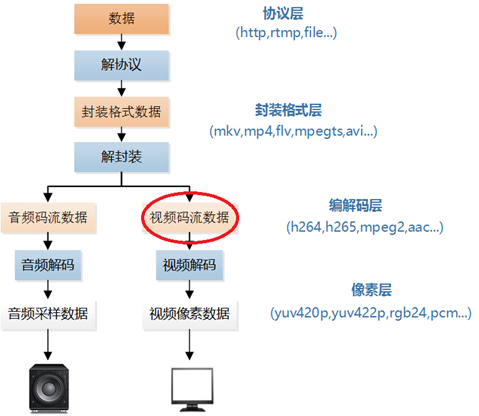
## 几个概念的联系

The video stream is compressed using a [video coding format](https://en.wikipedia.org/wiki/Video_coding_format) to make the file size smaller. Video coding formats include [H.264](https://en.wikipedia.org/wiki/H.264), [HEVC,](https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding) [VP8](https://en.wikipedia.org/wiki/VP8) or [VP9](https://en.wikipedia.org/wiki/VP9). Encoded audio and video streams are assembled in a container "[bitstream](https://en.wikipedia.org/wiki/Bitstream)" such as [MP4](https://en.wikipedia.org/wiki/MPEG-4), [FLV](https://en.wikipedia.org/wiki/Flash_Video), [WebM](https://en.wikipedia.org/wiki/WebM), [ASF](https://en.wikipedia.org/wiki/Advanced_Systems_Format) or [ISMA](https://en.wikipedia.org/wiki/Internet_Streaming_Media_Alliance). The bitstream is delivered from a streaming server to a streaming client (e.g., the computer user with their [Internet](https://en.wikipedia.org/wiki/Internet)-connected [laptop](https://en.wikipedia.org/wiki/Laptop)) using a transport protocol, such as Adobe's [RTMP](https://en.wikipedia.org/wiki/Real_Time_Messaging_Protocol) or [RTP](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol). The streaming client (the end user) may interact with the streaming server using a control protocol, such as [MMS](https://en.wikipedia.org/wiki/Microsoft_Media_Server) or [RTSP](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol).

<https://en.wikipedia.org/wiki/Streaming_media>

下图以一个流媒体播放器从收到数据到最后播放出流媒体的过程来说明各个概念的关系：



# 分类

## live streams

**Live streaming** refers to Internet content delivered in real-time, as events happen, much as [live television](https://en.wikipedia.org/wiki/Live_television) [broadcasts](https://en.wikipedia.org/wiki/Broadcast_television) its contents over the airwaves via a [television signal](https://en.wikipedia.org/wiki/Radio_wave). Live streaming does not need to be recorded at the origination point, although it frequently is.

## on-demand streams

**Video on demand (display)** (**VOD**) are systems which allow users to select and watch/listen to [video](https://en.wikipedia.org/wiki/Video) or [audio](https://en.wikipedia.org/wiki/Sound_recording_and_reproduction) content when they choose to, rather than having to watch at a specific broadcast time.

# Container

<https://en.wikipedia.org/wiki/Digital_container_format>

<https://en.wikipedia.org/wiki/Video_file_format>

<https://en.wikipedia.org/wiki/Audio_file_format>

各种video文件格式的比较：

<https://en.wikipedia.org/wiki/Comparison_of_video_container_formats>

A **container** or **wrapper format** is a [metafile](https://en.wikipedia.org/wiki/Metafile) [format](https://en.wikipedia.org/wiki/File_format) whose specification describes how different elements of data and [metadata](https://en.wikipedia.org/wiki/Metadata) coexist in a [computer file](https://en.wikipedia.org/wiki/Computer_file).

Since the container does not describe how data or metadata is encoded, a [program](https://en.wikipedia.org/wiki/Computer_program) able to identify and open a container file might not be able to decode the contained data. This may be caused by the program lacking the required [decoding algorithm](https://en.wikipedia.org/wiki/Codec).

## FLV

**Flash Video** is a [container file format](https://en.wikipedia.org/wiki/Container_format_(digital)) used to deliver [digital video](https://en.wikipedia.org/wiki/Digital_video) content (e.g., [TV shows](https://en.wikipedia.org/wiki/TV_show), [movies](https://en.wikipedia.org/wiki/Movie), etc.) over the[Internet](https://en.wikipedia.org/wiki/Internet) using [Adobe Flash Player](https://en.wikipedia.org/wiki/Adobe_Flash_Player) version 6 and newer.

FLV was originally developed by [Macromedia](https://en.wikipedia.org/wiki/Macromedia). In the early 2000s, Flash Video used to be the *de facto*standard for web-based streaming video (over [RTMP](https://en.wikipedia.org/wiki/Real_Time_Messaging_Protocol)). Notable users of it include [Hulu](https://en.wikipedia.org/wiki/Hulu), [VEVO](https://en.wikipedia.org/wiki/VEVO), [Yahoo! Video](https://en.wikipedia.org/wiki/Yahoo!_Video), [metacafe](https://en.wikipedia.org/wiki/Metacafe),[Reuters.com](https://en.wikipedia.org/wiki/Reuters), and many other news providers.

The most recent public releases of Flash Player (collaboration between [Adobe Systems](https://en.wikipedia.org/wiki/Adobe_Systems) and [MainConcept](https://en.wikipedia.org/wiki/MainConcept)) also support [H.264](https://en.wikipedia.org/wiki/H.264) video and [HE-AAC](https://en.wikipedia.org/wiki/HE-AAC) audio.

Flash Video is viewable on most [operating systems](https://en.wikipedia.org/wiki/Operating_systems) via the Adobe Flash Player and [web browser](https://en.wikipedia.org/wiki/Web_browser) [plugin](https://en.wikipedia.org/wiki/Plug-in_(computing)) or one of several third-party programs. Apple's [iOS](https://en.wikipedia.org/wiki/IOS_(Apple)) devices, along with almost all other mobile devices, do not support the Flash Player plugin and so require other delivery methods such as provided by the [Adobe Flash Media Server](https://en.wikipedia.org/wiki/Adobe_Flash_Media_Server).

