# 第20章 多媒体的传输

# 第20章 多媒体的传输

# 第20章 多媒体的传输

- 在IP网上传输声音和影视的研究和开发已成为网络应用的重要方向。
- 多媒体应用的主要问题是如何保障实时多媒体数据的传输质量，尤其是对那些实时互动的应用。
- 保障传输质量的技术集中体现在多媒体传输协议和控制协议上

# 20.1 多媒体应用协议套

- 在IP网上的多媒体应用有两种类型的协议
  - 会话协议

    两台设备或两个站点之间的持续连接和多媒体数据交换

  - 信令协议

    通信双方建立和控制连接所需信息的交换方法

# 20.1 多媒体应用协议套

## 20.1.1 多媒体应用协议套

- 数据传送服务：对数据的时延要求不高
- 多媒体应用：时延短和抖动小

在IP网络上，支持实时视听数据传输的协议构成了多媒体应用协议套

# 20.1 多媒体应用协议套

- (1) 实时传输协议(Real-time Transport Protocol，RTP)
  - 位于应用层和UDP之间，用于传输包括声音和影视等实时数据的协议。
- (2) 实时控制协议(Real-Time Control Protocol, RTCP)
  - 与实时协议(RTP)一起工作的传输控制协议，用于在发送者和接收者之间交换控制实时数据传输的消息。
- (3) 实时流播协议(Real-Time Streaming Protocol，RTSP)
  - 网上传输实时、现场的或存储的声音、影视和三维动画的控制协议

# 20.1 多媒体应用协议套

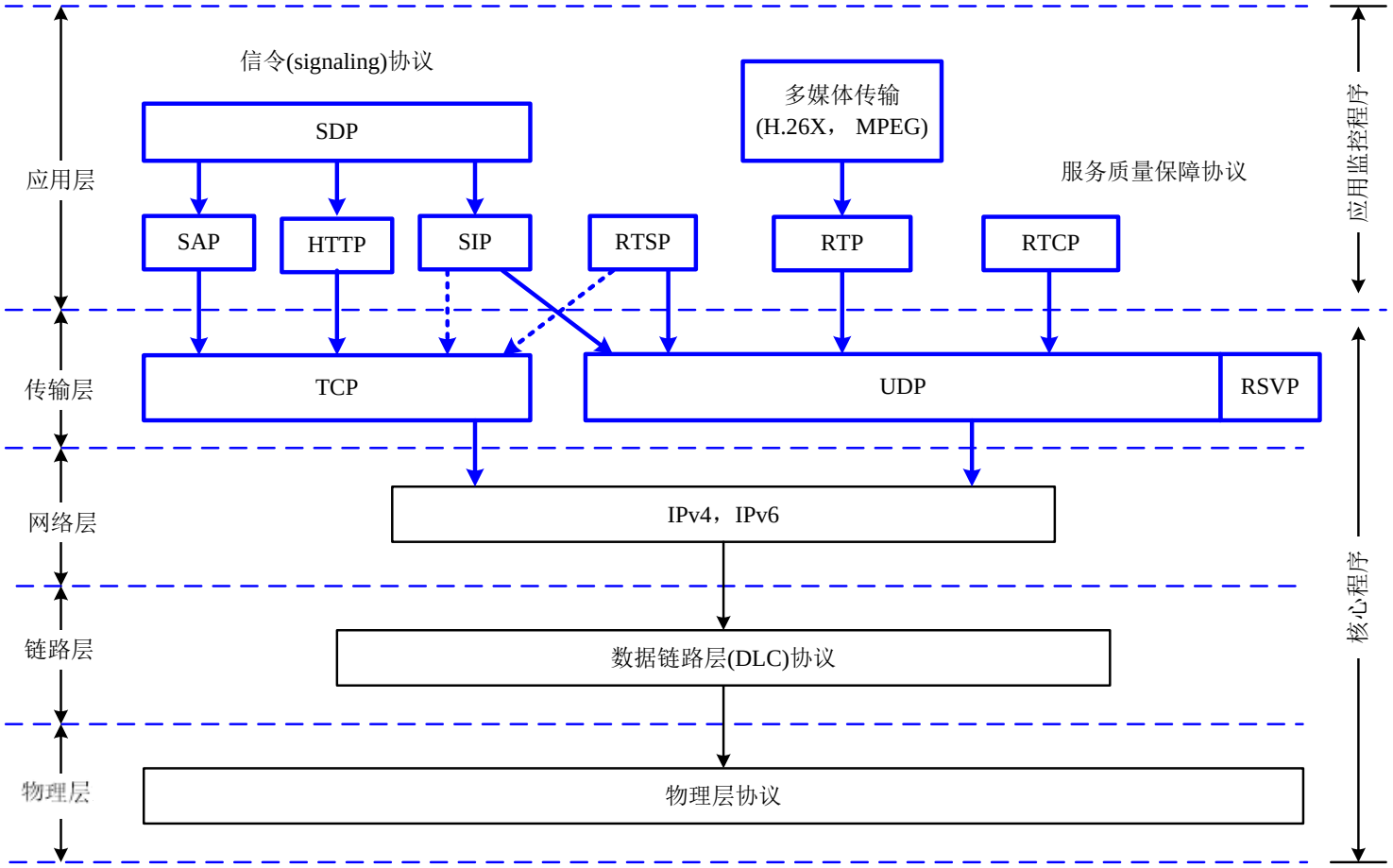
- (4) 资源保留协议(Resource Reservation Protocol, RSVP)
  - IETF核准的为“带宽按需调配”开发的传输协议。
- (5) 会话启动协议(Session Initiation Protocol，SIP)
  - 在IP网上建立呼叫的协议。
- (6) 会话描述协议(Session Description Protocol，SDP)
  - 描述流媒体初始化参数的格式

