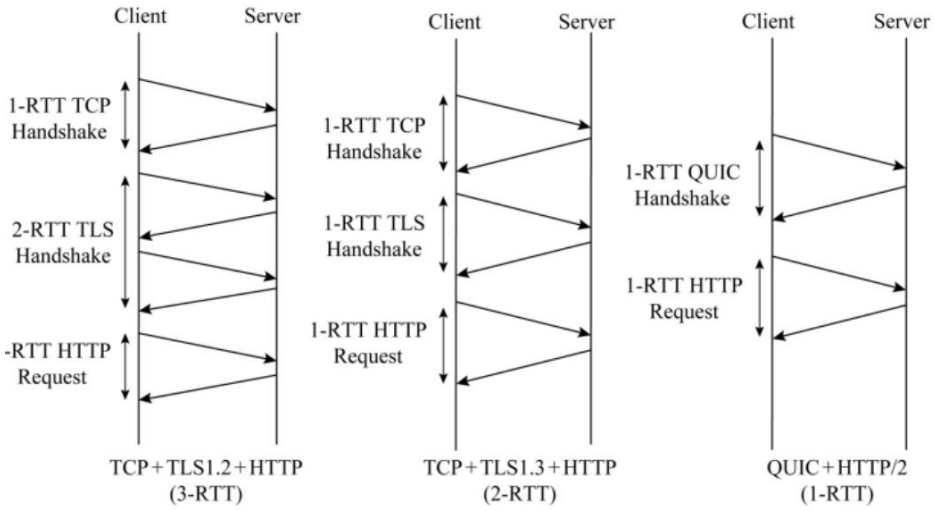
2.5 快速 UDP 网络连接

背景：2022 年 6 月 6 日，HTTP/3 标准化为 RFC 9114，古老的 TCP 协议已不再适合现代移动互联网。
特点：通过减少非必须的协商步骤来提升性能。



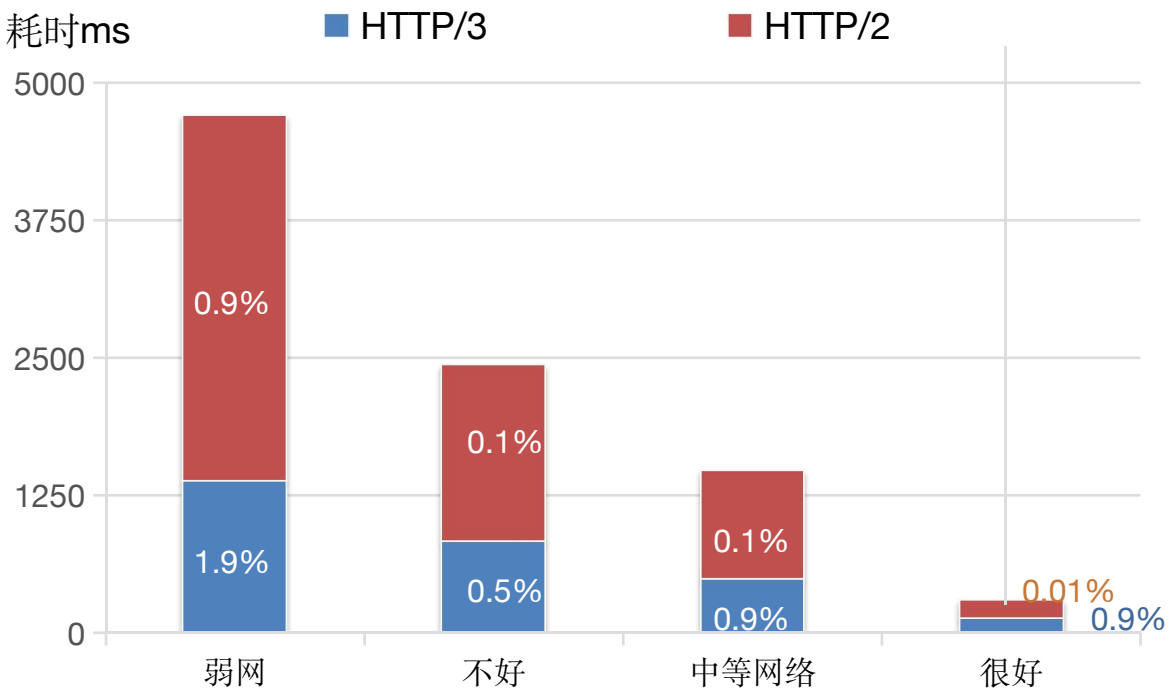
连接迁移
低时延连接
可插拔拥塞控制

各 HTTP 协议 RTT 对比



2.5 QUIC 实践效果

验证条件：全国、各网络状态用户 1000+，分别用 HTTP/2 和 HTTP/3 探测延迟和请求失败率
探测结果：延迟测试中 **QUIC 比 HTTP/2 降低 50%** 左右，但在失败率评测中 **QUIC 高于 HTTP/2**



