# 流媒体传输协议介绍

<http://www.planetb.ca/syntax-highlight-word> 收藏用于黏贴代码

# RTSP

RTSP协议以客户服务器方式工作，，如：暂停/继续、后退、前进等。它是一个多媒体播放控制协议，用来使用户在播放从因特网下载的实时数据时能够进行控制,

因此 RTSP 又称为“因特网录像机遥控协议”。

RTSP(Real-Time Stream Protocol)是一种基于文本的应用层协议，在语法及一些消息参数等方面，RTSP协议与HTTP协议类似。 是TCP/IP协议体系中的一个应用层协议, 由哥伦比亚大学, 网景和RealNetworks公司提交的IETF RFC标准. 对应的RFC编号是2326，可以在这里搜索 RFC Editor

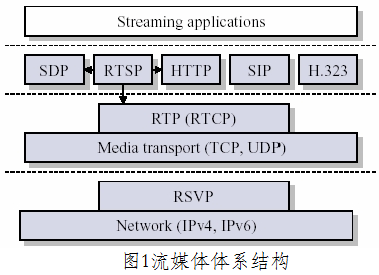
该协议定义了一对多应用程序如何有效地通过IP网络传送多媒体数据. RTSP在体系结构上位于RTP和RTCP之上, 它使用TCP或RTP完成数据传输.

RTSP被用于建立的控制媒体流的传输，它为多媒体服务扮演“网络远程控制”的角色。尽管有时可以把RTSP控制信息和媒体数据流交织在一起传送，但一般情况RTSP本身并不用于转送媒体流数据。媒体数据的传送可通过RTP/RTCP等协议来完成。

该协议用于C/S模型, 是一个基于文本的协议, 用于在客户端和服务器端建立和协商实时流会话.

## 网络体系

RTSP是类似http的应用层协议，一个典型的流媒体框架网络体系可参考下图



**一次基本的RTSP操作过程:**

* 首先，客户端连接到流服务器并发送一个RTSP描述命令（DESCRIBE）。
* 流服务器通过一个SDP描述来进行反馈，反馈信息包括流数量、媒体类型等信息。
* 客户端再分析该SDP描述，并为会话中的每一个流发送一个RTSP建立命令(SETUP)，RTSP建立命令告诉服务器客户端用于接收媒体数据的端口。流媒体连接建立完成后，
* 客户端发送一个播放命令(PLAY)，服务器就开始在UDP上传送媒体流（RTP包）到客户端。 在播放过程中客户端还可以向服务器发送命令来控制快进、快退和暂停等。
* 最后，客户端可发送一个终止命令(TERADOWN)来结束流媒体会话

1. sequenceDiagram
2. 客户端->>服务器:DESCRIBE
3. 服务器->>客户端: 200 OK (SDP)
4. 客户端->>服务器:SETUP
5. 服务器->>客户端: 200 OK
6. 客户端->>服务器:PLAY
7. 服务器->>客户端: (RTP包)

客户端->>服务器:DESCRIBE

服务器->>客户端: 200 OK (SDP)

客户端->>服务器:SETUP

服务器->>客户端: 200 OK

客户端->>服务器:PLAY

服务器->>客户端: (RTP包)

## 协议特点

* 可扩展性: 新方法和参数很容易加入RTSP.
* 易解析: RTSP可由标准HTTP或MIME解析器解析.
* 安全: RTSP使用网页安全机制.
* 独立于传输: RTSP可使用不可靠数据报协议(EDP), 可靠数据报协议(RDP); 如要实现应用级可靠, 可使用可靠流协议.
* 多服务器支持: 每个流可放在不同服务器上, 用户端自动与不同服务器建立几个并发控制连接, 媒体同步在传输层执行.
* 记录设备控制: 协议可控制记录和回放设备.
* 流控与会议开始分离: 仅要求会议初始化协议提供, 或可用来创建惟一会议标识号. 特殊情况下, 可用SIP或H.323来邀请服务器入会.
* 适合专业应用: 通过SMPTE时标, RTSP支持帧级精度, 允许远程数字编辑.
* 演示描述中立: 协议没强加特殊演示或元文件, 可传送所用格式类型; 然而, 演示描述至少必须包括一个RTSP URL.
* 代理与防火墙友好: 协议可由应用和传输层防火墙处理. 防火墙需要理解SETUP方法, 为UDP媒体流打开一个“缺口”.
* HTTP友好: 此处, RTSP明智地采用HTTP观念, 使现在结构都可重用. 结构包括Internet内容选择平台(PICS). 由于在大多数情况下控制连续媒体需要服务器状态, RTSP不仅仅向HTFP添加方法.
* 适当的服务器控制: 如用户启动一个流, 必须也可以停止一个流.
* 传输协调: 实际处理连续媒体流前, 用户可协调传输方法.
* 性能协调: 如基本特征无效, 必须有一些清理机制让用户决定哪种方法没生效. 这允许用户提出适合的用户界面.

