从TCP的角度理解速度优化

叔 度

@淘宝 - 核心系统研发 - 服务器平台组

课程大纲页

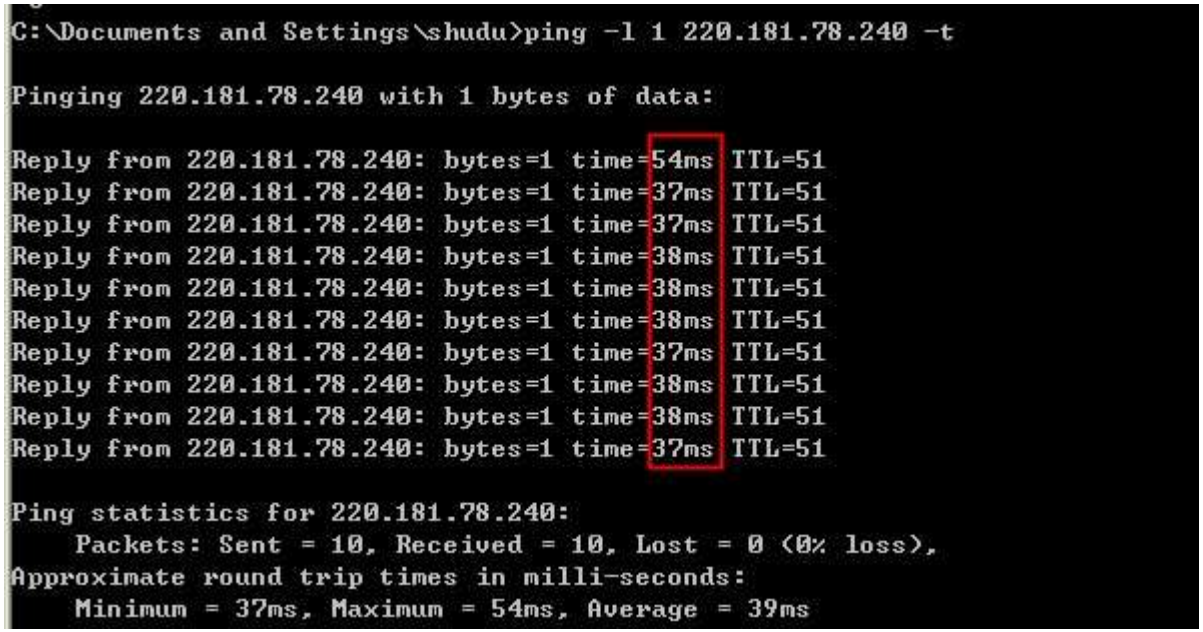
- 带宽和延时
- TCP协议核心概念
- TCP影响速度/性能的几个案例
- 针对TCP的特点做应用开发
- 总结