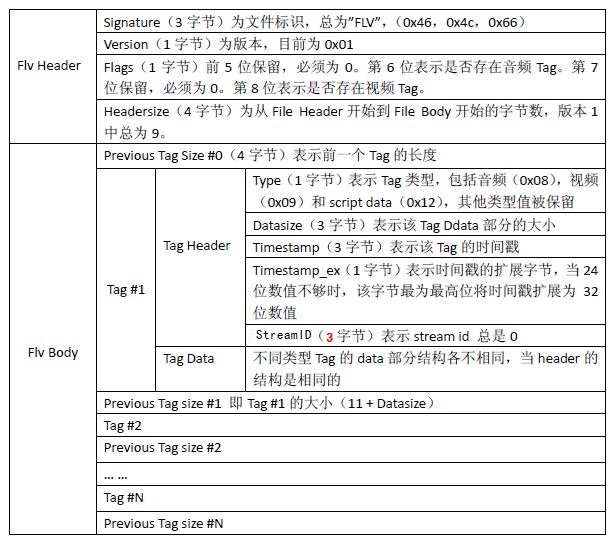
FLV files start with a standard header. After the header, the file is split into packets called "FLV tags", which have 15-byte packet headers. The first four bytes denote the size of the previous packet/tag (including the header), and aid in seeking backward.（详细见wiki的FLV词条）

Flash Video files can be delivered in several different ways:

* As a standalone .FLV file.
* Embedded in an SWF file using the Flash authoring tool.
* [Progressive download](https://en.wikipedia.org/wiki/Progressive_download) via [HTTP](https://en.wikipedia.org/wiki/HTTP).
* Streamed via [RTMP](https://en.wikipedia.org/wiki/Real_Time_Messaging_Protocol) to the Flash Player
* Delivering the content in an [MPEG-2](https://en.wikipedia.org/wiki/MPEG-2) stream using the [HTTP Live Streaming](https://en.wikipedia.org/wiki/HTTP_Live_Streaming) format.

详细可以见：<http://blog.csdn.net/leixiaohua1020/article/details/17934487>

FLV文件由文件头和若干个tag组成：



如果为音频tag：



如果为视频tag：



Script Tag，该类型Tag又通常被称为Metadata Tag，会放一些关于FLV视频和音频的元数据信息如：duration、width、height等。通常该类型Tag会跟在File Header后面作为第一个Tag出现，而且只有一个。



## MP4

**MPEG-4 Part 14** or **MP4** is a [digital](https://en.wikipedia.org/wiki/Digital_data) [multimedia](https://en.wikipedia.org/wiki/Multimedia) [container format](https://en.wikipedia.org/wiki/Digital_container_format) most commonly used to store [video](https://en.wikipedia.org/wiki/Digital_video) and [audio](https://en.wikipedia.org/wiki/Digital_audio). Like most modern [container formats](https://en.wikipedia.org/wiki/Container_format_(digital)), it allows [streaming](https://en.wikipedia.org/wiki/Streaming_media) over the [Internet](https://en.wikipedia.org/wiki/Internet). The only official [filename extension](https://en.wikipedia.org/wiki/Filename_extension) for MPEG-4 Part 14 files is **.mp4**, but many have other extensions, most commonly **.m4a** and **.m4p**. M4A (audio only) is often compressed using [AAC](https://en.wikipedia.org/wiki/Advanced_Audio_Coding) encoding ([lossy](https://en.wikipedia.org/wiki/Lossy)), but can also be in [Apple Lossless](https://en.wikipedia.org/wiki/Apple_Lossless) format. M4P is a protected format which employs [DRM](https://en.wikipedia.org/wiki/Digital_rights_management) technology to restrict copying.

In MP4 file, the widely supported [codecs](https://en.wikipedia.org/wiki/Codecs) and additional data streams are  [MPEG-4 Part 10 (H.264)](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) for video and [Advanced Audio Coding](https://en.wikipedia.org/wiki/Advanced_Audio_Coding) for audio.

## WebM

**WebM** is a [video file format](https://en.wikipedia.org/wiki/Video_file_format).[[3]](https://en.wikipedia.org/wiki/WebM#cite_note-3) It is primarily intended to offer a [royalty-free](https://en.wikipedia.org/wiki/Royalty-free) alternative to use in the [HTML5 video](https://en.wikipedia.org/wiki/HTML5_video)tag. It has a sister project [WebP](https://en.wikipedia.org/wiki/WebP) for images. The development of the format is sponsored by Google, and the corresponding software is distributed under a [BSD license](https://en.wikipedia.org/wiki/BSD_licenses).

WebM initially supported [VP8](https://en.wikipedia.org/wiki/VP8) video and [Vorbis](https://en.wikipedia.org/wiki/Vorbis) audio streams. In 2013 it was updated to accommodate [VP9](https://en.wikipedia.org/wiki/VP9) video and [Opus](https://en.wikipedia.org/wiki/Opus_(audio_format)) audio.

# 传输协议和控制协议

## RTMP

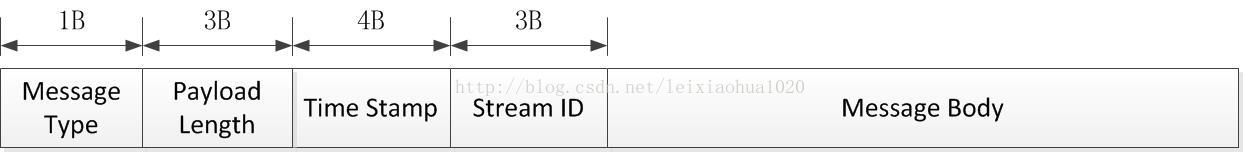
**Real-Time Messaging Protocol** (**RTMP**) was initially a [proprietary protocol](https://en.wikipedia.org/wiki/Proprietary_protocol) developed by [Macromedia](https://en.wikipedia.org/wiki/Macromedia) for [streaming](https://en.wikipedia.org/wiki/Streaming_media) audio, video and data over the Internet, between a [Flash](https://en.wikipedia.org/wiki/Adobe_Flash) player and a server. Macromedia is now owned by [Adobe](https://en.wikipedia.org/wiki/Adobe_Systems), which has released an incomplete version of the specification of the protocol for public use.

The RTMP protocol has multiple variations:

1. The "plain" protocol which works on top of and uses TCP port number 1935 by default.
2. **RTMPS**, which is RTMP over a [TLS/SSL](https://en.wikipedia.org/wiki/Transport_Layer_Security) connection.
3. **RTMPE**, which is RTMP encrypted using Adobe's own security mechanism. While the details of the implementation are proprietary, the mechanism uses industry standard cryptography primitives.[[1]](https://en.wikipedia.org/wiki/Real-Time_Messaging_Protocol#cite_note-RTMPE_overview-1)
4. **RTMPT**, which is [encapsulated](https://en.wikipedia.org/wiki/Encapsulation_(networking)) within [HTTP](https://en.wikipedia.org/wiki/HTTP) requests to traverse firewalls. RTMPT is frequently found utilizing cleartext requests on [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) [ports](https://en.wikipedia.org/wiki/Port_(computer_networking)) 80 and 443 to bypass most corporate traffic filtering. The encapsulated session may carry plain RTMP, RTMPS, or RTMPE packets within.
5. **RTMFP**, which is RTMP over UDP instead of TCP, replacing RTMP Chunk Stream. The Secure [Real-Time Media Flow Protocol](https://en.wikipedia.org/wiki/Real-Time_Media_Flow_Protocol) suite has been developed by Adobe Systems and enables end‐users to connect and communicate directly with each other (P2P).