## RTSP协议与HTTP协议区别

RTSP引入了几种新的方法，比如DESCRIBE、PLAY、SETUP 等，并且有不同的协议标识符，RTSP为rtsp 1.0,HTTP为http 1.1；

HTTP是无状态的协议，而RTSP为每个会话保持状态；

RTSP协议的客户端和服务器端都可以发送Request请求，而在HTTPF 协议中，只有客户端能发送Request请求。

在RTSP协议中，载荷数据一般是通过带外方式来传送的(除了交织的情况)，及通过RTP协议在不同的通道中来传送载荷数据。而HTTP协议的载荷数据都是通过带内方式传送的，比如请求的网页数据是在回应的消息体中携带的。

使用ISO 10646(UTF-8) 而不是ISO 8859-1，以配合当前HTML的国际化；

RTSP使用URI请求时包含绝对URI。而由于历史原因造成的向后兼容性问题，HTTP/1.1只在请求中包含绝对路径，把主机名放入单独的标题域中；

## RTSP 的报文结构

### RTSP URL的语法结构

一个终端用户是通过在播放器中输入URL地址开始进行观看流媒体业务的第一步，而对于使用RTSP协议的移动流媒体点播而言，URL的一般写法如下：

一个以“rtsp”或是“rtspu”开始的URL链接用于指定当前使用的是RTSP 协议。RTSP URL的语法结构如下：

1. rtsp\_url = (“rtsp:”| “rtspu:”) “//” host [“:”port”] /[abs\_path]/content\_name

**host**：可以是一个有效的域名或是IP地址。

**port**：端口号，对于RTSP协议来说，缺省的端口号为554。当我们在确认流媒体服务器提供的端口号为554时，此项可以省略

说明：当HMS服务器使用的端口号为554时，我们在写点播链接时，可以不用写明端口号，但当使用非554端口时，在RTSP URL中一定要指定相应的端口。

**abs\_path**: 为RTSPServer中的媒体流资源标识，见下章节的录像资源命名。

RTSPURL用来标识RTSPServer的媒体流资源，可以标识单一的媒体流资源，也可以标识多个媒体流资源的集合。

### RTSP的报文结构

**RTSP是一种基于文本的协议**，用CRLF作为一行的结束符。使用基于文本协议的好处在于我们可以随时在使用过程中的增加自定义的参数，也可以随便将协议包抓住很直观的进行分析。

RTSP有两类报文：请求报文和响应报文。请求报文是指从客户向服务器发送请求报文，响应报文是指从服务器到客户的回答。

由于 RTSP 是面向正文的(text-oriented)，因此在报文中的每一个字段都是一些 ASCII 码串，因而每个字段的长度都是不确定的。

### 请求报文

RTSP报文由三部分组成，即开始行、首部行和实体主体。在请求报文中，开始行就是请求行.