问题

- 杭州某用户带宽2Mbps，在想要下载一个位于北京的1个字节的文件，需要多少时间？
  - 1 * 8 / (2 * 1024 * 1024) = 4微秒？

实际情况

- 在杭州办公室ping淘宝北京电信CDN
  - 延时值为38毫秒

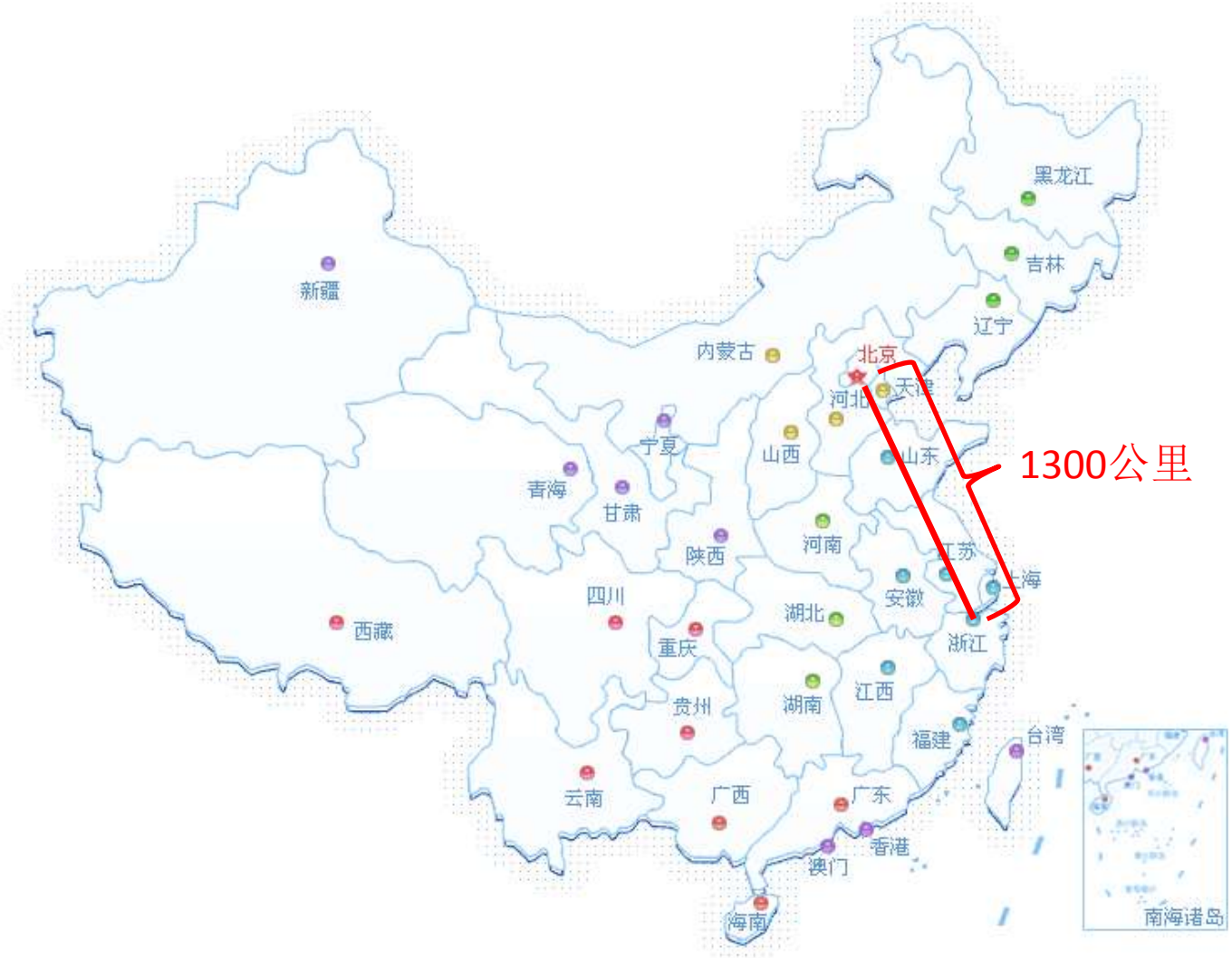


Why?

原因

- 比特在网络上要以光/电波传播
- 北京和杭州之间很近，但是也是有距离的
- 光/电的传播速度没有快到可以忽略不计！

两地直线距离


黑龙江
吉林
辽宁
新疆
内蒙古
北京
天津
河北
宁夏
山西
山东
1300公里
青海
甘肃
陕西
河南
江苏
上海
四川
安徽
西藏
湖北
重庆
浙江
贵州
湖南
江西
福建
台湾
云南
广西
广东
香港
澳门
海南
南海诸岛

理论值

- 真空中30万公里每秒
- 光在玻璃的传播速度是真空的2/3
- 往返距离1300 * 2 = 2600公里
- 1300 * 2 / (300000 * 2 / 3) = 13毫秒

其他因素

- 路很难是直线，光纤也不可能全部是直线
- 光在光纤里面全反射传输
- 中间传输过程中的损耗，光电转换等
- 信号干扰，容错修复
- 所以需要乘上2~3倍左右的因子
  - 30 / 13 = 2.3（CM4机房）
  - 38 / 13 = 2.9（杭州办公室）

带宽

- 带宽的表示是比特，不是字节
- 我们用的宽带上行下行是不相等的
- 带宽代表了能最大同时在网络上接收的比特数
- 带宽和延时没有关系
  - 传输时间不是 大小/带宽！