While the primary motivation for RTMP was to be a protocol for playing Flash video, it is also used in some other applications, such as the [Adobe LiveCycle Data Services ES](https://en.wikipedia.org/wiki/Adobe_LiveCycle_Data_Services_ES).

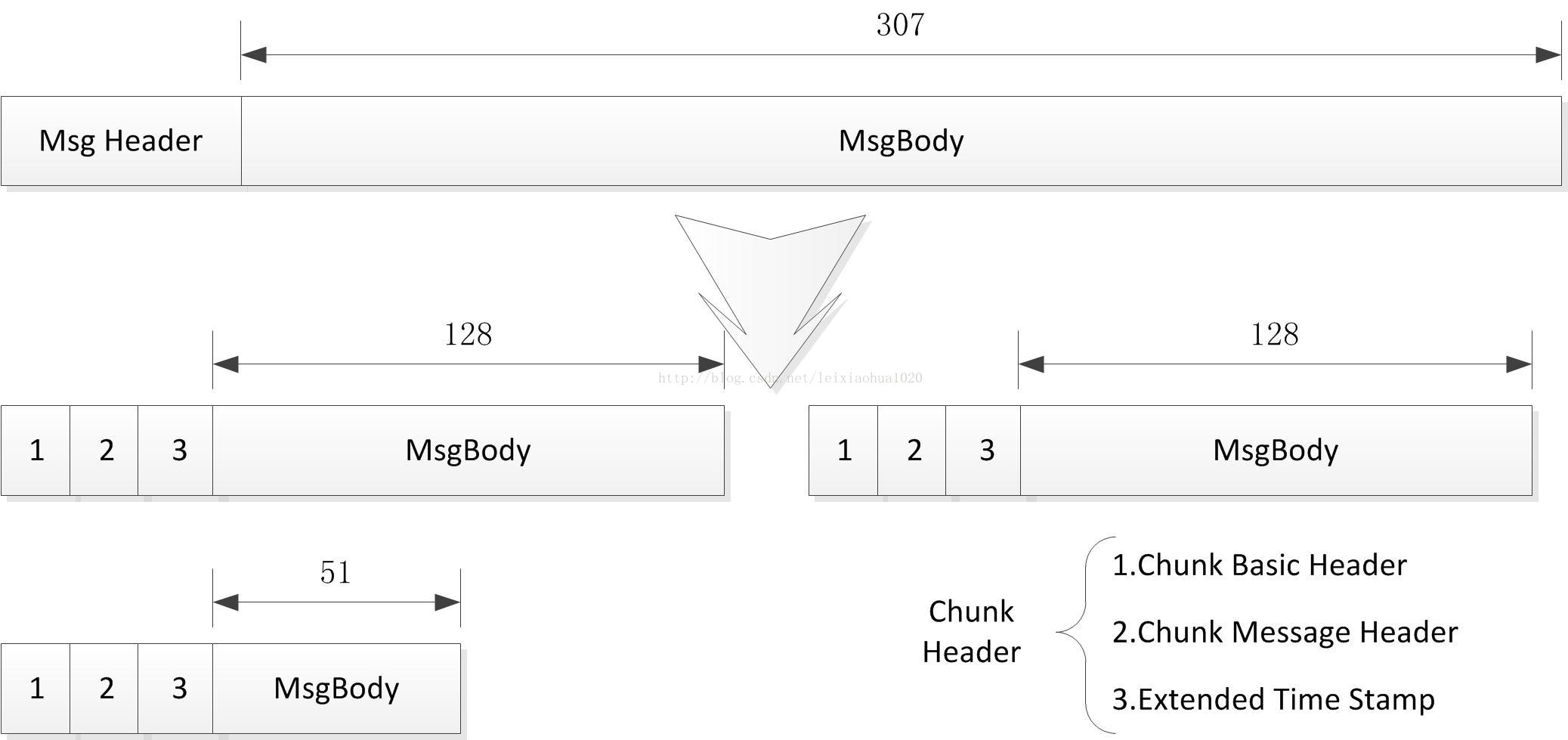
消息是RTMP协议中基本的数据单元。



Message Type ID在1-7的消息用于协议控制，这些消息一般是RTMP协议自身管理要使用的消息，用户一般情况下无需操作其中的数据。Message Type ID为8，9的消息分别用于传输音频和视频数据。Message Type ID为15-20的消息用于发送AMF编码的命令，负责用户与服务器之间的交互，比如播放，暂停等等。消息首部（Message Header）有四部分组成：标志消息类型的Message Type ID，标志消息长度的Payload Length，标识时间戳的Timestamp，标识消息所属媒体流的Stream ID。

在网络上传输数据时，消息需要被拆分成较小的数据块（Chunk）。分块使高层协议的大消息分割成小的消息，保证大的低优先级消息不阻塞小的高优先级消息。

在消息被分割成几个消息块的过程中，消息负载部分（Message Body）被分割成大小固定的数据块（默认是128字节，最后一个数据块可以小于该固定长度），并在其首部加上消息块首部（Chunk Header），就组成了相应的消息块。



RTMP传输媒体数据的过程中，发送端首先把媒体数据封装成消息，然后把消息分割成消息块，最后将分割后的消息块通过TCP协议发送出去。接收端在通过TCP协议收到数据后，首先把消息块重新组合成消息，然后通过对消息进行解封装处理就可以恢复出媒体数据。

## RTSP

准确的说，RTSP是一个控制协议，不是一个传输协议，RTP才是

The **Real Time Streaming Protocol** (**RTSP**) is a network control [protocol](https://en.wikipedia.org/wiki/Communications_protocol) designed for use in entertainment and communications systems to control [streaming media](https://en.wikipedia.org/wiki/Streaming_media)[servers](https://en.wikipedia.org/wiki/Web_server). The protocol is used for establishing and controlling media sessions between end points. Clients of media servers issue [VCR](https://en.wikipedia.org/wiki/VCR)-style commands, such as*play*, *record* and *pause*, to facilitate real-time control of the media streaming from the server to a client (Video On Demand) or from a client to the server (Voice Recording).

The transmission of streaming data itself is not a task of RTSP. Most RTSP servers use the [Real-time Transport Protocol](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol) (RTP) in conjunction with [Real-time Control Protocol](https://en.wikipedia.org/wiki/RTCP) (RTCP) for media stream delivery. However, some vendors implement proprietary transport protocols. The RTSP server software from [RealNetworks](https://en.wikipedia.org/wiki/RealNetworks), for example, also used RealNetworks' proprietary [Real Data Transport](https://en.wikipedia.org/wiki/Real_Data_Transport) (RDT).

