

网络开发那些事

yaocoder

2013-8-11

个人介绍：

网络ID: **yaocoder**，至今已有超过七年的IT从业经验，在架构设计、产品设计、网络编程、团队管理方面有较强的实践经验，对互联网、高性能、分布式处理技术兴趣浓厚。有丰富的**C++**语言经验，对**python**、**golang**颇感兴趣。此外热爱读书，音乐，电影，篮球，美女.....

个人博客: <http://yaocoder.blog.51cto.com/>

个人邮箱: **yaocoder@gmail.com**

网络开发究竟有些什么？

从招聘网站上截取典型的几段：

- 熟悉Socket编程，熟悉Tcp/Ip协议栈；
- 熟悉TCP/IP协议、UDP协议，有相关的协议开发经验；
- 熟悉网络编程/多线程编程技术；

关键词： **TCP/IP**， **Socket**， 协议

多线程

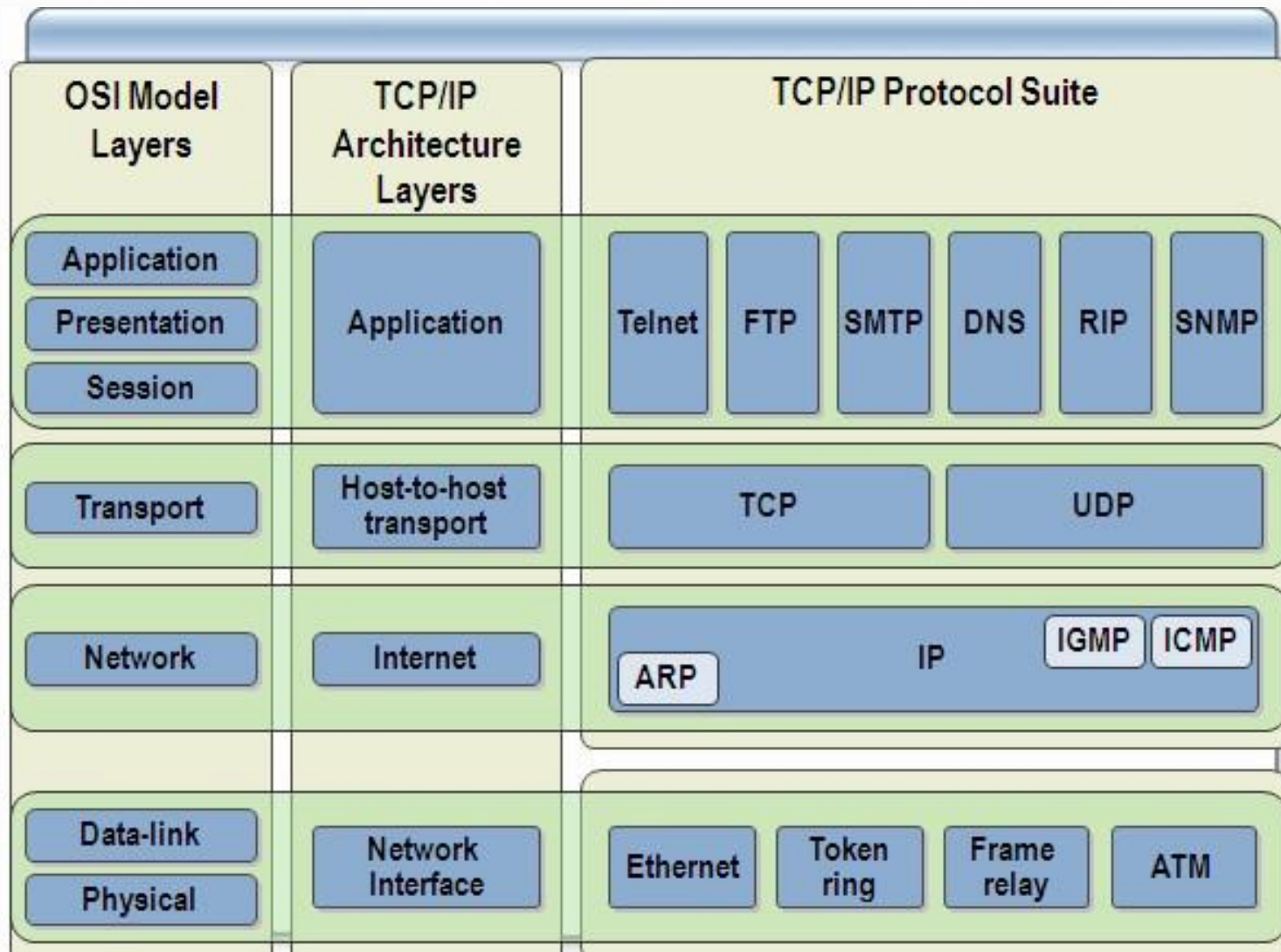


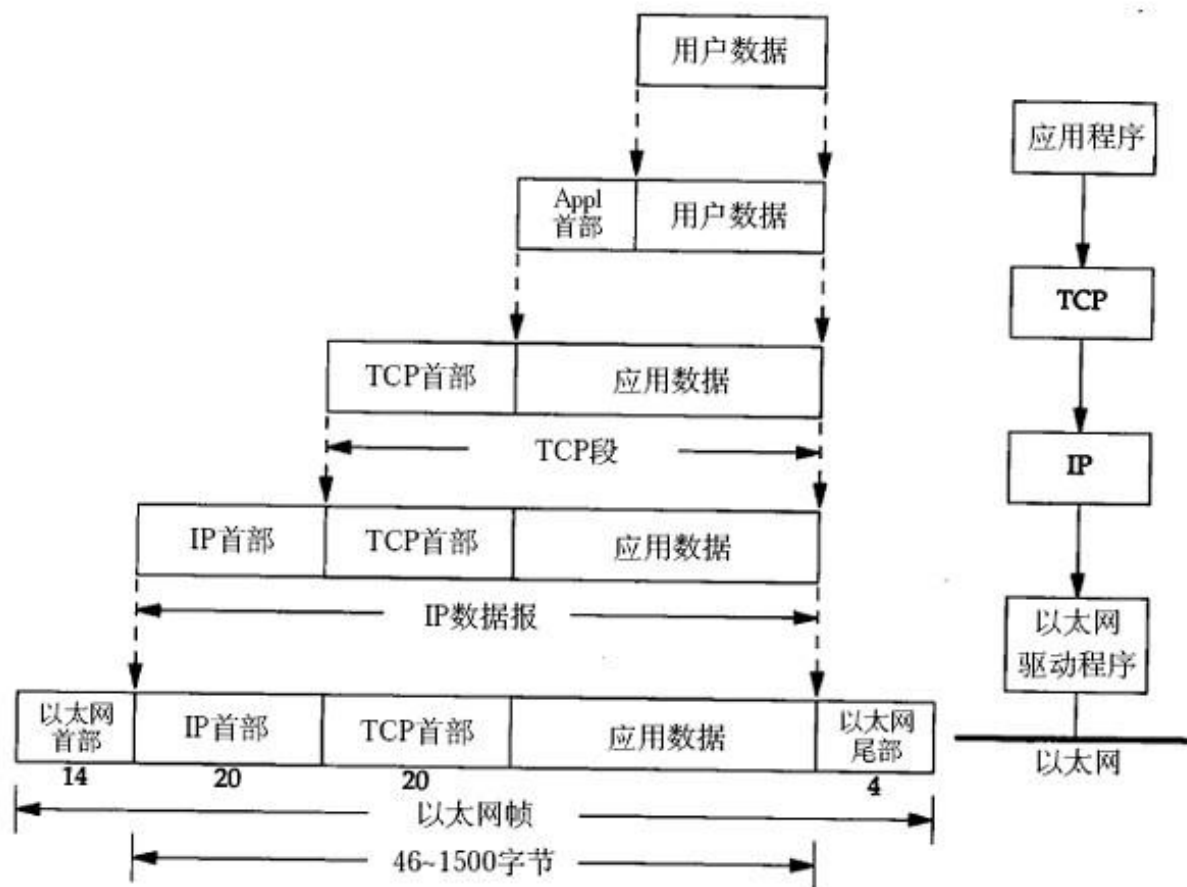
伴随我的历程一起
开始网络开发之旅吧！

基于TCP/IP 协议的分析

- 在公司不敢上无关网站，无聊！
- 在公司不敢下电影，浪费大好带宽！
- 在公司甚至都不敢发某些邮件，郁闷了吧！

都是他惹的祸





正是TCP/IP协议栈这种良好的分层设计为我们进行协议分析提供了极大的便利，我们该如何动手分析呢？

站在巨人的肩膀：[pcap](#)

In the field of computer network administration, **pcap** (packet capture) consists of an application programming interface (API) for capturing network traffic. Unix-like systems implement pcap in the **libpcap** library; Windows uses a port of libpcap known as **WinPcap**.