实践结论

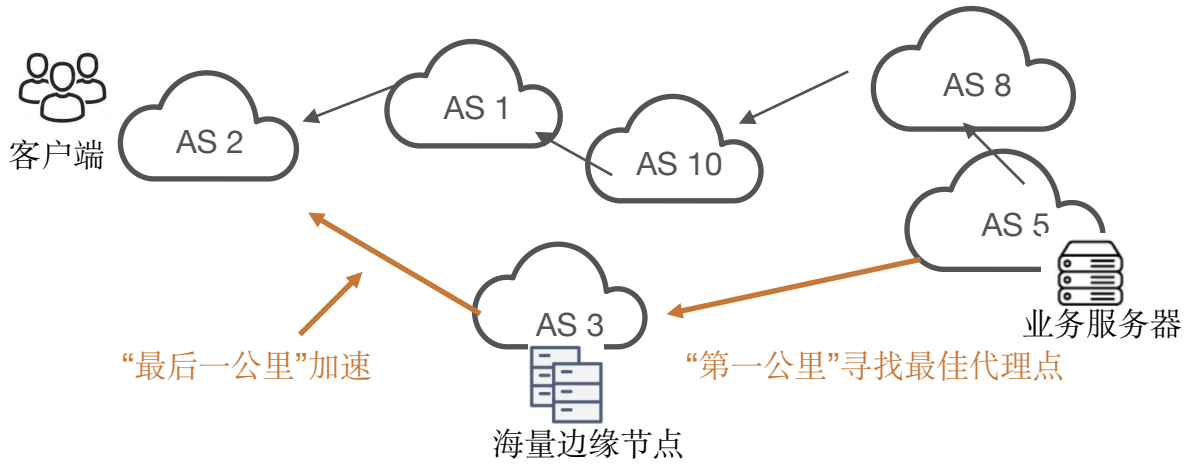
客户端问题
- 应用适配成本和收益之间的权衡
- 过度期间，新旧网络库要实现兼容，以及容错降级

服务端问题
- 网络事件模型需要适配 QUIC 协议栈做调整
- 要和 TCP 做兼容
- 虽然 TCP 是个古老协议，但也经过了无数优化，
- QUIC 在内核层的优化还需要时间

2.6 网络层优化

问题： 无论如何优化，链路瓶颈无法突破。
解决： 动态服务使用边缘节点进行代理加速。

两个节点之间往往并不是路由路径最短，而是费用最低的路由策略!
DSA 服务原理：“花钱”越过瓶颈链路!



海外链路加速效果

区域	直连	Akamai 加速	提升
Bangkok	0.58s	0.44	31%
jakarta	0.57s	0.44	31%
Kuala Lumpur	0.52s	0.38	36%
Taibei	0.51s	0.40	37%
Hanoi Bac Mai	0.54s	0.41	30%
Singapore	0.58s	0.39	48%
Hong Kong	0.38s	0.24	58%
Tokyo	0.60s	0.45s	32%
Surabaya	0.67s	0.52s	29%
Manila	0.46s	0.34s	36%

2.7 南北网络治理总结

整体治理思路：减少非必须的协商步骤（减少RTT）、减小传输量（压缩）

网络层

- 使用 anycast IP 技术实现多地同服
- 使用动态加速服务改善海外链路

链路层

- 使用更现代的传输协议 QUIC
- 高 BDP 、长链路下 使用 BBR 拥塞控制
- TCP 协议优化
  - 优化保守参数，失败立即传递应用层
  - fast open、
  - early data实际上会被阻断，不起作用