RTSP was developed by RealNetworks, [Netscape](https://en.wikipedia.org/wiki/Netscape)[[1]](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#cite_note-Inc.1998-1) and [Columbia University](https://en.wikipedia.org/wiki/Columbia_University), with the first draft submitted to IETF in 1996.[[2]](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#cite_note-Osso1999-2) It was standardized by the Multiparty Multimedia Session Control Working Group (MMUSIC WG) of the [Internet Engineering Task Force](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force) (IETF) and published as [RFC 2326](https://tools.ietf.org/html/rfc2326) in 1998.[[3]](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#cite_note-:0-3) RTSP 2.0 is currently under development as a replacement of RTSP 1.0. RTSP 2.0 is based on RTSP 1.0 but is not backwards compatible other than in the basic version negotiation mechanism.[[4]](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#cite_note-4)

RTSP using RTP and RTCP allows for the implementation of rate adaptation.

While similar in some ways to [HTTP](https://en.wikipedia.org/wiki/HTTP), RTSP defines control sequences useful in controlling multimedia playback. While HTTP is [stateless](https://en.wikipedia.org/wiki/Stateless_server), RTSP has state; an identifier is used when needed to track concurrent sessions. Like HTTP, RTSP uses TCP to maintain an end-to-end connection and, while most RTSP control messages are sent by the client to the server, some commands travel in the other direction (i.e. from server to client).

Presented here are the basic RTSP requests. Some typical [HTTP requests](https://en.wikipedia.org/wiki/HTTP_request), like the OPTIONS request, are also available. The default transport layer [port number](https://en.wikipedia.org/wiki/Port_number) is 554[[3]](https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#cite_note-:0-3) for both [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) and [UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol), the latter being rarely used for the control requests.

## RTP和RTCP

The **Real-time Transport Protocol** (**RTP**) is a [network protocol](https://en.wikipedia.org/wiki/Network_protocol) for delivering audio and video over [IP networks](https://en.wikipedia.org/wiki/IP_networks). RTP is used extensively in communication and entertainment systems that involve [streaming media](https://en.wikipedia.org/wiki/Streaming_media), such as [telephony](https://en.wikipedia.org/wiki/Telephony), [video teleconference](https://en.wikipedia.org/wiki/Video_teleconference) applications, [television services](https://en.wikipedia.org/wiki/IPTV) and web-based [push-to-talk](https://en.wikipedia.org/wiki/Push-to-talk) features.

RTP typically runs over [User Datagram Protocol](https://en.wikipedia.org/wiki/User_Datagram_Protocol) (UDP). RTP is used in conjunction with the [RTP Control Protocol](https://en.wikipedia.org/wiki/RTP_Control_Protocol) (RTCP). While RTP carries the media streams (e.g., audio and video), RTCP is used to monitor transmission statistics and [quality of service](https://en.wikipedia.org/wiki/Quality_of_service) (QoS) and aids synchronization of multiple streams. RTP is one of the technical foundations of [Voice over IP](https://en.wikipedia.org/wiki/Voice_over_IP)and in this context is often used in conjunction with a [signaling protocol](https://en.wikipedia.org/wiki/Signaling_protocol) such as the [Session Initiation Protocol](https://en.wikipedia.org/wiki/Session_Initiation_Protocol) (SIP) which establishes connections across the network.

RTP provides facilities for [jitter](https://en.wikipedia.org/wiki/Jitter) compensation and detection of out of sequence arrival in data, which are common during transmissions on an IP network. RTP allows data transfer to multiple destinations through [IP multicast](https://en.wikipedia.org/wiki/IP_multicast).[[1]](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol#cite_note-Hardy_298-1) RTP is regarded as the primary standard for audio/video transport in IP networks and is used with an associated profile and payload format.

Real-time [multimedia](https://en.wikipedia.org/wiki/Multimedia) streaming applications require timely delivery of information and often can tolerate some packet loss to achieve this goal. For example, loss of a packet in audio application may result in loss of a fraction of a second of audio data, which can be made unnoticeable with suitable [error concealment](https://en.wikipedia.org/wiki/Error_concealment)algorithms.[[3]](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol#cite_note-Perkins_46-3) The [Transmission Control Protocol](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) (TCP), although standardized for RTP use,[[4]](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol#cite_note-4) is not normally used in RTP applications because TCP favors reliability over timeliness. Instead the majority of the RTP implementations are built on the [User Datagram Protocol](https://en.wikipedia.org/wiki/User_Datagram_Protocol) (UDP).

上面都是些什么鬼？三个是什么区别？网上这样说：

RTP: Real-time Transport Protocol,实时传输协议，一般用于多媒体数据的传输。

RTCP: RTP Control Protocol，实时传输控制协议，同RTP一起用于数据传输的监视，控制功能。

RTSP: Real Time Streaming Protocol,实时流协议，用于多媒体数据流的控制，如播放，暂停等。

RTP/RTCP相对于底层传输层，和RTSP，SIP等上层协议一起可以实现视频会议，视频直播等应用：rtsp发起/终结流媒体（通过sdp），rtp传输流媒体数据 ，rtcp对rtp进行控制，同步。

## HTTP Live Streaming(HLS)

HLS works by breaking the overall stream into a sequence of small HTTP-based file downloads, the client may select from a number of different alternate streams containing the same material encoded at a variety of data rates, allowing the streaming session to adapt to the available data rate. At the start of the streaming session, it downloads an [extended M3U playlist](https://en.wikipedia.org/wiki/Extended_M3U) containing the metadata for the various sub-streams which are available.

HTTP Live Streaming uses a conventional [web server](https://en.wikipedia.org/wiki/Web_server) to distribute audiovisual content and requires specific software to fit into the proper format transmission in real time.

Since its requests use only standard HTTP transactions, HTTP Live Streaming is capable of traversing any[firewall](https://en.wikipedia.org/wiki/Firewall_(computing)) or [proxy server](https://en.wikipedia.org/wiki/Proxy_server) that lets through standard HTTP traffic, unlike UDP-based protocols such as [RTP](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol). This also allows content to be offered from conventional HTTP servers as origin and delivered over widely available HTTP-based[content delivery networks](https://en.wikipedia.org/wiki/Content_delivery_network).

HLS also specifies a standard encryption mechanism[[4]](https://en.wikipedia.org/wiki/HTTP_Live_Streaming#cite_note-4) using [AES](https://en.wikipedia.org/wiki/Advanced_Encryption_Standard) and a method of secure key distribution using [HTTPS](https://en.wikipedia.org/wiki/HTTPS) with either a device specific realm login or [HTTP cookie](https://en.wikipedia.org/wiki/HTTP_cookie) which together provide a simple [DRM](https://en.wikipedia.org/wiki/Digital_Rights_Management) system.

To make the system scalable and adaptable to the bandwidth of the network, the video flow is coded in different qualities. Thus, depending on the bandwidth and transfer network speed, the video will play at different qualities. To implement this, the system must encode the video in different qualities and generate an index file that contains the locations of the different quality levels.