利用**pcap**库来对网络数据进行剥茧抽丝！

还有很多巨人在供我们踩

[Snort](#), [tcpdump](#), [wireshark](#).....

snort代码架构清晰巧妙；

tcpdump代码短小精悍；

wireshark对各协议分析最全面；



模块化设计，插件机制

这么简单？

难点：数据过滤，数据存储

- 防御系统：实时监测，匹配算法？
- 审计系统：大数据，存储挖掘？

期待高手分享这方面的话题。

推荐书籍：《TCP/IP详解 卷1：协议》

小知识：我们的网络数据是否真的安全？

MSN明文，有些邮箱明文。（<http://yaocoder.blog.51cto.com/2668309/483499>）

互动话题



木马的网络行为？（期待情人郭给大家开源内裤）

- 反向链接；
- 端口复用；
- 无端口技术；

TCP Socket编程

socket编程的基础知识。（各种书籍和资料）

简单调用几个API，理解下三次握手，bind，listen...?

少量并发  大量并发  海量并发

数量级的增加，处理难度增加

C10K问题：网络服务在处理数以万计的客户端连接时，往往出现效率低下甚至完全瘫痪。

复杂的网络环境也会给我们带来挑战

服务端网络模型

先清楚几个概念：阻塞I/O，非阻塞I/O，I/O复用，异步I/O

（参考UNP1 第六章）



同步

阻塞，非阻塞：进程/线程要访问的数据是否就绪，进程/线程是否需要等待；

同步，异步：访问数据的方式，同步需主动读写数据，还是会阻塞；异步只需等待I/O操作完成的通知，并不主动读写数据，由系统内核完成；

all connections one thread+blocking I/O 多并发情况找死

per connection per thread+blocking I/O 线程资源有限 (Go routine, Erlang actor)

thread pool+blocking I/O cpu有限, 线程切换消耗

non-blocking I/O+I/O multiplexing 单线程但是高性能, 继续优化?

non-blocking I/O+I/O multiplexing+async I/O 很高级, windows完成端口, linux支持好吗?

两种高性能I/O设计模式: **Reactor, Proactor**

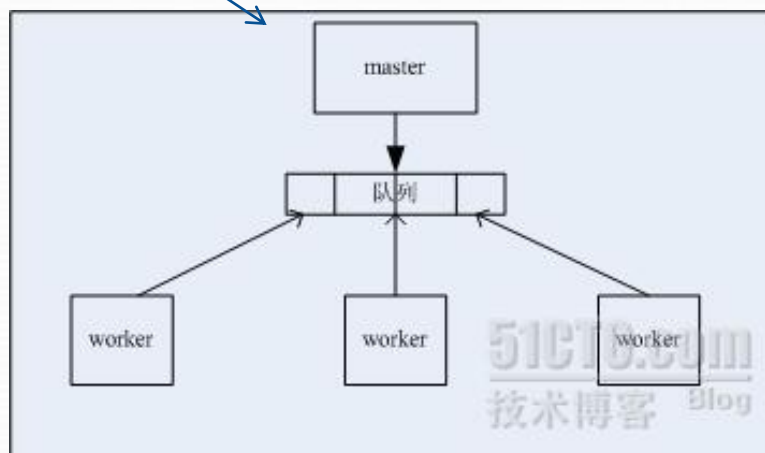
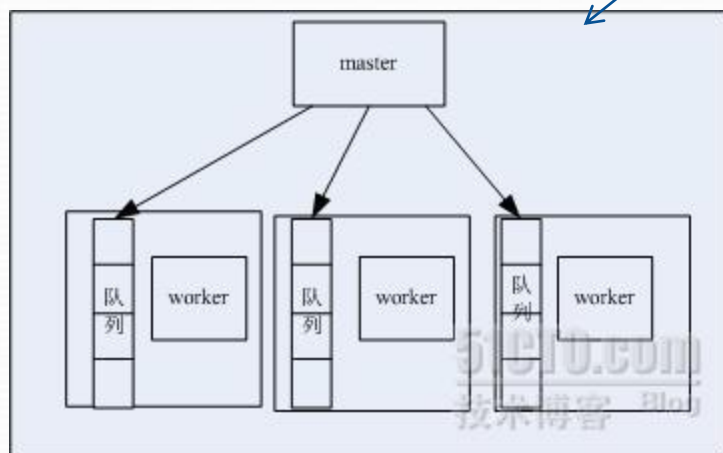
non-blocking I/O+I/O multiplexing

