计算机网络

目 次

[1 计算机网络和因特网 1](#_Toc59146481)

计算机网络和因特网

* NFC 早期网络控制协议
* ISP(Internet Service Provider) 因特网服务提供商
* DSL (Digital Subscriber Line)数字用户线 使用双绞铜线
* DSLAM 数字用户线接入复用器
* 电缆因特网接入 常被称为HFC(Hybrid Fiber Coax)混合光纤同轴系统
* CMTS(Cable Modem Termination System) 电缆调制解调器端接系统
* FTTH(Fiber To The Home) 光纤到户
* AON(Active Optical Network) 主动光纤网络
* PON(Passive Optical Network) 被动光纤网络
* ONT(Optical Network Terminator) 光纤网络端接器 家庭
* OLT(Optical Line Terminator) 光纤线路端接器 本地电话和公司的中心局
* LAN(Local Area Network) 局域网
* WiFi 基于IEEE802.11技术的无线局域网（无线LAN）
* LTE（Long-Term Evolution）来源于3G的一种无线通信技术，又名长期演进
* 物理媒体分类：引导型媒体(guided media，电波延固体媒体前行)和非引导型媒体(unguided media，电波在空气或外层空间传播)
* UTEP(Unshielded Twisted Pair) 无屏蔽双绞铜线 常用于局域网
* 报文：在各种网络应用中，端系统彼此交换报文
* 分组(packet)：为了从源端系统向目的端系统发送一个报文，源将长报文划分为较小的数据块，称之为分组传送
* 分组交换机：在源和目的地之间，每个分组都通过通信链路和分组交换机(packet switch)传送，交换机主要有两类：路由(router)和链路层交换机(link-layer switch)
* 储存转发传输(store-and-forward transmission)机制：多数分组交换机在链路的输入端使用该机制，是指在交换机能够开始向输出链路传输该分组的第一个比特之前，必须接收到整个分组。此过程形成储存转发时延
* 分组交换机输出缓存(output buffer)：分组交换机具有一个输出缓存，也称输出队列(output queue)，用于储存路由器准备发往某条链路的分组。如果到达的分组需要传输到某条链路，但发现该链路正忙于传输其他分组，该到达分组必须在输出缓存中等待，此过程形成排队时延(queuing delay)。
* 分组丢失（丢包） (packet loss)：输出缓存已经被其他等待传输的分组完全充满了，到达的分组或者已经排队的分组之一将被丢弃。
* 转发表（forwarding table）和路由选择协议（routing protocal）：源主机在该分组的 首部包含了目的地的IP地址，每台路由器具有一个转发表，用于将目的地址（或目的地址的一部分）映射成为输出链路。路由选择协议用于自动设置这些转发表。
* 电路交换：传统的电话网络是典型例子。需要在发送方和接受方建立一条名副其实的链接，该链接成为一条电路。
* 链接中的电路FDM 频分复用实现：用频段划分链路，如，电话4KHZ，调频无线电台88MHZ~108MHZ
* 链接中的电路TDM 时分复用实现：通过帧和时隙划分链路
* PoP 存在点
* IXP 因特网交换点 多个ISP在这里一起对等，全球大概400多个IXP
* WWW World Wide Web 万维网
* HTTP协议：超文本传输协议，Web文档的请求和传送 应用层
* SMTP协议：电子邮件报文传输 应用层
* FTP协议:两个端系统之间的文件传送
* DNS：域名系统，DNS协议运行在UDP上，属于应用层，使用53号端口
* BIND软件：运行在DNS服务器
* 报文：位于应用层的信息分组
* TCP(Transmisson Control Protocol)运输层协议：面相连接的服务
* UDP运输层协议：无连接服务
* 报文段：运输层的分组
* 数据报：网络层分组
* IP(Internet Protocol)网络层协议：定义数据报中的各个字段以及如何运用这些字段
* IP层：即网络层
* 帧：链路层分组
* OSI开放系统互联模型模型：具有7层的网络协议栈，自顶向下：应用层、表示层、会话层、运输层、网络层、链路层、物理层
* DOS:拒绝服务攻击 攻击服务器
* SSL：安全套接字层 加强版TCP 增加安全性服务
* RTT：往返时间，一个短分组从客户到服务器，再返回客户所花费的时间
* CND：内容分发网络 安装了许多地理上分散的Web缓存器。
* SMTP：简单邮件传输协议
* POP3: 第三版的邮局协议
* IMAP：因特网邮件访问协议
* MIME：多用途互联网邮件扩展 新的格式规范编码方式 使得图像、声音、动画更方便得通过电子邮件传递

番外----流媒体传输协议

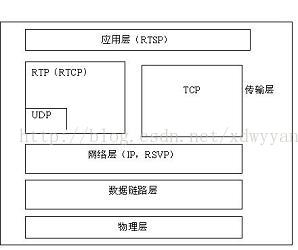
HTTP渐进式下载

基于TCP，仅能传输完整的音视频文件，在给定时刻，用户只能观看已下载的那部分，而不能跳到还未下载的部分。HTTP边下载边播放，严格意义上讲，不是直播协议。他的原理是先下载文件的基本信息，音频视频的时间戳，再下载音视频数据，以播放mp4为例，先下载文件头，根据文件头指引下载文件尾，然后再下载文件的音视频数据。