HLS streams can carry generic [ID3](https://en.wikipedia.org/wiki/ID3) data as a separate pid in the transport stream (see [[7]](https://en.wikipedia.org/wiki/HTTP_Live_Streaming#cite_note-7)). ID3 metadata is specified by Apple in separate audio streams for the purposes of synchronisation with video.

## HTTP-FLV

FLV是一种封装（container），而HTTP-FLV是一种传输协议。HTTP-FLV主要的特征是使用http传输协议来传输flv格式的音视频内容。HTTP协议中有个content-length字段，如果在响应包中有包含，客户端就会在接收这个长度的数据后就认为数据传输完成。

       如果服务器回复http包中没有这个字段，客户端就一直会接收数据，直到服务器跟客户端的socket连接断开,HTTP-FLV直播协议就是利用第二个原理。当然除了这个方法外还可以使用其他方式，如VLC中请求包中使用Range头域，可以一直不间断的请求不同范围内的音视频数据。

对比目前视频的几种下行协议，他们的延迟情况是：HTTP-FLV：内容延迟可以低于3秒；RTMP：内容延迟可以低于3秒；HLS:：延迟较高，一般在10秒左右。HTTP-FLV和RTMP一样都需要服务器有专门的流媒体服务器，而且播放器也要专门的flashplayer。因此在手机端一般需要专门的播放器。相比RTMP，因为它本身是一种http协议，所以对于网络防火墙的穿透性会更好。

# 压缩/编码格式

List of open-source codecs

<https://en.wikipedia.org/wiki/List_of_open-source_codecs>

A Large-Scale Comparison of x264, x265, and libvpx

<http://techblog.netflix.com/2016/08/a-large-scale-comparison-of-x264-x265.html>

List of codecs

<https://en.wikipedia.org/wiki/List_of_codecs>

## H264

The intent of the H.264/AVC project was to create a standard capable of providing good video quality at substantially lower bit rates than previous standards .  An additional goal was to provide enough flexibility to allow the standard to be applied to a wide variety of applications on a wide variety of networks and systems, including low and high bit rates, low and high resolution video, [broadcast](https://en.wikipedia.org/wiki/Broadcasting), [DVD](https://en.wikipedia.org/wiki/DVD) storage, [RTP](https://en.wikipedia.org/wiki/Real-time_Transport_Protocol)/[IP](https://en.wikipedia.org/wiki/Internet_Protocol) packet networks, and[ITU-T](https://en.wikipedia.org/wiki/ITU-T) multimedia [telephony](https://en.wikipedia.org/wiki/Telephony) systems.

H.264 is typically used for [lossy compression](https://en.wikipedia.org/wiki/Lossy_compression), although it is also possible to create truly [lossless-coded](https://en.wikipedia.org/wiki/Lossless_compression) regions within lossy-coded pictures or to support rare use cases for which the entire encoding is lossless.

H.264 is perhaps best known as being one of the video encoding standards for [Blu-ray Discs](https://en.wikipedia.org/wiki/Blu-ray_Disc); all Blu-ray Disc players must be able to decode H.264. It is also widely used by streaming internet sources, such as videos from [Vimeo](https://en.wikipedia.org/wiki/Vimeo), [YouTube](https://en.wikipedia.org/wiki/YouTube), and the [iTunes Store](https://en.wikipedia.org/wiki/ITunes_Store), web software such as the [Adobe Flash Player](https://en.wikipedia.org/wiki/Adobe_Flash_Player) and [Microsoft Silverlight](https://en.wikipedia.org/wiki/Microsoft_Silverlight), and also various [HDTV](https://en.wikipedia.org/wiki/HDTV) broadcasts over terrestrial ([Advanced Television Systems Committee standards](https://en.wikipedia.org/wiki/Advanced_Television_Systems_Committee_standards), [ISDB-T](https://en.wikipedia.org/wiki/ISDB-T), [DVB-T](https://en.wikipedia.org/wiki/DVB-T) or [DVB-T2](https://en.wikipedia.org/wiki/DVB-T2)), cable ([DVB-C](https://en.wikipedia.org/wiki/DVB-C)), and satellite ([DVB-S](https://en.wikipedia.org/wiki/DVB-S) and [DVB-S2](https://en.wikipedia.org/wiki/DVB-S2)).

With the use of H.264, bit rate savings of 50% or more compared to [MPEG-2 Part 2](https://en.wikipedia.org/wiki/MPEG-2_Part_2) are reported.

H.264 can often perform radically better than MPEG-2 video—typically obtaining the same quality at half of the bit rate or less, especially on high bit rate and high resolution situations.

## MPEG-2

MPEG-2 describes a combination of [lossy](https://en.wikipedia.org/wiki/Lossy_compression) [video compression](https://en.wikipedia.org/wiki/Video_compression) and [lossy](https://en.wikipedia.org/wiki/Lossy_compression) [audio data compression](https://en.wikipedia.org/wiki/Audio_data_compression) methods, which permit storage and transmission of movies using currently available storage media and transmission bandwidth. While MPEG-2 is not as efficient as newer standards such as [H.264](https://en.wikipedia.org/wiki/H.264) and [H.265/HEVC](https://en.wikipedia.org/wiki/HEVC), backwards compatibility with existing hardware and software means it is still widely used, for example in over-the-air digital television broadcasting and in the [DVD-Video](https://en.wikipedia.org/wiki/DVD-Video)standard.

## VP8/VP9/VP10

**VP8** is an [open](https://en.wikipedia.org/wiki/Open_format) and [royalty free](https://en.wikipedia.org/wiki/Royalty_free) [video compression format](https://en.wikipedia.org/wiki/Video_coding_format) owned by Google and created by [On2 Technologies](https://en.wikipedia.org/wiki/On2_Technologies) as a successor to [VP7](https://en.wikipedia.org/wiki/VP7).

[Opera](https://en.wikipedia.org/wiki/Opera_(web_browser)), [Firefox](https://en.wikipedia.org/wiki/Firefox_(web_browser)), [Chrome](https://en.wikipedia.org/wiki/Chrome_(web_browser)), and [Chromium](https://en.wikipedia.org/wiki/Chromium_(web_browser)) support playing VP8 video in [HTML5 video](https://en.wikipedia.org/wiki/HTML5_video) tag.[[3]](https://en.wikipedia.org/wiki/VP8#cite_note-Nokia_lines_up_patents_against_VP8_video_codec.-3) [Internet Explorer](https://en.wikipedia.org/wiki/Internet_Explorer) officially supports VP8 with a separate codec.[[4]](https://en.wikipedia.org/wiki/VP8#cite_note-IE9-4) According to Google VP8 is mainly used in connection with [WebRTC](https://en.wikipedia.org/wiki/WebRTC) and as a format for short looped animations, as a replacement for the [Graphics Interchange Format](https://en.wikipedia.org/wiki/Graphics_Interchange_Format) (GIF).