Linux下最成熟的模型，select、epoll、kqueue。

select和epoll的区别？

单线程→多线程（one loop per thread）；

master——worker模型（memcached，nginx）；



第三方网络库

- ACE: 学之者生, 用之者死 (陈硕);
- boost asio: has a “near STL” status (stackoverflow);
- Poco: 全面;
- Libev: 速度更快, bug更少, 特性更多, 体积更小;
- Libevent: 简单, 强大;

参考:

<http://stackoverflow.com/questions/992069/ace-vs-boost-vs-poco>

<http://stackoverflow.com/questions/9433864/whats-the-difference-between-libev-and-libevent>

网络协议选则

- 1.网络数据大小——占用带宽，传输效率；
- 2.网络数据安全性——敏感数据的网络安全；
- 3.编码复杂度——序列化和反序列化复杂度，效率，数据结构的可扩展性，可维护性；
- 4.协议通用性——大众规范；

- 自定义二进制：TLV
- 提供序列化和反序列化库的开源协议：protocol buffers, json, thrift
- 文本化协议：xml, json
(《unix编程艺术》，第5章--文本化，好协议产生好实践)

实践中你常常会遇到：

关键词：MTU, SO_LINGER, TCPNODELAY, TIMEWAIT, keepalive, 串话...

辅助工具：python, netcat, tcpdump, wireshark

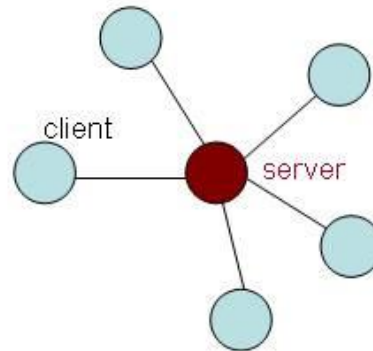
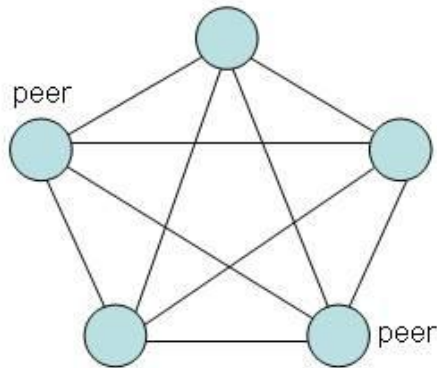
推荐书籍：《UNIX网络编程卷1》
《Linux多线程服务端编程》

互动话题：《UNIX网络编程卷1》常用的简写是什么？
《UNIX环境高级编程》呢？

P2P相关话题

P2P特点:

- 系统中节点的能力和作用是平等的;
- 系统中的通信是对等的, 节点同时扮演客户端和服务端两种角色;



参考:

<http://zh.wikipedia.org/wiki/%E5%B0%8D%E7%AD%89%E7%B6%B2%E8%B7%AF>

重点不是他们..., 是NAT穿透技术和流媒体传输技术

NAT穿透（俗名打洞）：

NAT积极作用：解决IP地址短缺；隐藏网络，安全。

NAT消极作用：破坏了端到端的网络通信，使网络结构复杂。

公司交换机， 家庭路由器...都是NAT设备

P2P穿越NAT的几种方案：

1.反向链接技术

2.UDP打洞技术

3.TCP打洞技术

NAT有哪几种类型？

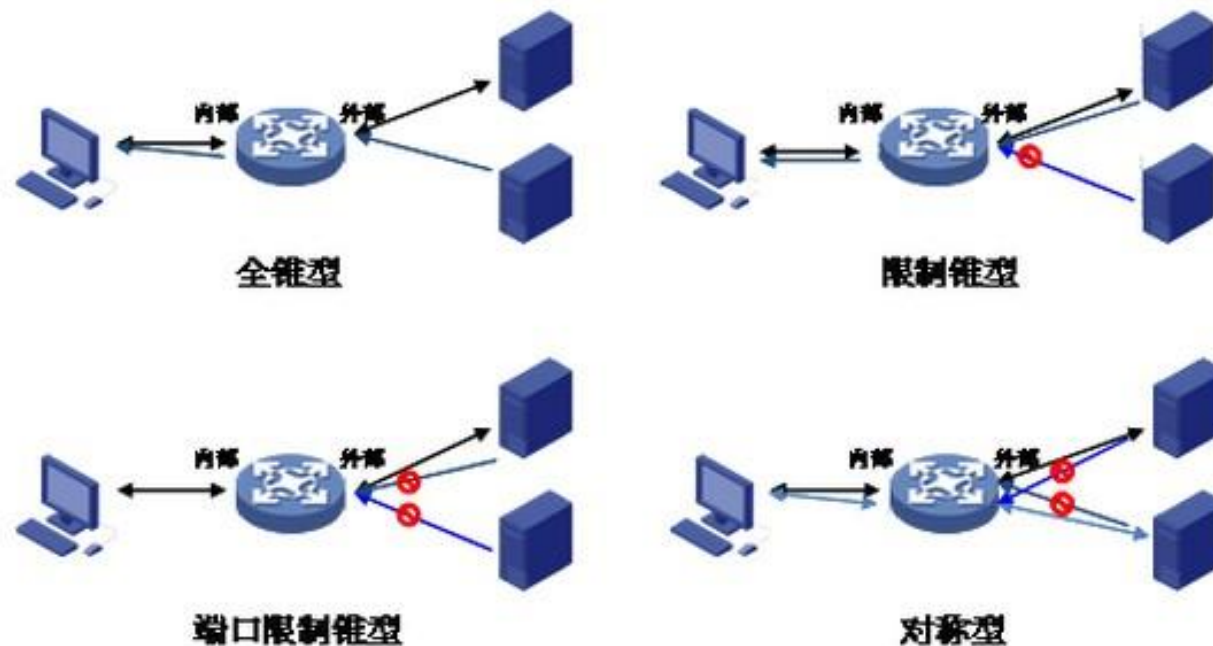
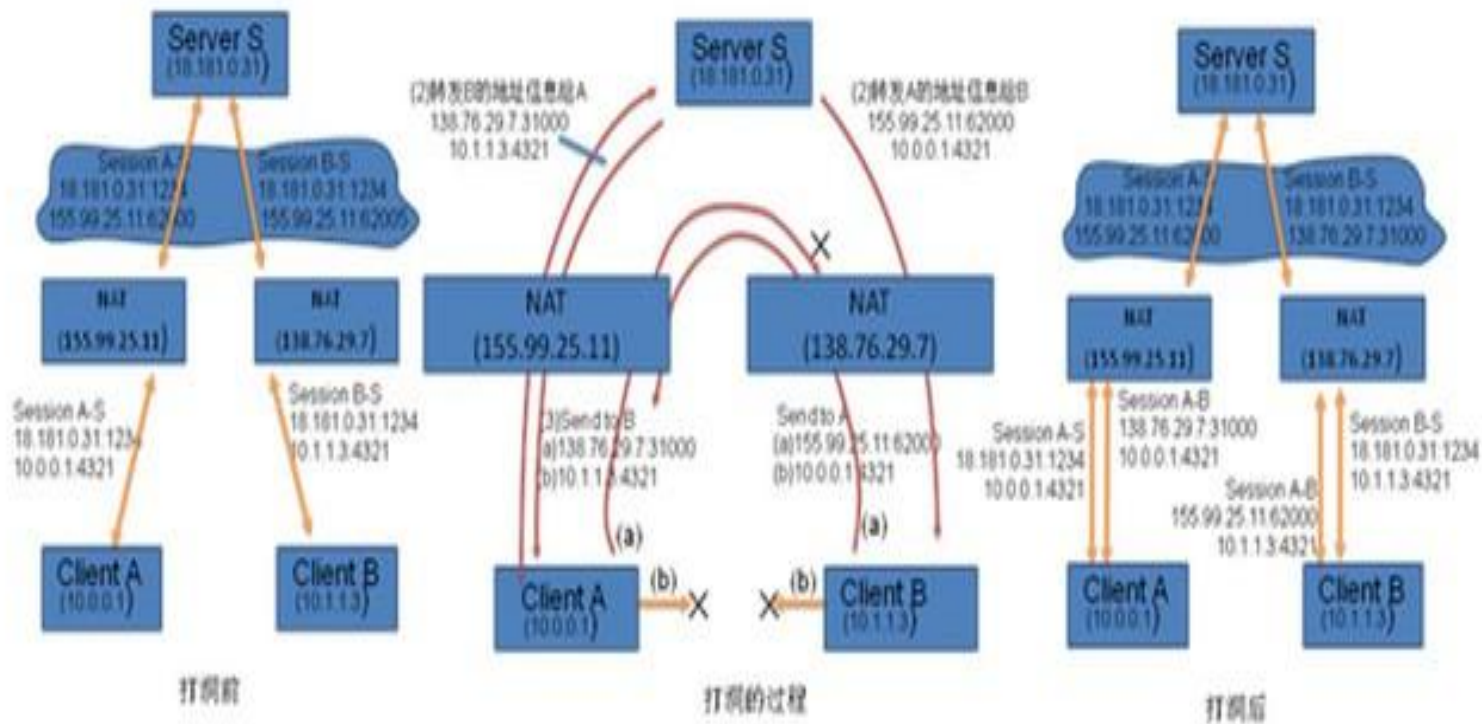