# 20.1 多媒体应用协议套

- (7) 会话通告协议(Session Announcement Protocol, SAP)
  - 用于向参与多目标广播(multicast)的潜在主机发布广播会话消息。

- 以上这些重要协议构成的协议套也称互联网多媒体协议套(Internet multimedia protocol stack)

# 多媒体应用协议套




多媒体应用协议套是TCP/IP协议套中的一部分

# 20.2 实时传输和控制协议

- 实时传输协议(RTP)和实时控制协议(RTCP)是为网上传送实时多媒体数据开发的协议
  - RTP提供端对端的实时数据传输服务
  - RTCP协议用于监视和控制实时数据的传输

- RTP和RTCP协议的详细规范定义在RFC 3550(2003)中，并取代1996年发布的RFC 1889

# 20.2.1 实时传输协议(RTP)

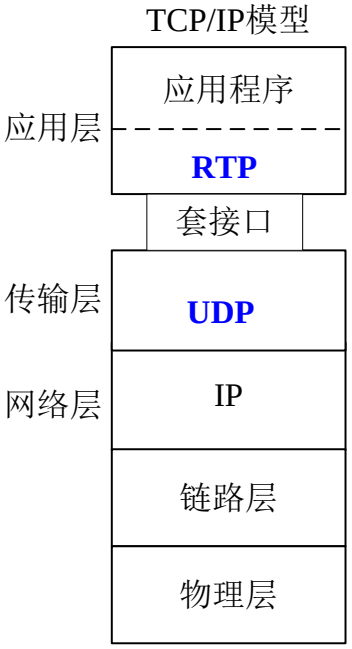
- **实时传输协议(Real-time Transport Protocol，RTP)**
  - 用于通过IP网络传送音频和视频的网络协议，定义了标准的数据包
  - 应用:声音点播(AoD)、影视点播(VoD)、因特网电话和电视会议等
  - 提供端对端的实时声音和视像数据的传输，而对声音和视像数据的压缩和编码格式没有限制。