实时流媒体协议

可用于实况直播，也可传输完整的音视频文件。例如RTSP/RTP、RTMP、HLS、HTTP-FLV。

RTSP/RTP



1. RTSP

既可以基于UDP传输也可以基于TCP传输，是纯粹的传输控制协议，它本身与它负载的媒体数据不相关，RTSP协议需要自定义客户端向服务器发送RTSP命令。其视频数据由RTP传输，视频质量由RTCP控制，视频控制（如播放、暂停等）由RTSP提供。

1. RTP

是用于Internet上针对多媒体数据流的一种传输层协议.RTP 协议和 RTP 控制协议 RTCP 一起使用，而且它是建立在 UDP 协议上的.  
RTP 不像http和ftp可完整的下载整个影视文件，它是以固定的数据率在网络上发送数据，客户端也是按照这种速度观看影视文件，当影视画面播放过后，就不可以再重复播放，除非重新向服务器端要求数据。

RTMP

直播流协议的一种。

协议比较全能，既可以用来推送，又可以用来直播。其核心理念是将大块的视频帧和音频帧“剁碎”，然后以小数据包的形式在互联网上进行传输，且支持加密，因此隐私性相对比较理想，但拆包组包的过程比较复杂，所以在海量并发时容易出现一些不可预期的稳定性问题。

HLS

直播流协议的一种。

苹果推出的解决方案，将视频分成 5-10 秒的视频小分片，然后用 M3U8 索引表进行管理。由于客户端下载到的视频都是 5-10 秒的完整数据，故视频的流畅性很好，但也同样引入了很大的延迟（HLS 的一般延迟在 10-30s 左右）。相比于 FLV，HLS 在iPhone 和大部分 Android 手机浏览器上的支持非常给力，所以常用于 QQ 和微信朋友圈的 URL 分享。

HTTP-FLV

直播流协议的一种。

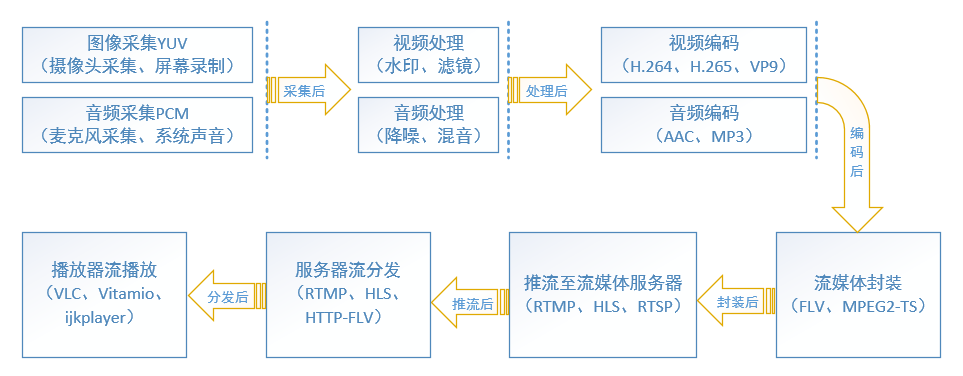
由 Adobe 公司主推，格式极其简单，只是在大块的视频帧和音视频头部加入一些标记头信息，由于这种极致的简洁，在延迟表现和大规模并发方面都很成熟。唯一的不足就是在手机浏览器上的支持非常有限，但是用作手机端 APP 直播协议却异常合适。

RTMP、HLS、HTTP-FLV 协议对比



想要实现直播，需要经历怎样的过程？

如果用一句话描述整体过程，其实就是直播时，主播端将直播流推向服务器，用户端发起请求从服务器拉视频流过来解码播放，流程如下图所示：



**前端需要实现的，如何在移动端的内嵌h5页面中实现直播流的播放。所以我们只需要关注后端是通过什么协议给我们返回直播流以及我们如何有效的播放就可以了。**

客户端直播插件

 除了采用h5原生的<video></<video>标签，我们还能用什么插件实现视频直播：**hls.js、video.js、vue-video-player**

### hls.js

hls.js是一个可以实现HTTP实时流媒体客户端的js库，主要依赖于<video></<video>标签和Media Source ExtensionsAPI。它的工作原理是将MPEG2传输流和AAC/MP3流转换成ISO BMFF (MP4)片段。由于hls.js是基于标准的<video></<video>标签实现播放，所以它不需要额外的播放器。

优点： 包比较小，很纯净，UI可以根据自己的业务自行扩展，功能可以根据需求进行封装，比较灵活，而且专业直播HLS协议流。

缺点： 对于常规的通用性播放器没有封装好的UI，功能上也需要自己调API实现，协议单一，只支持HLS。

### video.js

video.js是一个基于h5的网络视频播放器，支持h5视频、现代流媒体格式（MP4、WebM、HLS、DASH等）以及YouTube、Vimeo，甚至连flash也支持(通过插件实现，后面会详细说明)，可在桌面端或移动端实现视频播放。

优点： 支持多种格式的流媒体播放，浏览器不支持时可实现优雅降级；专门有一套针对直播流的UI；插件机制强大，目前社区已有数百个皮肤和插件供下载；兼容性好，几乎支持所有桌面及移动端的浏览器。

缺点： 包比较大，实现hls直播的时候其实是内嵌了hls.js的代码，由于封装好UI和功能，使其不够灵活，修改UI时需要通过插件实现。

### vue-video-player

vue-video-player其实就是将video.js集成到了Vue中，在Vue项目中使用起来会更方便。

安装引用以及效果参考：

https://www.cnblogs.com/dreamsqin/p/12557070.html