中国网络的现状

- 带宽和非洲接轨
- 价格和欧洲接轨
- 世界上最遥远的距离是网通和电信的距离
- 中国的平均网速
  - 世界排名第78
  - 平均速度1.7Mbps

带宽 vs. 延时

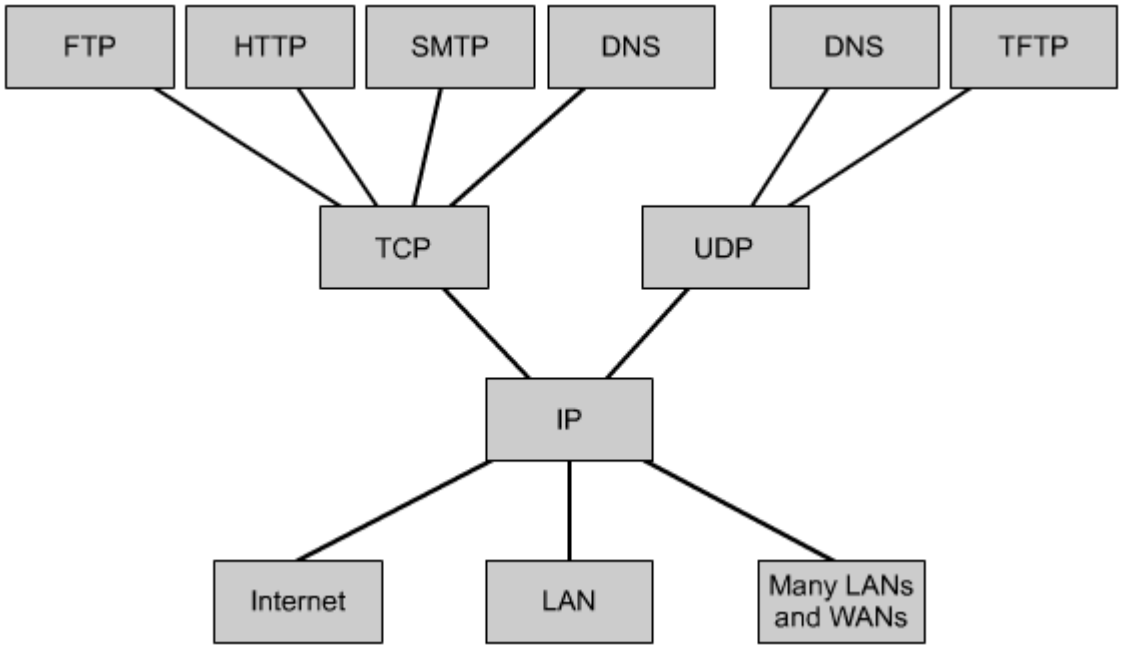
- 带宽和延时没有关系
- 可以提高带宽，但是无法降低延时
  - 两个铁球同时落地
    - 2G的带宽和2M的网速传输1个字节的时间是一样的

小结

- 光的传输需要时间
- 带宽和延时没有关系
- 你可以提高带宽，但是你无法降低延时
- 延时不能被忽略
- 延时对网络程序速度影响很大
- 中国的网络情况不理想