RTSP请求报文的结构如下图所示

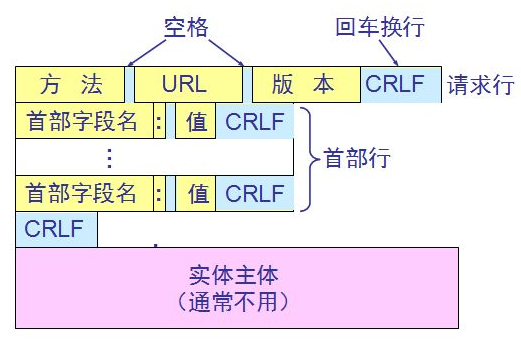


图2 RTSP请求报文的结构

RTSP请求报文的方法包括：**OPTIONS、DESCRIBE、SETUP、TEARDOWN、PLAY、PAUSE、GET\_PARAMETER和SET\_PARAMETER**。

一个请求消息（a request message）即可以由客户端向服务端发起也可以由服务端向客户端发起。请求消息的语法结构如下

1. Request = Request-Line
3. \*(  general-header   | request-header | entity-header)
5. CRLF
7. [message-body]

**1. Request Line**

请求消息的第一行的语法结构如下：

1. Request-Line    =   Method 空格 Request-URI 空格 RTSP-Version CRLF

其中在消息行中出现的第一个单词即是所使用的信令标志。目前已经有的信息标志如下：

1. Method      =   “DESCRIBE”
2. |   “ANNOUNCE”
3. |   “GET\_PARAMETER”
4. |   “OPTIONS”
5. |   “PAUSE”
6. |   “PLAY”
7. |   “RECORD”
8. |   “REDIRECT”
9. |   “SETUP”
10. |   “SET\_PARAMETER”
11. |   “TEARDOWN”

例子：

1. DESCRIBE rtsp://211.94.164.227/3.3gp RTSP/1.0

**2. Request Header Fields**

在消息头中除了第一行的内容外，还有一些需求提供附加信息。其中有些是一定要的，后续我们会详细介绍经常用到的几个域的含义。

1. Request-header      =   Accept
2. |   Accept-Encoding
3. |   Accept-Language
4. |   Authorization
5. |   From
6. |   If-Modified-Since
7. |   Range
8. |   Referer
9. |   User-Agent

### 响应报文

响应报文的开始行是状态行，RTSP响应报文的结构如下图所示

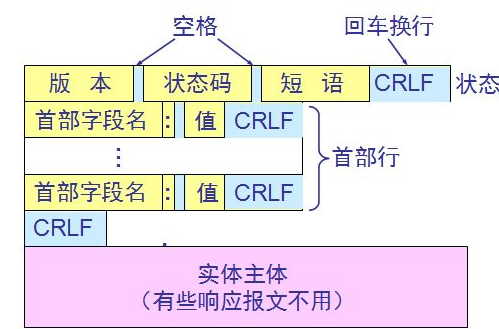


图3 RTSP响应报文的结构

响应消息的语法结构如下：

1. Response = Status-Line
2. \*( general-header
3. | response-header
4. | entity-header)
5. CRLF
6. [message-body]

**1. Status-Line**

响应消息的第一行是状态行（status-line），每个元素之间用空格分开。除了最后的CRLF之外，在此行的中间不得有CR或是LF的出现。它的语法格式如下，

1. Status-Line = RTSP-Version 空格 Status-Code 空格 Reason-Phrase CRLF

状态码（Status-Code） 是一个三位数的整数，用于描述接收方对所收到请求消息的执行结果

Status-Code的第一位数字指定了这个回复消息的种类，一共有5类：

* - [ ] 1XX: Informational – 请求被接收到，继续处理
* - [ ] 2XX: Success – 请求被成功的接收，解析并接受
* - [ ] 3XX: Redirection – 为完成请求需要更多的操作
* - [ ] 4XX: Client Error – 请求消息中包含语法错误或是不能够被有效执行
* - [ ] 5XX: Server Error – 服务器响应失败，无法处理正确的有效的请求消息

我们在处理问题时经常会遇到的状态码有如下：

|  | |
| --- | --- |
| Status-Code | = |
| . | | |
| . | | |
| . | | |

**2. Response Header Fields**

在响应消息的域中存放的是无法放在Status-Line中,而又需要传送给请求者的一些附加信息。

1. Response-header    =   Location
2. |   Proxy-Authenticate
3. |   Public
4. |   Retry-After
5. |   Server
6. |   Vary
7. |   WWW-Authenticate