# 20.2.1 实时传输协议(RTP)

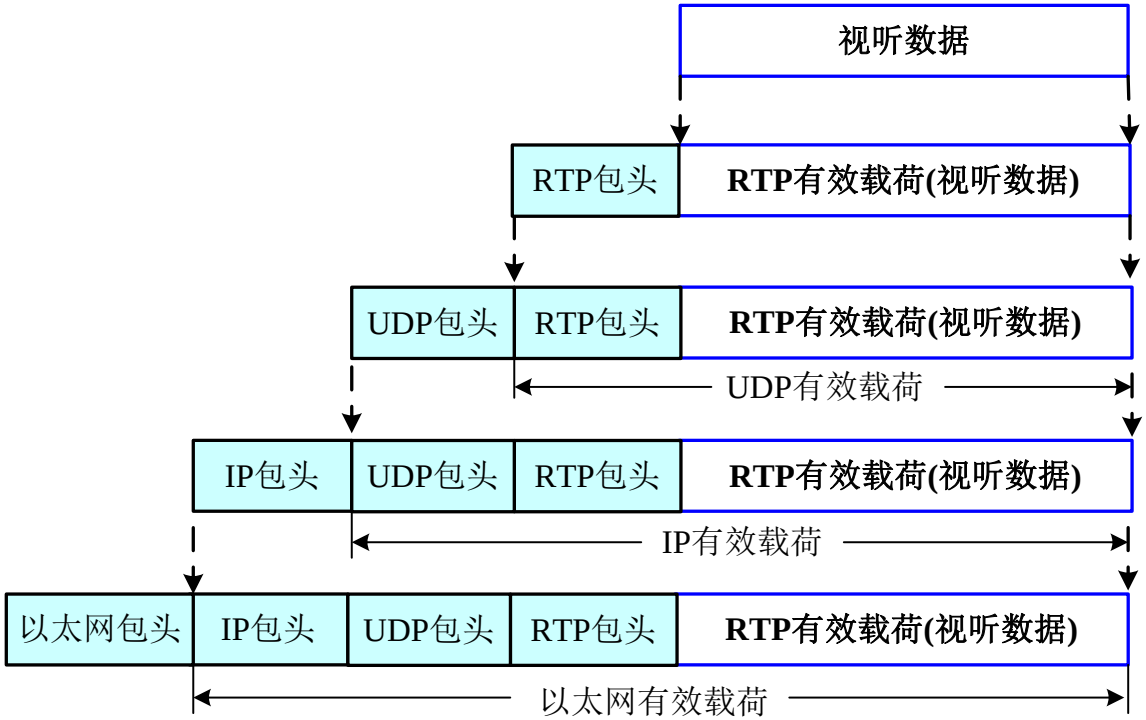
- RTP 为“实时数据”的传输协议。
  - 本身不提供任何机制来确保把实时数据及时送到接收端，不保证在递送过程中不丢失数据包，也没有使用防止数据包次序被打乱的方法
  - 提供了减少或消除抖动、视听数据同步和视听数据流复合的方法。
  - RTP协议需要使用RTCP来提高服务质量