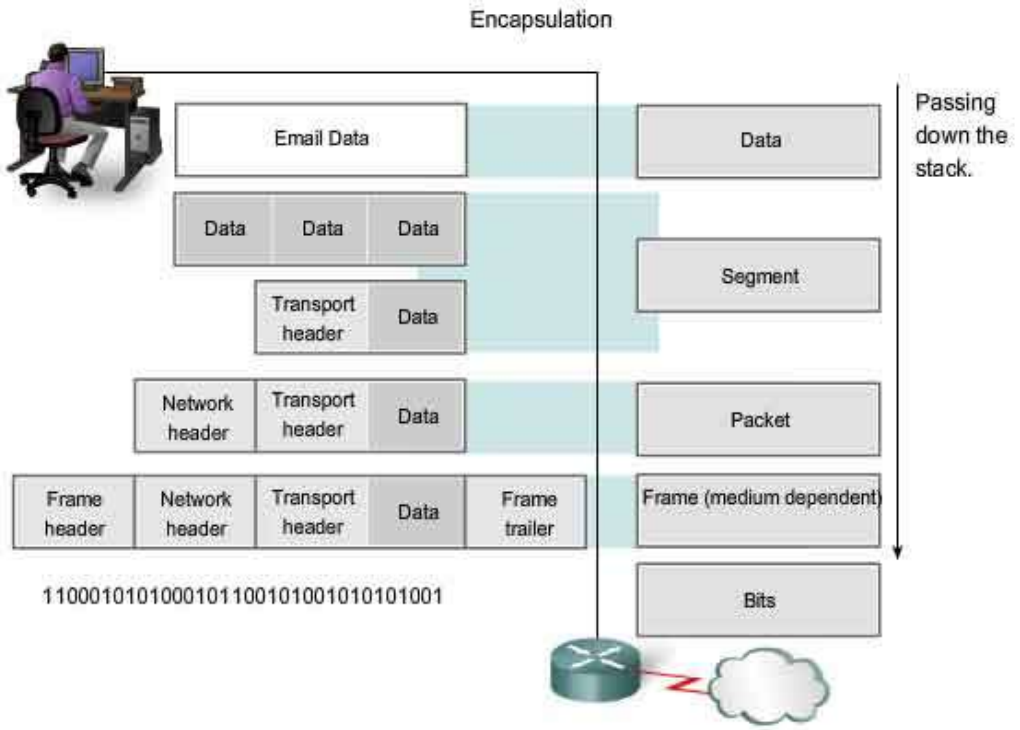
互联网的基石

- 应用最广泛的协议是HTTP
- HTTP和其他协议的基础是TCP



TCP/IP的层层封包

- 下层在上层的基础上加一个头（图示）
- 不同的层次的不同类型
  - Segment（TCP）
  - Packet（IP）
  - Frame（Link）



以太网传输的对象大小

- Frame
  - 1542
- MTU
  - 1500
- MSS
  - 1460
  - 一次TCP能传输的最大大小

记住这些数字！

TCP的基本特点

- 可靠的
- 面向字节流
- 面向连接
- 全双工
- 底层使用IP协议进行传输

IP协议的特点

- 无连接
- 基于包
- 不可靠
  - 包的顺序不能保证
  - 路由器可根据需要丢包

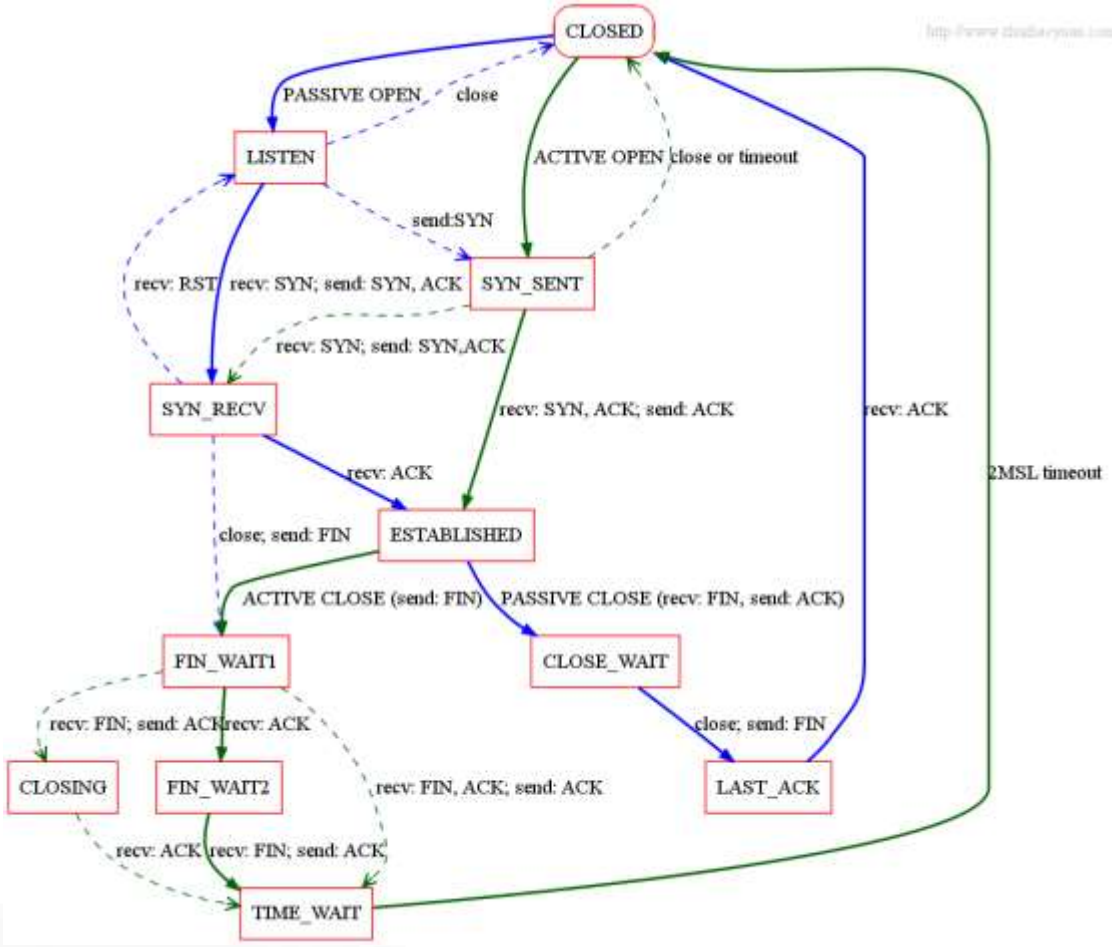
很多问题的根源

常见问题：为什么性能压测上不去

- 短连接
- 大量TIME_WAIT
- 性能上不去

TCP的状态

- 11个状态
- 三路握手
  - 主动打开
  - 被动打开
- 四路关闭
  - 主动关闭
  - 被动关闭



TIME_WAIT vs. CLOSE_WAIT

- TIME_WAIT
  - 主动关闭
  - 2MSL
- CLOSE_WAIT
  - 被动关闭
  - 99%意味着你的应用程序有bug

Linux内核网络参数调优

- 本地端口

  net.ipv4.ip_local_port_range = 1024 65535

- 优化短连接

  net.ipv4.tcp_fin_timeout = 15

  net.ipv4.tcp_tw_reuse = 1

  net.ipv4.tcp_tw_recycle = 1

真实案例：为什么传输速度上不去？

- 背景
  - CM4机房（杭州）和CM5机房（青岛）
  - 带宽 1G bits/s，延时 30ms
  - 传输大文件
- 问题
  - RHEL 5之间40MB/s
  - RHEL 4之间4MB/s

Why?

TCP的传输速度流控

- 原则
  - 快的发送方不能淹没慢的接收方
- 接收方窗口控制
  - 接收方通告窗口（awnd）
- 发送方窗口控制