应用层

- 使用 HTTPDNS
- 使用 TLS1.3
- 使用 ECC 证书
- 使用 Brotli 压缩

03

东西向网络治理

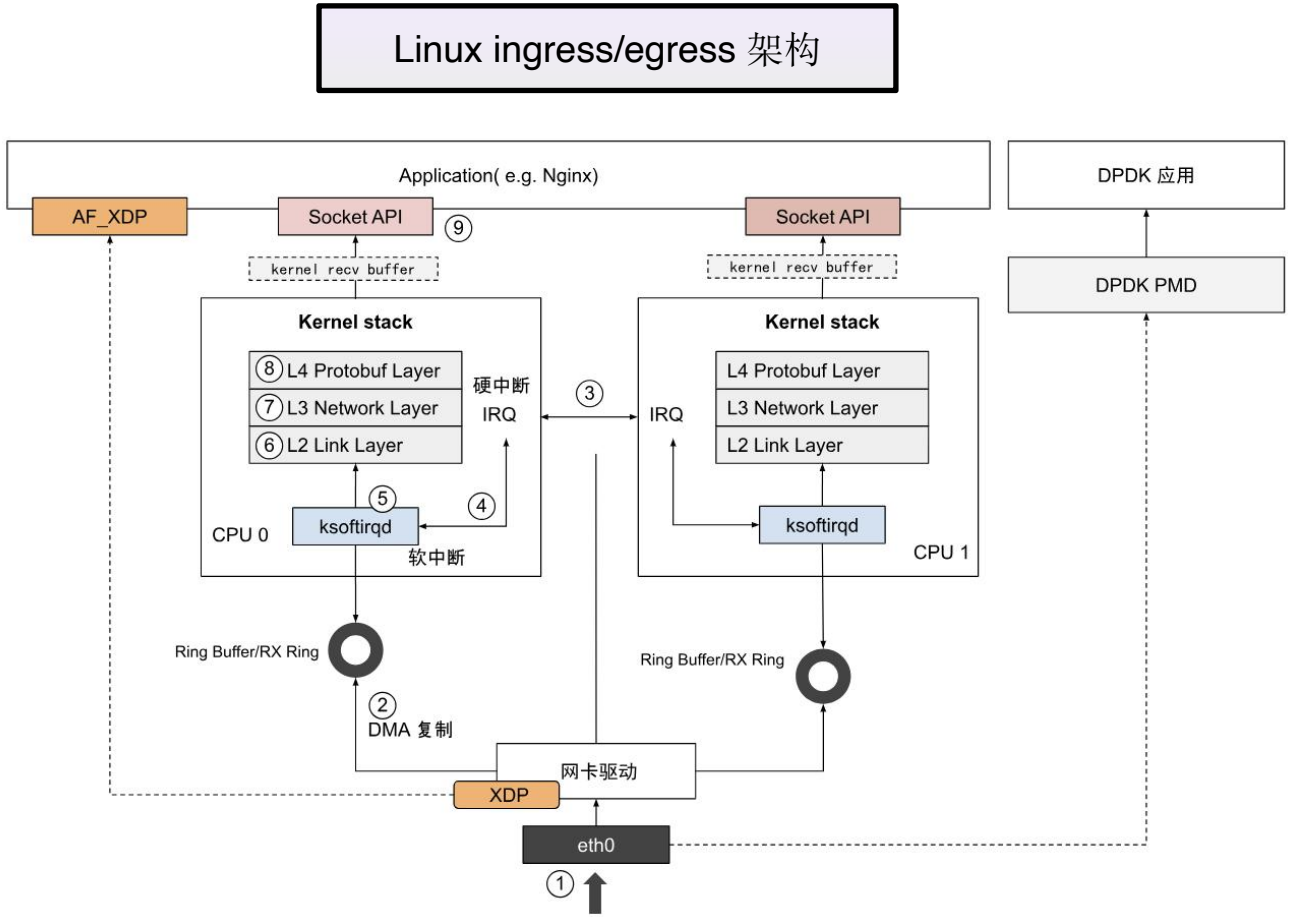
3.1 东西向网络治理概论

Linux 传统网络的问题

- C10K 已经发展到 C10M（单机1000万并发）
- 云计算中大规模集群服务治理
- ServiceMesh 中 Sidecar 代理模式加重了网络消耗
- 硬件性能提升，但软件跟不上

由问题到解决方案

- kernel bypass DPDK
- XDP Linux 高性能网络框架
- RDMA 远程直接访问内存



3.2 东西向网络核心是代理

系统的入口

不论是 LVS 还是 Nginx，流量已经到 L4，
我们统称代理

节点的入口

听名字就是个“代理”