# 20.2.1 实时传输协议(RTP)

- **RTP协议原理**



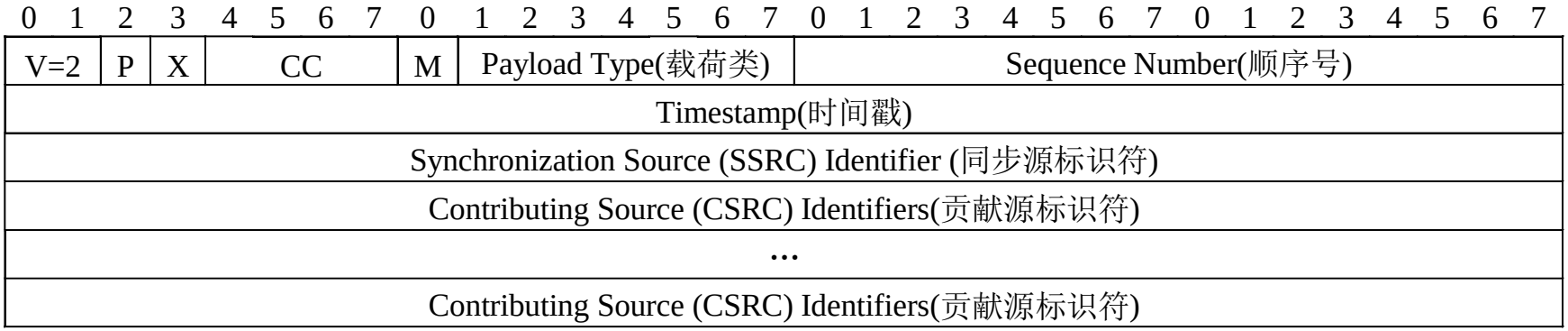

(a) RTP在协议套中的位置




(b) 数据包的封装

# 20.2.1 实时传输协议(RTP)

- RTP包头组成：有效载荷类型、顺序号、时间戳和同步源标识符



# 20.2.1 实时传输协议(RTP)

**表20-2 RFC 3551指定的部分声音有效载荷类型**

| PT | 编码名称 | *时钟率(Hz) | | PT | 编码名称 | 时钟率(Hz) |
|---|---|---|---|---|---|---|
| 0 | PCM-μ率 | 8,000 | | 10 | L16 | 44,100 |
| 1 | 保留 | | | 11 | L16 | 44,100 |
| 2 | 保留 | | | 12 | QCELP | 8,000 |
| 3 | GSM | 8,000 | | 13 | CN | 8,000 |
| 4 | G723 | 8,000 | | 14 | MPEG-Audio | 90,000 |
| 5 | DVI4 | 8,000 | | 15 | G728 | 8,000 |
| 6 | DVI4 | 16,000 | | 16 | DVI4 | 11,025 |
| 7 | LPC | 8,000 | | 17 | DVI4 | 22,050 |
| 8 | PCM-A率 | 8,000 | | 18 | G729 | 8,000 |
| 9 | G722 | 8,000 | | 19 | 保留 | |