VP8 only supports [progressive scan](https://en.wikipedia.org/wiki/Progressive_scan) video signals with 4:2:0 [chroma subsampling](https://en.wikipedia.org/wiki/Chroma_subsampling) and 8 bits per [sample](https://en.wikipedia.org/wiki/Sampling_(signal_processing)).

The [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of a VP8 (and VP9) codec is found in the [programming library](https://en.wikipedia.org/wiki/Library_(computing)) libvpx which is released as [free software](https://en.wikipedia.org/wiki/Free_software).libvpx is capable of decoding VP8 video streams. The [Nvidia](https://en.wikipedia.org/wiki/Nvidia) [Tegra](https://en.wikipedia.org/wiki/Tegra) mobile chipsets have full VP8 hardware encoding and decoding (since [Tegra 4](https://en.wikipedia.org/wiki/Tegra_4)). [Nexus 5](https://en.wikipedia.org/wiki/Nexus_5) could use hardware encoding.

WebP is based on VP8's intra-frame coding and uses a container based on [Resource Interchange File Format](https://en.wikipedia.org/wiki/Resource_Interchange_File_Format) (RIFF).

While [H.264/MPEG-4 AVC](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) contains patented technology and requires licenses from patent holders and limited royalties for hardware, Google has irrevocably released the VP8 patents it owns under a royalty-free public license.

A review conducted by [streamingmedia.com](https://en.wikipedia.org/wiki/Information_Today,_Inc.) in May 2010 concluded that [H.264](https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) offers slightly better quality than VP8. Google's claims that VP8 offers the "highest quality real-time video delivery".

**VP9** is an [open](https://en.wikipedia.org/wiki/Open_format) and [royalty free](https://en.wikipedia.org/wiki/Royalty_free)[[1]](https://en.wikipedia.org/wiki/VP9#cite_note-Gigaom-1) [video coding format](https://en.wikipedia.org/wiki/Video_coding_format) developed by [Google](https://en.wikipedia.org/wiki/Google).

VP9 is a successor to [VP8](https://en.wikipedia.org/wiki/VP8) and competes with MPEGs [High Efficiency Video Coding](https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding) (HEVC/H.265). At first, VP9 was mainly used on Google's popular video platform [YouTube](https://en.wikipedia.org/wiki/YouTube).

x265 and libvpx demonstrate superior compression performance compared to x264, with bitrate savings reaching up to 50% especially at the higher resolutions. x265 outperforms libvpx for almost all resolutions and quality metrics, but the performance gap narrows (or even reverses) at 1080p.

# 相关分布式技术

## CDN

A **content delivery network** or **content distribution network** (**CDN**) is a globally distributed network of proxy [servers](https://en.wikipedia.org/wiki/Server_(computing))deployed in multiple [data centers](https://en.wikipedia.org/wiki/Data_center). The goal of a CDN is to serve content to end-users with high availability and high performance. CDNs serve a large fraction of the Internet content today, including web objects (text, graphics and scripts), downloadable objects (media files, software, documents), applications (e-commerce, portals), [live streaming](https://en.wikipedia.org/wiki/Live_streaming) media, on-demand streaming media, and [social networks](https://en.wikipedia.org/wiki/Social_network).

Besides better performance and availability, CDNs also offload the traffic served directly from the content provider's origin infrastructure, resulting in possible cost savings for the content provider.[[1]](https://en.wikipedia.org/wiki/Content_delivery_network#cite_note-nss10-1) In addition, CDNs provide the content provider a degree of protection from [DoS](https://en.wikipedia.org/wiki/Denial-of-service_attack) attacks by using their large distributed server infrastructure to absorb the attack traffic.

Several protocol suites are designed to provide access to a wide variety of content services distributed throughout a content network. The Internet Content Adaptation Protocol (ICAP) was developed in the late 1990s[[12]](https://en.wikipedia.org/wiki/Content_delivery_network#cite_note-12)[[13]](https://en.wikipedia.org/wiki/Content_delivery_network#cite_note-13) to provide an open standard for connecting application servers. A more recently defined and robust solution is provided by the Open Pluggable Edge Services (OPES) protocol.

以腾讯云CDN为例，假设我们要加速<http://haomiao.qq.com>上的安装包、图片什么的

1. CDN会为我们的域名分配域名，我们修改自己的DNS服务器，将haomiao.qq.com这个域名cname到CDN提供的域名



1. 我们在CDN的配置页面上配置该cdn域名的回源IP或者域名、缓存类型和时间等信息：



1. 浏览器访问haomiao.qq.com的时候，首先查询localDNS（假设配置的是8.8.8.8），
   1. localDNS通过DNS树查询到腾讯的GSLB
   2. GSLB返回说haomiao.qq.com 指向了haomiao.qq.com.cdn.dnsv1.com
   3. LocalDNS又去相应的dns服务器查询haomiao.qq.com.cdn.dnsv1.com
   4. Cdn的dns服务器根据客户端IP等信息，返回一个距离上比较近的CDN节点服务器IP
   5. localDNS将cname的域名（可能有多次）和最终的服务器IP一次性的返回给浏览器
   6. 浏览器向该cdn节点发起http请求，该节点服务器如果没有缓存对应的资源，就会经过外网到源节点（可能是cdn的级联的中间节点，不一定直接回源，避免拉跨了源站）拉取资源并缓存，同时返回一份给到浏览器。
2. 如果是要对https进行CDN加速（2016年的时候还比较少的CDN能支持），需要把源站的证书和私钥提交给CDN，CDN服务器起到中间人的作用。安全性应该会受到一定的影响。



1. 如果源站发布了新的版本，可以强制刷新（清空）CDN缓存

## P2P

In P2P networks, clients both provide and use resources. This means that unlike client-server systems, the content serving capacity of peer-to-peer networks can actually *increase* as more users begin to access the content (especially with protocols such as [Bittorrent](https://en.wikipedia.org/wiki/Bittorrent) that require users to share, refer a performance measurement study[[39]](https://en.wikipedia.org/wiki/Peer-to-peer#cite_note-39)). This property is one of the major advantages of using P2P networks because it makes the setup and running costs very small for the original content distributor.