## RTSP的主要方法

| **方法** | **方向** | **对象** | **要求** | **含义** |
| --- | --- | --- | --- | --- |
| DESCRIBE | C->S | P,S | 推荐 | 检查演示或媒体对象的描述，也允许使用接收头指定用户理解的描述格式。DESCRIBE的答复-响应组成媒体RTSP初始阶段 |
| ANNOUNCE | C->S S->C | P,S | 可选 | 当从用户发往服务器时，ANNOUNCE将请求URL识别的演示或媒体对象描述发送给服务器；反之，ANNOUNCE实时更新连接描述。如新媒体流加入演示，整个演示描述再次发送，而不仅仅是附加组件，使组件能被删除 |
| GET\_PARAMETER | C->S S->C | P,S | 可选 | GET\_PARAMETER请求检查RUL指定的演示与媒体的参数值。没有实体体时，GET\_PARAMETER也许能用来测试用户与服务器的连通情况 |
| OPTIONS | C->S S->C | P,S | 要求 | 可在任意时刻发出OPTIONS请求，如用户打算尝试非标准请求，并不影响服务器状态 |
| PAUSE | C->S | P,S | 推荐 | PAUSE请求引起流发送临时中断。如请求URL命名一个流，仅回放和记录被停止；如请求URL命名一个演示或流组，演示或组中所有当前活动的流发送都停止。恢复回放或记录后，必须维持同步。在SETUP消息中连接头超时参数所指定时段期间被暂停后，尽管服务器可能关闭连接并释放资源，但服务器资源会被预订 |
| PLAY | C->S | P,S | 要求 | PLAY告诉服务器以SETUP指定的机制开始发送数据；直到一些SETUP请求被成功响应，客户端才可发布PLAY请求。PLAY请求将正常播放时间设置在所指定范围的起始处，发送流数据直到范围的结束处。PLAY请求可排成队列，服务器将PLAY请求排成队列，顺序执行 |
| RECORD | C->S | P,S | 可选 | 该方法根据演示描述初始化媒体数据记录范围，时标反映开始和结束时间；如没有给出时间范围，使用演示描述提供的开始和结束时间。如连接已经启动，立即开始记录，服务器数据请求URL或其他URL决定是否存储记录的数据；如服务器没有使用URL请求，响应应为201（创建），并包含描述请求状态和参考新资源的实体与位置头。支持现场演示记录的媒体服务器必须支持时钟范围格式，smpte格式没有意义 |
| REDIRECT | S->C | P,S | 可选 | 重定向请求通知客户端连接到另一服务器地址。它包含强制头地址，指示客户端发布URL请求；也可能包括参数范围，以指明重定向何时生效。若客户端要继续发送或接收URL媒体，客户端必须对当前连接发送TEARDOWN请求，而对指定主执新连接发送SETUP请求 |
| SETUP | C->S | S | 要求 | 对URL的SETUP请求指定用于流媒体的传输机制。客户端对正播放的流发布一个SETUP请求，以改变服务器允许的传输参数。如不允许这样做，响应错误为”455 Method Not Valid In This State”。为了透过防火墙，客户端必须指明传输参数，即使对这些参数没有影响 |
| SET\_PARAMETER | C->S S->C | P,S | 可选 | 请求设置演示或URL指定流的参数值。请求仅应包含单个参数，允许客户端决定某个特殊请求为何失败。如请求包含多个参数，所有参数可成功设置，服务器必须只对该请求起作用。服务器必须允许参数可重复设置成同一值，但不让改变参数值。注意：媒体流传输参数必须用SETUP命令设置。将设置传输参数限制为SETUP有利于防火墙。将参数划分成规则排列形式，结果有更多有意义的错误指示 |
| TEARDOWN | C->S | P,S | 要求 | TEARDOWN请求停止给定URL流发送，释放相关资源。如URL是此演示URL，任何RTSP连接标识不再有效。除非全部传输参数是连接描述定义的，SETUP请求必须在连接可再次播放前发布 |

注：P—演示，C—客户端,S—服务器， S（对象栏）—流

信令指的是在Request-URI中指定的需要被接收者完成的操作。信令（The method）大小写敏感，不能以字符”$”开始，并且一定要是一个标记符。

## RTSP重要头字段参数

**Accept:**

用于指定客户端可以接受的媒体描述信息类型。比如:

Accept: application/rtsl, application/sdp;level=2

**Bandwidth:**

用于描述客户端可用的带宽值。

**CSeq：**

指定了RTSP请求回应对的序列号，在每个请求或回应中都必须包括这个头字段。对每个包含一个给定序列号的请求消息，都会有一个相同序列号的回应消息。

**Rang：**

用于指定一个时间范围，可以使用SMPTE、NTP或clock时间单元。

**Session:**

Session头字段标识了一个RTSP会话。Session ID 是由服务器在SETUP的回应中选择的，客户端一当得到Session ID后，在以后的对Session 的操作请求消息中都要包含Session ID.