图3 按照端口转换映射方式分类

<http://zh.wikipedia.org/zh/%E7%BD%91%E7%BB%9C%E5%9C%B0%E5%9D%80%E8%BD%AC%E6%8D%A2>

如何进行穿透？



并不是所有情况都能穿透:

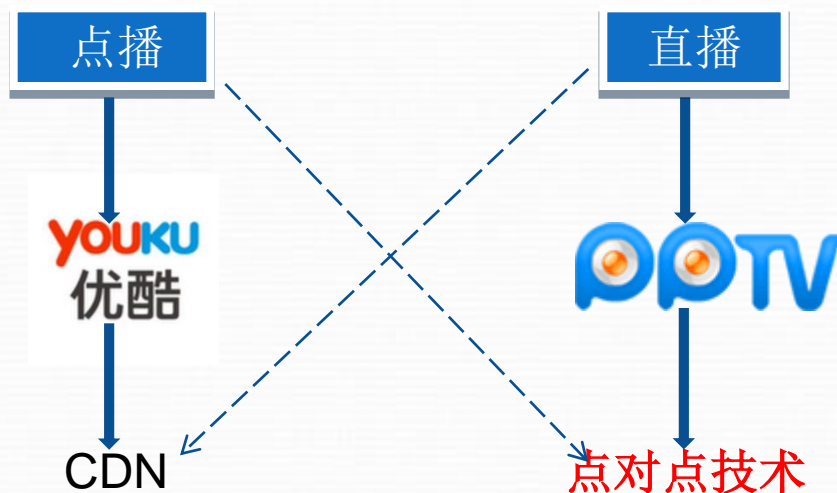
- 端口限制型——对成型
- 对称型——对称型

<http://www.goto.info.waseda.ac.jp/~wei/file/wei-apan-v10.pdf>

并不是所有情况都需要自己代码实现穿透:

- 应用层网关技术 (ALG)
- 探针技术 (STUN和TURN)
- 中间件技术 (UPNP)
- 中继代理技术
- 特定协议的自穿越技术 (IKE和Ipsec)