服务的入口

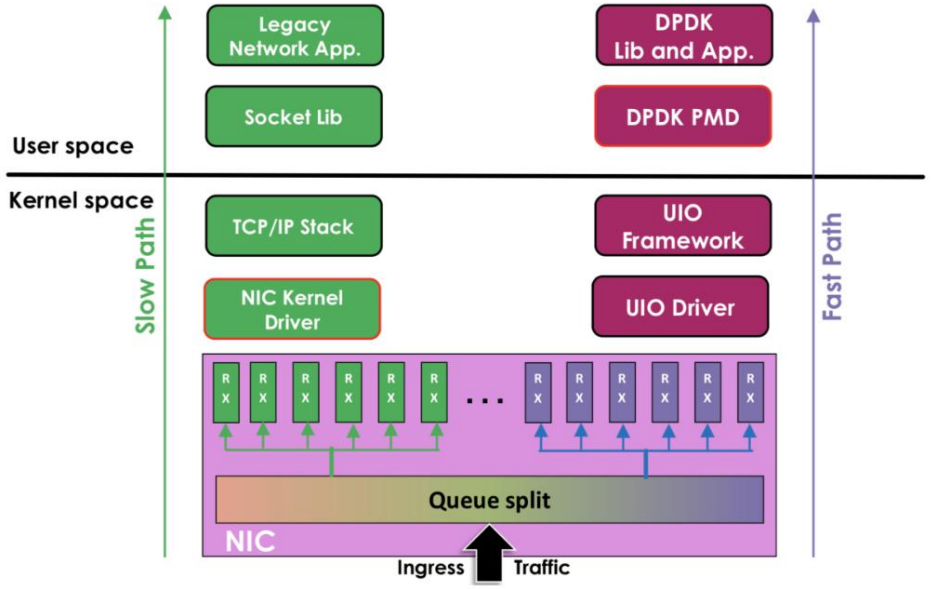
ServiceMesh

ServiceMesh 中的 Sidecar 本质也是个代理

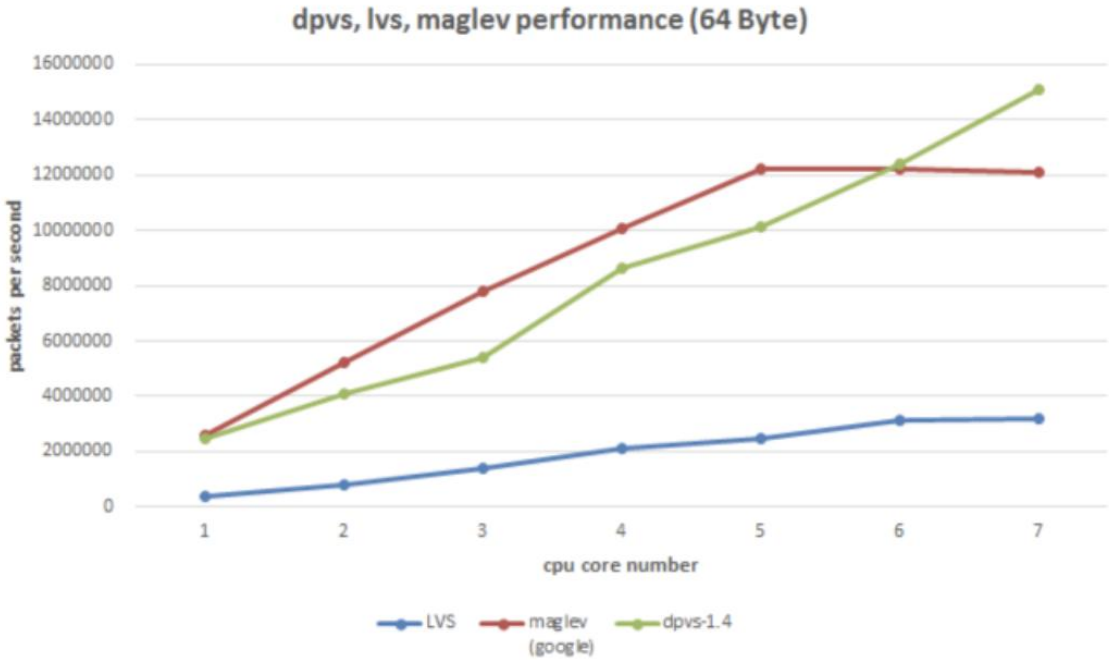
3.3 代理治理的黑科技 - 内核旁路

如果能将单机性能提升 10 倍甚至百倍，无论是从硬件投入还是运营成本上来看都能带来非常可观的成本削减

内核技术 vs 内核旁路技术



DPVS vs LVS PPS转发包效率对比

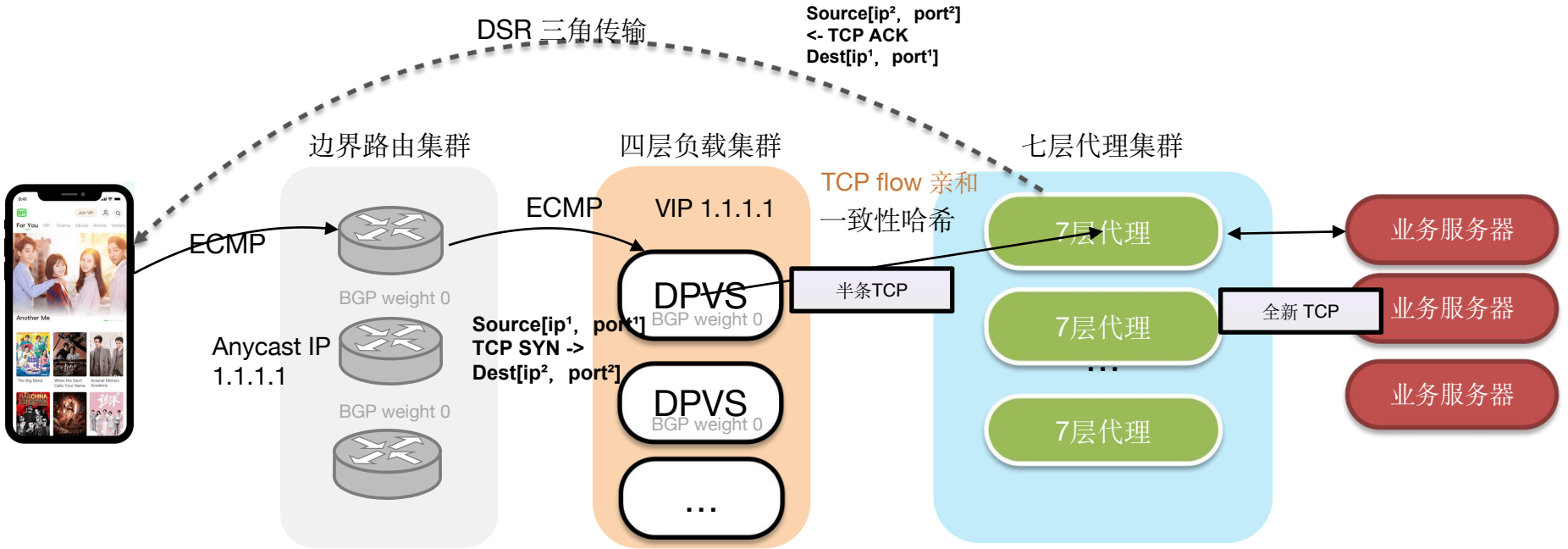


DPDK：数据从网卡 -> DPDK 轮询模式-> DPDK 基础库 -> 业务
（无内核协议栈层层封装、无各类中断、无各类用户态到内核态转换）