**Transport:**

Transport头字段包含客户端可以接受的转输选项列表，包括传输协议，地址端口，TTL等。服务器端也通过这个头字段返回实际选择的具体选项。如:

Transport: RTP/AVP;multicast;ttl=127;mode=”PLAY”,

RTP/AVP;unicast;client\_port=3456-3457;mode=”PLAY”

## 简单的RTSP消息交互过程

C表示RTSP客户端,S表示RTSP服务端

**第一步：查询服务器端可用方法**

C->S OPTION request //询问S有哪些方法可用

S->C OPTION response //S回应信息的public头字段中包括提供的所有可用方法

**第二步：得到媒体描述信息**

C->S DESCRIBE request //要求得到S提供的媒体描述信息

S->C DESCRIBE response //S回应媒体描述信息，一般是sdp信息

**第三步：建立RTSP会话**

C->S SETUP request //通过Transport头字段列出可接受的传输选项，请求S建立会话

S->C SETUP response //S建立会话，通过Transport头字段返回选择的具体转输选项，并返回建立的Session ID;

**第四步：请求开始传送数据**

C->S PLAY request //C请求S开始发送数据

S->C PLAY response //S回应该请求的信息

**第五步： 数据传送播放中**

S->C 发送流媒体数据 // 通过RTP协议传送数据

**第六步：关闭会话，退出**

C->S EARDOWN request //C请求关闭会话

S->C TEARDOWN response //S回应该请求

上述的过程只是标准的、友好的RTSP流程，但实际的需求中并不一定按此过程。

**其中第三和第四步是必需的**！第一步，只要服务器和客户端约定好有哪些方法可用，则option请求可以不要。第二步，如果我们有其他途径得到媒体初始化描述信息（比如http请求等等），则我们也不需要通过rtsp中的describe请求来完成。

## RTSP的请求响应示例

RTSP的主要方法：其中C是客户端，S是服务端。

**OPTIONS**

1. C->S:       OPTION request //询问S有哪些方法可用
2. S->C:       OPTION response //S回应信息中包括提供的所有可用方法

\* 客户端到服务端 \*

1. OPTIONS rtsp://218.207.101.236:554/mobile/3/67A451E937422331 RTSP/1.0?
2. Cseq: 1

服务端对OPTIONS的回应：（服务器的回应信息会在Public字段列出提供的方法。）

1. RTSP/1.0 200 OK?
2. Server: PVSS/1.4.8 (Build/20090111; Platform/Win32; Release/StarValley; )?
3. Cseq: 1?
4. Public: DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE, OPTIONS, ANNOUNCE, RECORD

**DESCRIBE**

1. C->S:      DESCRIBE request //要求得到S提供的媒体初始化描述信息
2. S->C:      DESCRIBE response //S回应媒体初始化描述信息，主要是sdp

客户端到服务端的请求举例:（客户端向服务器端发送DESCRIBE，用于得到URI所指定的媒体描述信息，一般是SDP信息。客户端通过Accept头指定客户端可以接受的媒体述信息类型。）

1. DESCRIBE rtsp://218.207.101.236:554/mobile/3/67A451E937422331/8jH5QPU5GWS07Ugn.sdp RTSP/1.0
2. Cseq: 2

服务端对DESCRIBE的回应：（服务器回应URI指定媒体的描述信息）

1. RTSP/1.0 200 OK
2. Server: PVSS/1.4.8 (Build/20090111; Platform/Win32; Release/StarValley; )
3. Cseq: 2
4. Content-length: 421
5. Date: Mon, 03 Aug 2009 08:21:33 GMT
6. Expires: Mon, 03 Aug 2009 08:21:33 GMT
7. Content-Type: application/sdp
8. x-Accept-Retransmit: our-retransmit
9. x-Accept-Dynamic-Rate: 1
10. Content-Base: rtsp://218.207.101.236:554/mobile/3/67A451E937422331/8jH5QPU5GWS07Ugn.sdp/

13. v=0
14. o=MediaBox 127992 137813 IN IP4 0.0.0.0
15. s=RTSP Session
16. i=Starv Box Live Cast
17. c=IN IP4 218.207.101.236
18. t=0 0
19. a=range:npt=now-
20. a=control:\*
21. m=video 0 RTP/AVP 96
22. b=AS:20
23. a=rtpmap:96 MP4V-ES/1000
24. a=fmtp:96 profile-level-id=8; config=000001b008000001b5090000010000000120008440fa282c2090a31f; decode\_buf=12586
25. a=range:npt=now-
26. a=framerate:5
27. a=framesize:96 176-144
28. a=cliprect:0,0,144,176
29. a=control:trackID=1

# 参考文档

[1] EasyDarwin开源社区流媒体视频课程：流媒体传输控制协议(RTSP RTP SDP)详解之RTSP http://blog.csdn.net/xiejiashu/article/details/51914024