# 小实验

### 实验一：live555作为点播服务器(rtsp)

使用live555非常容易就搭建了一个流媒体点播服务器，

使用vlc作为客户端播放webm视频或者mp3音频

使用格式工厂这个工具可以用来方便的转各种文件格式，

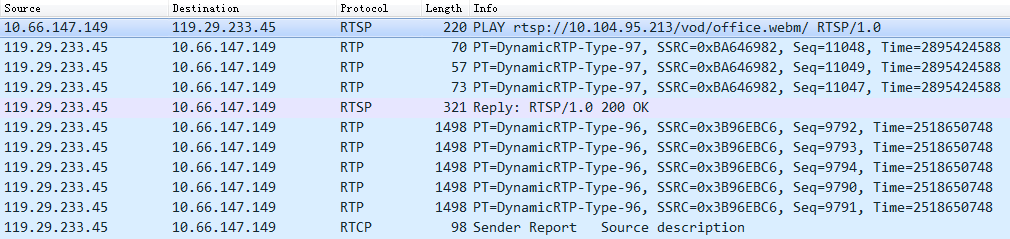
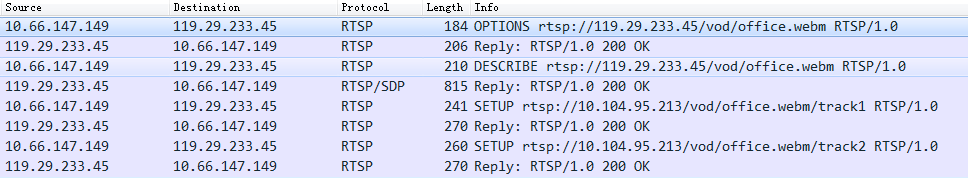
使用wireshark可以方便的抓包分析

通过抓包可以看到：服务器与播放器之间的交互主要是三类协议：

RTSP：流媒体控制，基于TCP

RTP：传输流媒体，基于UDP

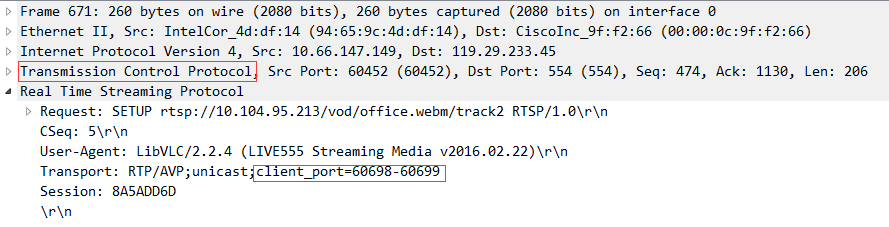
RTCP：状态报告和同步：基于UDP



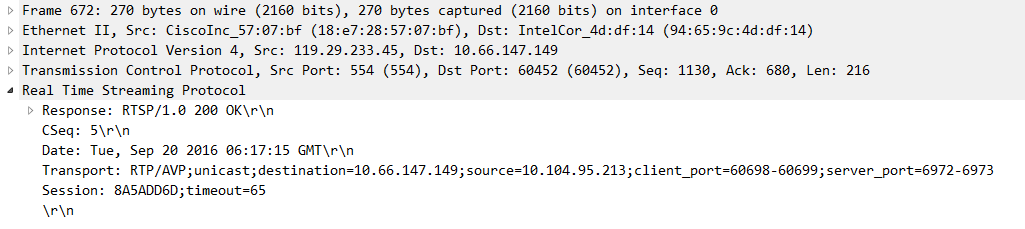
RTSP协议的格式与http协议非常类似，支持DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE等一系列方法；过程中会协商RTP和RTCP端口，支持服务器对播放器的push数据，所以如果防火墙比较严格，可能不能正常播放。

以一次RTSP协议交互为例：

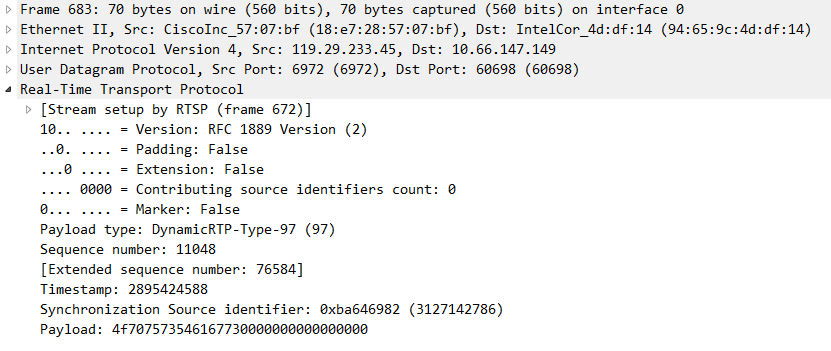
请求包：



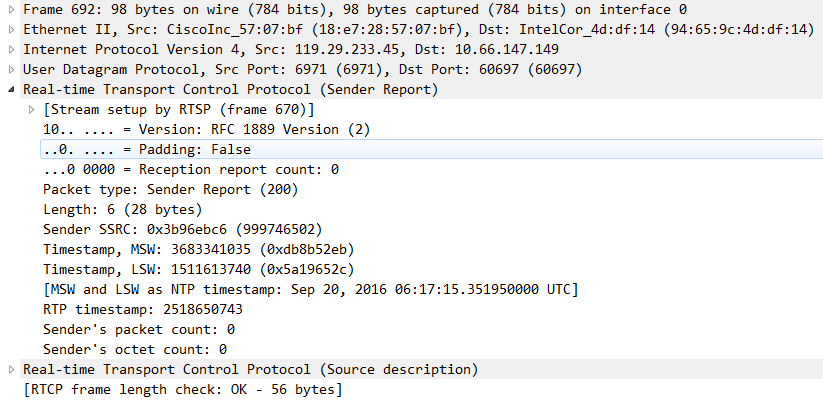
应答包：



服务器向播放器推流的某一个RTP报文：



某一个RTCP报文：



### 实验二：apache作为点播服务器(hls)

使用apache httpd一个普通的http服务器搭建基于hls协议的视频点播，使用vlc收看

Ffmpeg工具可将MP4文件切割成HLS文件：

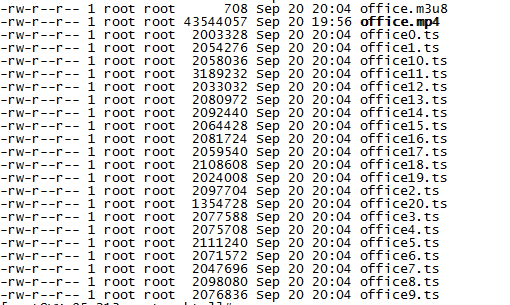
网上一堆垃圾文章说这样可以切割为2s大小的多个切片，实际上是不行的。

ffmpeg -i office.mp4 -f hls office.m3u8 -hls\_time 2 -hls\_list\_size 0 -c:v libx264 -c:a aac