* 时钟率用于产生时间戳

# 20.2.1 实时传输协议(RTP)

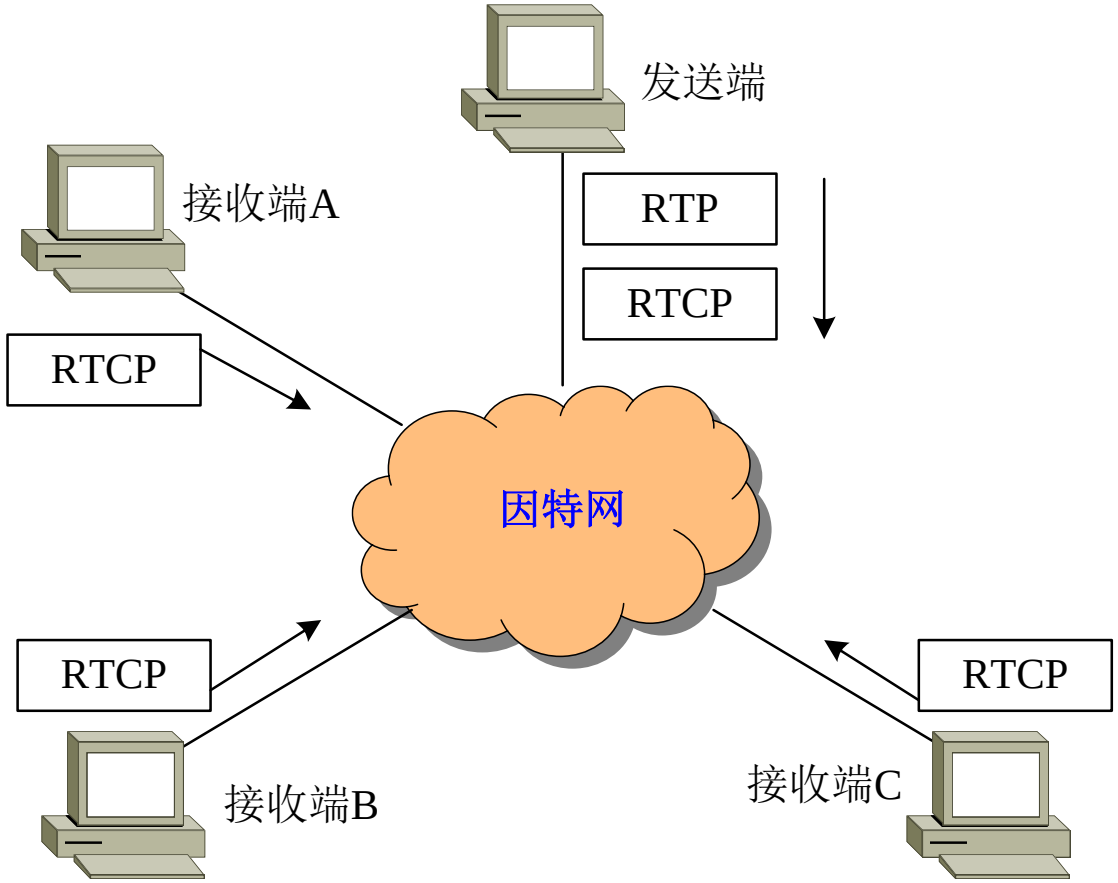
**表20-3 RFC 3551指定的部分视像有效载荷类型**

| PT | 编码名称 | 媒体类型 | 时钟率(Hz) | 注释 |
|---|---|---|---|---|
| 26 | JPEG | V | 90,000 | |
| 27 | 未指定 | V | | |
| 28 | nv | V | 90,000 | Sun公司的专有格式 |
| 29 | 未指定 | V | | |
| 30 | 未指定 | V | | |
| 31 | H261 | V | 90,000 | |
| 32 | MPV | V | 90,000 | MPEG-1和-2 |
| 33 | MP2T | AV | 90,000 | MPEG-2传输流 |
| 34 | H263 | V | 90,000 | |
| * 在35-127中，有些作为保留、未指定或动态指定 | | | | |

# 20.2.2 实时控制协议(RTCP)

- **实时控制协议(Real-Time Control Protocol, RTCP)**
  - 主要功能:为收发两端的应用程序提供有关会话传送质量的数据包
    - 每个RTCP数据包不是封装声音数据或视像数据，而是封装收发两端的统计信息。
    - RTCP规范没有指定应用程序如何使用控制数据包中的信息，这完全取决于应用程序开发人员

# 20.2.2 实时控制协议(RTCP)




每个参与者周期性地发送RTCP控制数据包

# 20.2.2 实时控制协议(RTCP)

- RTCP数据包的5种类型
  - (1) SR(Sender report)—发送者报告
    - 实时数据的传送摘要，包括RTP流的同步源标识符，当前的时间，发送的数据包数目和发送的字节数等
  - (2) RR(Receiver report)—接收者报告
    - 包括丢失的数据包、最后接收到的顺序号和平均的抖动间隔等统计信息

# 20.2.2 实时控制协议(RTCP)

- (3) SDES(Source description items)—RTP源描述项
  - 包含标识RTP源的标识符，称为“规范名称(canonical name，简写为CNAME)”。
- (4) BYE(Goodbye)—再见；
- (5) APP(Application-specific functions)—特定应用功能。

# 20.3 实时流播协议（RTSP)

- **实时流播协议(Real-Time Streaming Protocol，RTSP)**
  - 在应用层用来控制RTP会话的协议，用于控制实时多媒体数据在网上的传输
  - 可为客户端的媒体播放器提供远程控制功能，如暂停、快播和从头开始播放。
  - 大多数情况下，RTSP使用TCP协议传送播放器的控制消息，使用UDP协议传送视听数据。