  - 滑动窗口（swnd）
  - 拥塞窗口（cwnd）
  - swnd = min(awnd, cwnd)

TCP的窗口大小的上限

- 受套接字的缓冲区大小限制
- 内核会根据传输情况自动调整
  - 从Linux2.4起，发送端自动调整
  - 从Linux2.6.7开始，接收端自动调整
- 系统默认值
  - RHEL 4（2.6.9）

    net.ipv4.tcp_rmem = 4096 87380 174760

    net.ipv4.tcp_wmem = 4096 16384 131072
  - RHEL 5（2.6.18）

    net.ipv4.tcp_rmem = 4096 87380 4194304

    net.ipv4.tcp_wmem = 4096 16384 4194304

窗口的计算公式

- BDP（Bandwidth Delay Product）
  - 带宽（Bytes/s) * RTT（s）
  - 代表了能在网络上fly的最大字节数
- 问题定位
  - RHEL 4：

    131072 / (0.015 * 2) / 1024 / 1024 = 4.17MB/s
  - RHEL 5

    4194304 / (0.015 * 2) / 1024 / 1024 = 133.33MB/s

Linux内核网络参数调优

- 缓冲区大小

  net.core.rmem_default = 262144

  net.core.wmem_default = 262144

  net.core.rmem_max = 16777216

  net.core.wmem_max = 16777216

  net.ipv4.tcp_rmem = 4096 87380 16777216

  net.ipv4.tcp_wmem = 4096 65536 16777216

TCP的核心是拥塞控制

- 手段
  - 慢启动
  - 拥塞避免
  - 快速重传
  - 快速恢复
- 目的
  - 探测网络速度
  - 保证传输顺畅

慢启动

- 规则
  - 窗口从一个小的值开始
  - 指数增长
  - 上限阈值
- 合理性
  - 避免淹没慢的接收方
  - 避免网络瘫痪
- 问题
  - 往往慢启动还没终止，连接已经结束
  - 用户的速度极限还没到