DPDK 加速访问较传统内核 LVS **提升 600%**

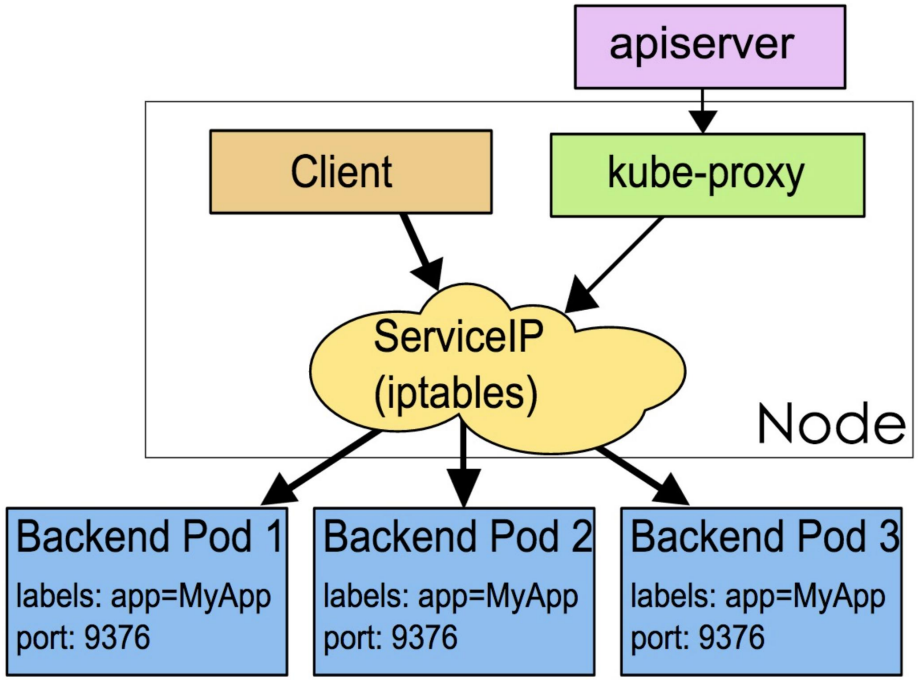
3.4 系统的入口

概念：基于集群的一致性哈希容错和可扩展设计的入口多级负载均衡
特点：通过ECMP保证每个 TCP flow 到达同一个节点。边缘路由和四层负责均衡按需添加，预留足够的突发量和容错的前提，充分利用资源。

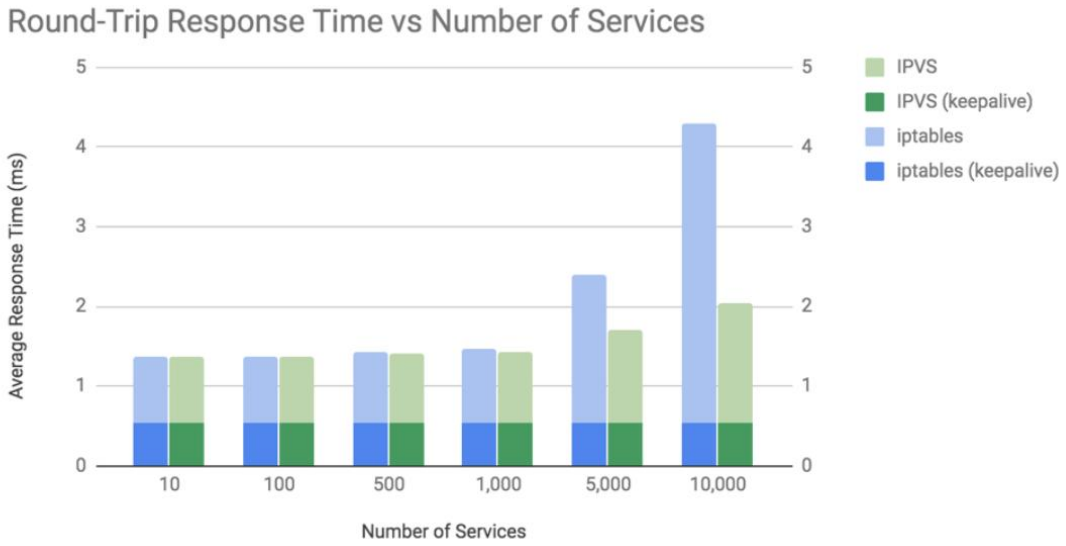


3.5 节点的入口

问题：K8s 的 Service Ip 本质是操作 iptables 创建自定义链，然后把流量转发给各个 Pod，iptables 的问题 线性匹配，全量更新!
解决： kube-proxy 模式改为 **IPVS**，更进一步使用使用 eBPF 替换 kube-proxy，例如 Cilium。



当 1000 个服务（10000 个 Pod）以上时，会开始观察到差异



3.5 服务的入口

问题： 既然 sidecar 是问题，那么就去掉 sidecar
解决： 从 proxyless 到 Sidecarless 再到 **中心化的Ambient Mesh**

- 需要一个 Agent 与控制平面交互
- 流控能力被集成在 gRPC 库中

Proxyless == 传统SDK

- L3/L4 的流量使用 eBPF 支持
- L7 公共代理服务

Ambient Mesh 可以被理解为一种“中心化代理模式”

侧重于通过共享和中心化的代理进行流量管理，
以替代位于应用容器旁边的 Sidecar 代理。

本课件内容来源

THANK YOU