流媒体

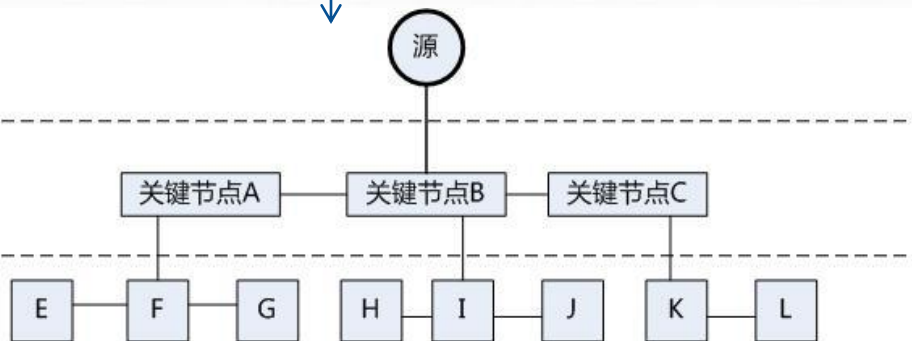
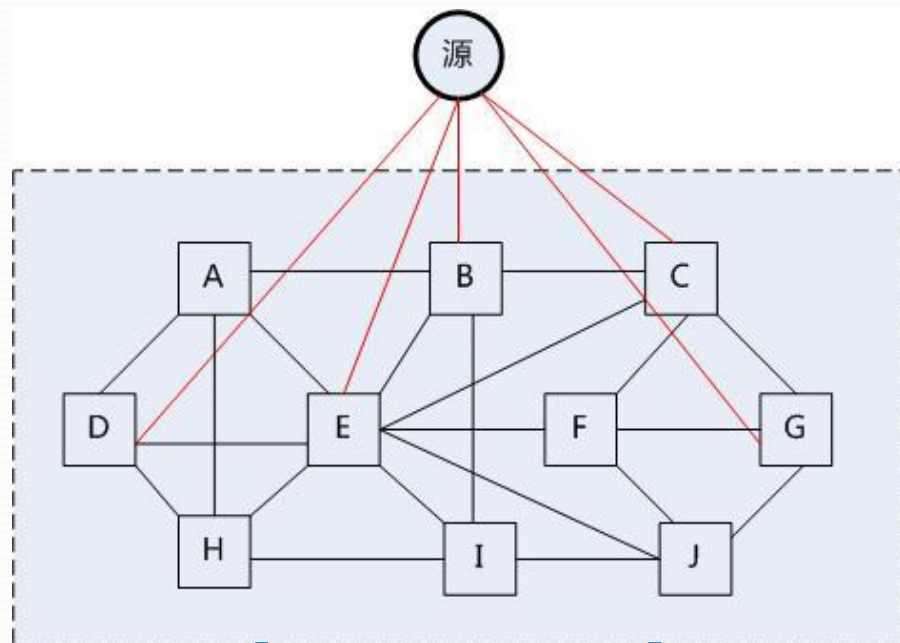


对等网络媒体直播的挑战

- 可扩展性：节点规模的增大不影响性能；
- 带宽利用率和吞吐率：如何有效的利用上行和下行带宽；
- 丢包和拥塞：如何保证服务质量（Quality of Service）；
- 延迟：对等网络的高度动态性，无法保证延迟；
- 异构性：DSL, Wireless LAN, NAT

P2P直播技术几种类型

单播树，多播树，网状



系统特点		组播树		随机拓扑 (网状)
		单棵树	多棵树	
网络特性	可扩展性	差	中	好
	网络稳定性	差	好	好
	负载均衡	差	好	好
数据特性	额外通信开销	小	中	大
	多源下载	单源	多源	多源
	传输速率	中	高	高
	视频延迟	好	中	差
应用型	实现	简单	复杂	简单
	应用场景	小规模	大规模	大规模
	管理复杂度	低	高	中



推荐书籍：《对等网络:结构、应用与设计》

其他：论文资料，源码资料

互动话题：大家熟知的迅雷的是怎样的应用？

服务端开发技术选型

- 系统平台的选择：CentOS, SUSE, Ubuntu...
- 开发语言的选择：C++, go...
- 数据库的使用、优化、部署：MySQL
- 缓存的使用：memcached, redis
- 监控系统：日志，负载...

稳定性，高性能，可扩展性



谢谢！