优化案例：增大初始拥塞窗口

- 慢启动初始窗口值
  - 默认值2、3、4
  - Google的实验，10最佳
  - 我们的实验，7最佳
- 设置方法
  - 内核版本（< 2.6.30, >= 2.6.38）
  - ip route change
- 考虑因素
  - 性能提高百分比 +
  - TCP重传率的影响 -

优化案例：增大初始拥塞窗口

- 提升速度
  - 商品详情15%，减少下载时间311毫秒
  - 店铺提升20%
  - 基调数据

TCP拥塞窗口的增加方式

- 慢启动期间
  - 发送方每收到一个ack，拥塞窗口加1
  - 接收方收到2个包，发送1个ack

多少个来回

- TCP的基本交互过程：发送-等待-发送-等待…
- 等待时间的单位就是RTT

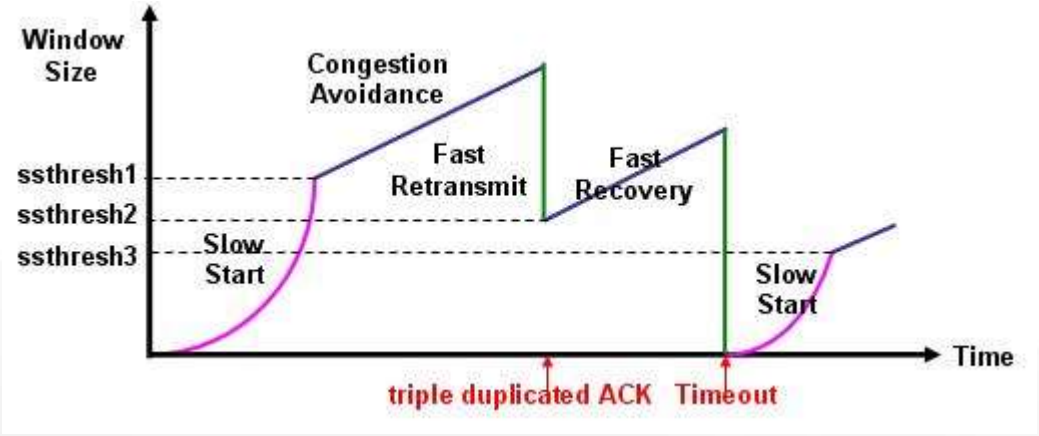
Round Trip	Congestion window size
1	3
2	4
3	6
4	9
5	13
6	19
7	28
8	42
9	44

节省多少个RTT？

- 节省多少个RTT就可以节省几倍的往返时间

TCP拥塞控制的核心

- 慢启动结束就进入拥塞避免
  - 到达ssthresh
  - 超时或丢包发生
- 拥塞避免
  - AIMD（线性增长，乘性减少）
  - 增长变缓（每RTT时间窗口加1）



真实案例：为什么淘宝首页这么慢？

- 移动用户
- 淘宝移动节点
- 咆哮体邮件（江苏移动 vs. 浙江移动）

发件人：[redacted]@js.chinamobile.com
2011-03-23 16:35:12

收件人：[redacted]@zj.chinamobile.com

抄送：[redacted]@js.chinamobile.com>; [redacted]@js.chinamobile.com>; [redacted]
[redacted]@js.chinamobile.com>; [redacted]@nj.js.chinamobile.com>; [redacted]
[redacted]@cz.js.chinamobile.com>; [redacted]<liusd@cz.js.chinamobile.com>

密送：[IMAGE]

主题：Fw：江苏公司访问浙江移动网内淘宝质量不佳的问题--------（咆哮体）如此牛X的网速！！！！！打开淘宝却需要如此多的时间，沮丧！！！！！！！见正文！！！！！！亲们，谁能告诉我该咋办！！！！！！！！！

胡总：您好！

近期全国移动的用户都在反应到浙江移动引入的淘宝网站质量不佳。详见 QQ 群中兄弟们的抱怨。

江苏公司目前对淘宝，均采用回源的方式访问，即访问浙江移动网内资源。

从初步测试看，ping 服务器的网络连通性指标很好，但是总体页面打开时间非常慢。

以淘宝首页为例，通过 HTTPwatch 抓包，其中 img02.gslb.taobao.com 解析为 211.138.122.240、211.138.122.250，均为浙江移动地址，Ping 时延只有 8ms 左右，但是打开该域名历时 25s 以上。