# 20.3 实时流播协议

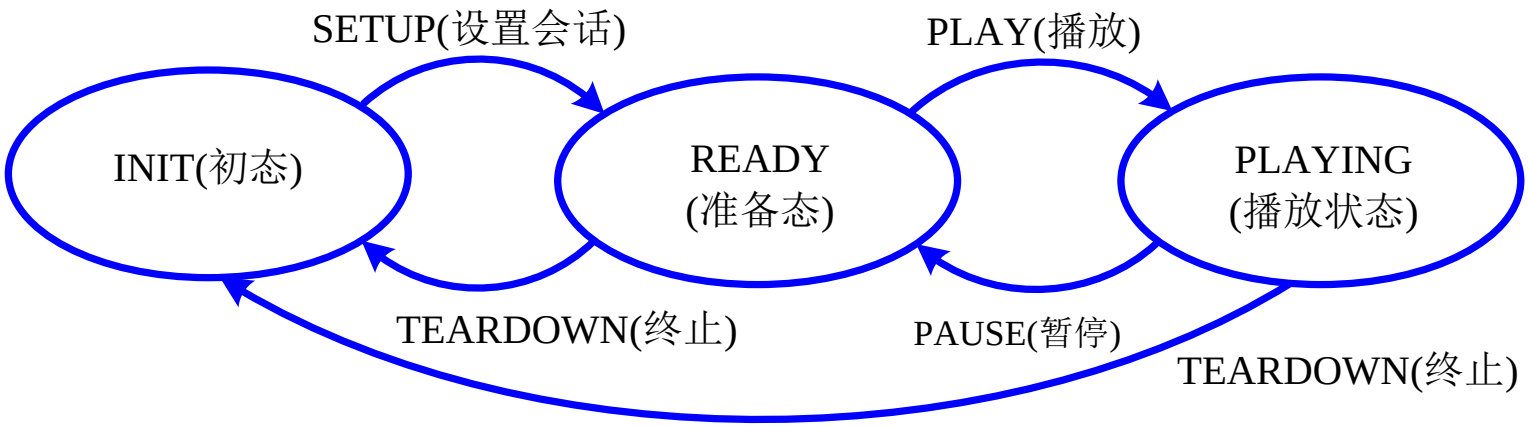
- RTSP协议定义了控制媒体流的方法(method)，包括
  - (1) SETUP(设置)：服务器为媒体流配置资源(如存放媒体流的URL)，并启动RTSP会话
  - (2) PLAY(播放)：根据SETUP设置的资源启动数据传输，开始播放一个或多个媒体流
  - (3) PAUSE(暂停): 暂停播放一个或多个媒体流，但不释放服务器资源
  - (4) TEARDOWN(终止)：用于终止会话，释放与流播有关的所有资源。
  - (5) DESCRIBE(描述)：描述视听媒体流。
  - (6) RECORD(录制)：启动流媒体录制功能；
  - (7) ANNOUNCE(通告)：改变媒体对象的描述；
  - (8) REDIRECT(重定向)：告诉客户需要连接到另一个服务器地址

# 20.3 实时流播协议

- **RTSP协议原理**
  - 执行RTSP协议的程序实际上就是维护客户机和服务机的状态。
  - 客户机和服务机都有三个状态
    - (1) INIT(初态)：在客户机和服务机之间没有RTSP会话
    - (2) READY(准备态)：创建RTSP会话，准备传输数据
    - (3) PLAYING(播放态)：传输和播放流媒体