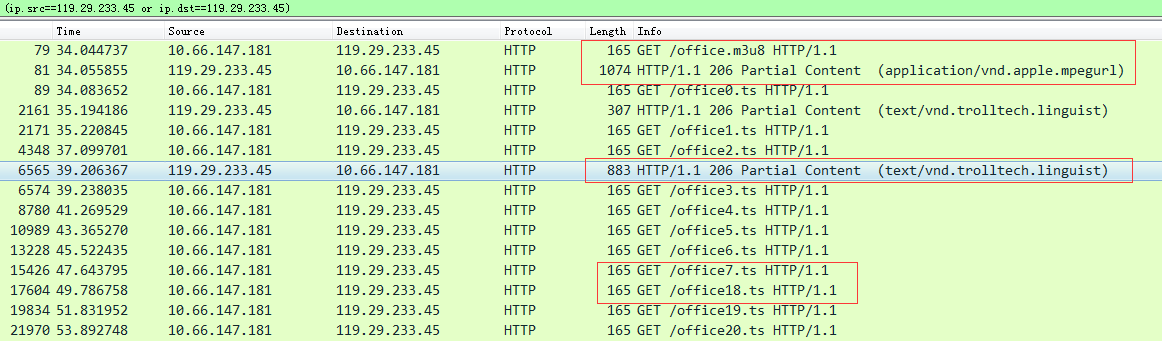
用这个命令才可以，不要问我为什么，我不知道

ffmpeg -y -i office.mp4 -hls\_time 2 -hls\_list\_size 0 -vcodec copy -acodec copy -vbsf h264\_mp4toannexb office.m3u8

我们将一个40秒长的office.mp4切割为20个切片和一个索引文件：



播放的时候抓包如下：



可以看出：

1. 请求和应答报文是http的（抓包好像有点问题，没有看到ts文件的应答数据流）
2. 播放器先请求m3u8索引文件，服务器应答索引文件里包含的数据切片列表
3. 播放器开始挨个请求切片，当请求到第三个切片的时候，开始同步播放音视频了
4. 用户拖拉进度条直接跳到尾部，浏览器直接请求第18个切片，跳过了6-17这些切片

### 实验三：nginx作为直播服务器(rtmp)

使用nginx作为服务器，ffmpeg作为客户端进行推流，vlc使用rtmp协议收看ffmpeg发布的视频。

【step1: 搭建nginx推流服务器】

下载nginx、nginx-rtmp-module，编译的时候注意一些参数：

./configure --prefix=/usr/local/nginx

--add-module=/root/install/nginx-rtmp-module-master

--with-http\_ssl\_module

--with-http\_mp4\_module

--with-http\_flv\_module

并在/usr/local/nginx/conf/nginx.conf配置文件中增加：

rtmp {

server {

listen 1935;

application myapp {

live on;

}

}

}

详细可以参考一下网上的一些资料：

<http://www.tuicool.com/articles/iauQNr>

<http://blog.csdn.net/u011244446/article/details/46956671>

【step2: 使用ffmpeg向服务器推流】

在官网下载已经编译好的ffmpeg工具，

用手机录制一段mp4视频

执行下面命令推流：

ffmpeg -re -i /tmp/office.mp4 -f flv -c copy rtmp://119.29.234.72/myapp/test

其中119.29.234.72是nginx服务器的外网IP

网上有一些资料介绍如何使用ffmpeg库函数，编写自己的推流程序：

<http://www.cnblogs.com/bandy/archive/2013/02/19/2916641.html>

<http://www.thinksaas.cn/topics/0/350/350162.html>

还有一些资料介绍如何开发android程序，获得camera的数据，然后推流直播：

<http://wpf814533631.iteye.com/blog/1684997>

<http://www.jianshu.com/p/e36b7258df53>

【step3: 使用vlc客户端收看上面的直播内容】

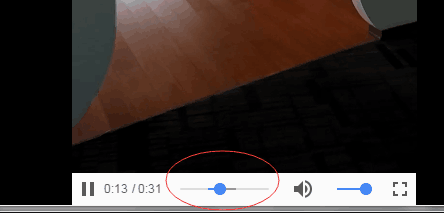
在vlc的“媒体”-> “打开网络串流”菜单中，输入

rtmp://119.29.234.72/myapp/test

即可收看ffmpeg推送的mp4文件内容。

【另：nginx也支持点播】

不需要修改配置文件，在nginx的http服务的文档根目录下建立vod子目录，将office.mp4放到vod下，通过web浏览器（我用的chrome）可直接点播该视频文件，且支持进度条拖动，从下面图片可以看出：web浏览器不是把整个mp4文件下载下来再播放，而是边下边播，而且可以跳转到指定的进度处。



### 实验四：nginx作为直播服务器（hls）

使用nginx作为服务器，ffmpeg作为客户端进行推流，浏览器使用hls协议收看ffmpeg发布的视频。

【step1: 搭建nginx推流服务器】

编译安装同实验三，不一样的是配置文件。

在http server配置里增加：

#HLS配置开始,这个配置为了`客户端`能够以http协议获取HLS的拉流

location /hls {

# Serve HLS fragments

types {

application/vnd.apple.mpegurl m3u8;

video/mp2t ts;

}

root html;

add\_header Cache-Control no-cache;

}

#HLS配置结束

在rtmp server配置里增加：

#增加对HLS支持开始

application hls {

live on;

hls on;

hls\_path /usr/local/nginx/html/hls;

hls\_fragment 5s;

}

#增加对HLS支持结束

特别注意hls\_path目录根据实际情况配置

【step2：使用ffmpeg工具向服务器推流】

ffmpeg -re -i /tmp/office.mp4 -f flv -c copy rtmp://119.29.234.72/hls/test

【step3:使用浏览器收看直播】

Android环境下v30或者以上的chrome浏览器、苹果safari v6以上的版本，都支持直接输入url收看直播(注意有m3u8的后缀)：

http://119.29.234.72/hls/test.m3u8

如果没有这样的浏览器，那用vlc输入上述url也可以收看

直播过程中，可以看到/usr/local/nginx/html/hls目录下不断更新的视频文件片段

网上可参考的文档：

<http://www.cnblogs.com/jys509/p/5653720.html>

### 实验五：使用SRS作为直播服务器（HTTP-FLV）

**Step1：**搭建SRS服务器，并修改配置，使其支持HTTP-FLV协议，SRS服务器会将收到的RTMP推流，转封装为HTTP-FLV形式提供播放端拉取

listen 1935;

max\_connections 1000;

srs\_log\_tank file;

srs\_log\_file ./objs/srs.log;

http\_api {

enabled on;

listen 1985;

}

http\_server {

enabled on;

listen 8080;

dir ./objs/nginx/html;

}

stats {

network 0;

disk sda sdb xvda xvdb;

}

vhost \_\_defaultVhost\_\_ {

http\_remux {

enabled on;

mount [vhost]/[app]/[stream].flv;

hstrs on;

}

}

**step2：**使用ffmpeg工具向服务器推流

ffmpeg -re -i /tmp/office.mp4 -f flv -c copy rtmp://123.206.199.45/live/demo1

另外使用百度云推流SDK的demo也可以快速创建一个推流app向服务器推流：

<https://cloud.baidu.com/product/lss.html>



**step3：**播放rtmp流（可选）

打开SRS提供的web页面：<http://123.206.199.45:8080/，点击“JWPlayer6>播放”tab，可以直接在页面播放rtmp流

**Step4:**播放HTTP-FLV流

打开VLC播放器，输入<http://123.206.199.45:8080/live/demo1.flv>，即可播放