问题排查

- 服务淘宝首页的Nginx顶不住了吗？
  - 去年一年没有对Nginx服务器扩容
- 定位：不是Nginx，而是TCP重传率超高

```
[shudu@www195 ~]$ tsar --live
Time         ---cpu-- ---mem-- -----traffic---- ---load- ---tcp-- --sda---
Time           util     util    pktin  pktout    load1   retran     util
21/03-22:32    7.0      6.7     4.6K    6.1K      1.4     19.5      1.3
21/03-22:32    6.2      6.7     4.2K    5.5K      1.4     20.2      1.0
21/03-22:32    6.2      6.7     4.3K    5.7K      1.4     18.8      1.7
21/03-22:33    6.6      6.7     4.5K    5.9K      1.2     18.7      2.4
21/03-22:33    6.5      6.7     4.2K    5.6K      1.5     19.8      1.3
21/03-22:33    6.6      6.7     4.5K    6.0K      1.5     19.0      6.8
21/03-22:33    7.8      6.7     4.2K    5.6K      1.5     19.6     16.5
21/03-22:33    8.8      6.7     4.8K    6.4K      1.4     18.3     16.1
21/03-22:33    7.3      6.7     4.5K    6.0K      1.5     19.1      4.9
21/03-22:33    6.8      6.7     4.6K    6.2K      1.5     18.8      5.7
21/03-22:33    6.5      6.7     4.5K    6.0K      1.5     18.3      1.2
21/03-22:33    7.3      6.7     4.6K    6.2K      1.4     19.6      1.4
21/03-22:33    6.6      6.7     4.7K    6.3K      1.2     19.6      1.2
21/03-22:33    6.2      6.7     4.6K    6.2K      1.2     20.5      3.5
21/03-22:33    9.6      6.7     4.5K    6.1K      1.1     20.7     20.6
21/03-22:34    7.1      6.7     4.4K    6.1K      1.0     21.1      1.9
21/03-22:34    5.7      6.7     4.4K    6.0K      1.2     22.4      1.7
21/03-22:34    7.2      6.7     4.6K    6.2K      1.1     19.5      1.3
21/03-22:34    6.9      6.7     4.5K    6.1K      1.0     20.4      1.0
Time         ---cpu-- ---mem-- -----traffic---- ---load- ---tcp-- --sda---
Time           util     util    pktin  pktout    load1   retran     util
21/03-22:34    6.6      6.7     4.6K    6.1K      0.9     19.3      1.1
21/03-22:34    6.7      6.7     4.3K    5.7K      0.9     17.3      4.3
21/03-22:34    6.3      6.7     4.3K    5.5K      0.9     19.4      4.7
21/03-22:34    8.2      6.7     4.2K    5.4K      0.9     18.9     25.3
21/03-22:34    6.4      6.7     4.1K    5.6K      0.9     19.6      1.0
```

问题解决

- 运营商导致我们的网络丢包严重
  - 省内OK，出省质量巨差
- 换了一个IP，终于和谐了

06/04-20:05	15.6	6.7	10.3K	14.0K	2.5	4.3	6.5
Time	---cpu--	---mem--	-----traffic----		---load-	---tcp--	--sda---
Time	util	util	pktin	pktout	load1	retran	util
06/04-20:10	15.7	6.5	9.2K	11.9K	2.2	4.2	6.4
06/04-20:15	15.8	6.7	9.1K	11.6K	2.7	4.1	6.6
06/04-20:20	16.2	6.5	9.1K	11.7K	2.8	4.0	6.7
06/04-20:25	16.2	6.7	9.2K	11.8K	2.1	4.2	6.5
06/04-20:30	16.4	6.6	9.5K	12.1K	2.7	4.0	6.3
06/04-20:35	17.5	6.7	9.6K	12.2K	2.8	4.2	6.8
06/04-20:40	17.2	6.5	9.5K	12.2K	3.0	4.2	7.1
06/04-20:45	17.4	6.7	9.4K	12.1K	2.5	4.2	6.6
06/04-20:50	17.5	6.6	9.4K	12.1K	2.7	4.3	6.9
06/04-20:55	18.2	6.7	9.6K	12.2K	2.0	4.3	7.0
06/04-21:00	18.0	6.6	9.5K	12.1K	2.8	4.1	6.4
06/04-21:05	18.3	6.7	9.8K	12.6K	2.5	4.1	7.3
06/04-21:10	17.7	6.6	9.6K	12.4K	2.0	4.4	6.5
06/04-21:15	17.5	6.7	9.6K	12.4K	3.1	4.3	6.5
06/04-21:20	17.7	6.6	9.6K	12.3K	2.8	4.5	6.1
06/04-21:25	16.8	6.6	9.4K	12.0K	2.3	4.1	6.2
06/04-21:30	16.4	6.6	9.6K	12.3K	2.4	4.2	5.6
06/04-21:35	16.1	6.7	9.4K	12.0K	2.1	4.3	5.4
06/04-21:40	16.2	6.6	9.7K	12.4K	2.5	4.6	6.6
Time	---cpu--	---mem--	-----traffic----		---load-	---tcp--	--sda---
Time	util	util	pktin	pktout	load1	retran	util
06/04-21:45	15.6	6.7	9.3K	11.9K	3.1	4.4	5.1
06/04-21:50	15.7	6.5	9.3K	11.9K	2.5	4.5	5.9
06/04-21:55	15.3	6.7	9.1K	11.5K	2.3	4.3	5.6
MAX	18.3	7.6	11.7K	16.4K	5.6	5.8	9.0
MEAN	8.9	6.3	6.3K	8.5K	1.4	3.9	5.1
MIN	0.6	6.0	291.0	280.0	0.1	1.8	3.3

原因分析：丢包和超时的影响

- 重传是由丢包和超时引起的
- 丢包
  - 重新进入慢启动
  - 拥塞窗口阈值变为原来的一半，速度抖降
- 超时
  - 重传超时（RTO）
  - 指数后退
- 超时例子
  - 连接超时
    - 3 + 6 + 12 + 24 + 48 + 96 = 189秒

Linux内核网络参数调优

- 拥塞控制算法

  net.ipv4.tcp_congestion_control = cubic

服务器开发：理解TCP

- 操作的对象是socket读写缓冲区
  - send/write成功并不代表已经发送到对端
  - 应用程序中响应时间的含义
  - 明白设置缓冲区大小的影响
- 关键应用需要保证可靠性
- 需要应用级别的心跳检测

服务器开发：高效读写

- 合理使用重要的TCP选项
  - TCP_DEFER_ACCEPT
  - TCP_CORK
  - TCP_NODELAY
- 使用writev/readv
- 真正理解non-blocking的套接字编程
- 真正理解epoll

服务器开发：诊断

- Linux内核TCP统计信息
  - /proc
    - cat /proc/net/netstat
    - cat /proc/net/snmp
  - netstat
    - netstat –s
- 重视dmesg中和网络相关的信息

Linux内核网络参数调优

- Listen队列

  net.ipv4.tcp_max_syn_backlog = 16384

  net.core.somaxconn = 2048

- 网卡的接收队列

  net.core.netdev_max_backlog = 10000

- 网卡发送队列

  ifconfig eth0 txqueuelen 10000

前端开发：文件尺寸

- 体积越小越好
  - 去空格、注释
- 启用gzip
  - 针对文本类型的文件
  - 设置允许gzip的最小尺寸，1K左右
  - 在速度和压缩比之间平衡

前端开发：cookie和URL的影响

- 现在的网络往往上传带宽小
- 分成几个包影响GET/POST的速度
  - 3K，2个包（MSS）
- 减小cookie和URL
  - 在HTTP头，不能被gzip压缩
  - 挑战浏览器的URL限制没意义

前端开发：CSS/Javascript

- HTML中把assets放前面
- 同等重要程度下，尺寸小的assets放前面
- 使用CDN Combo功能，减少请求数

前端开发：长连接 vs. 短连接

- 连接建立（三路握手）需要的时间
- 权衡资源占用与响应时间
- 针对应用特点设置时间

勿在浮沙筑高台 – 进阶阅读

- RFC文档
- 源码之前，了无秘密（RTFSC）
  - Linux内核TCP/IP协议栈实现

谢 谢 !