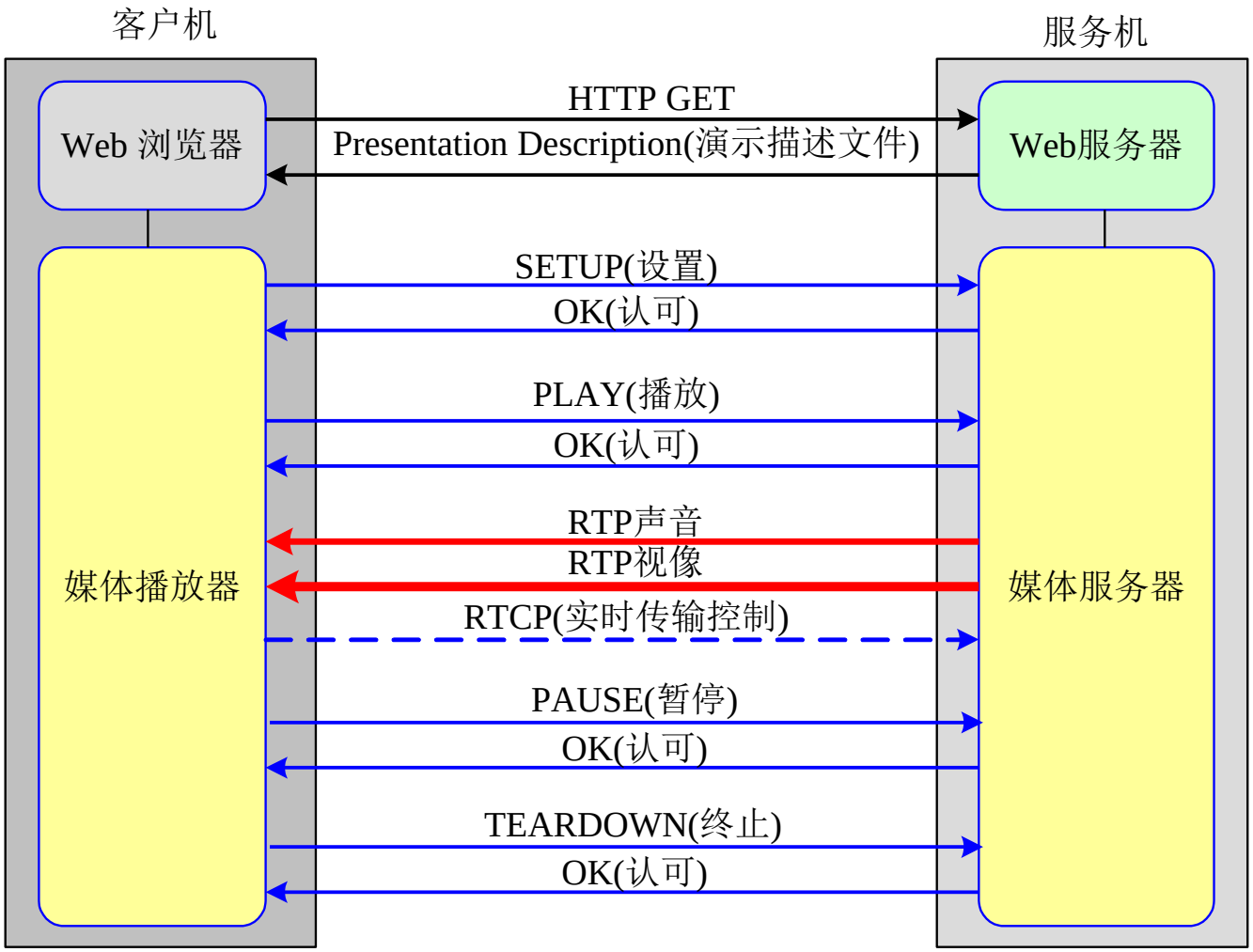
    这些状态之间的转换是通过执行各种方法实现的

# 20.3 实时流播协议


SETUP(设置会话)
PLAY(播放)
INIT(初态)
READY
(准备态)
PLAYING
(播放状态)
TEARDOWN(终止)
PAUSE(暂停)
TEARDOWN(终止)

# 20.3 实时流播协议

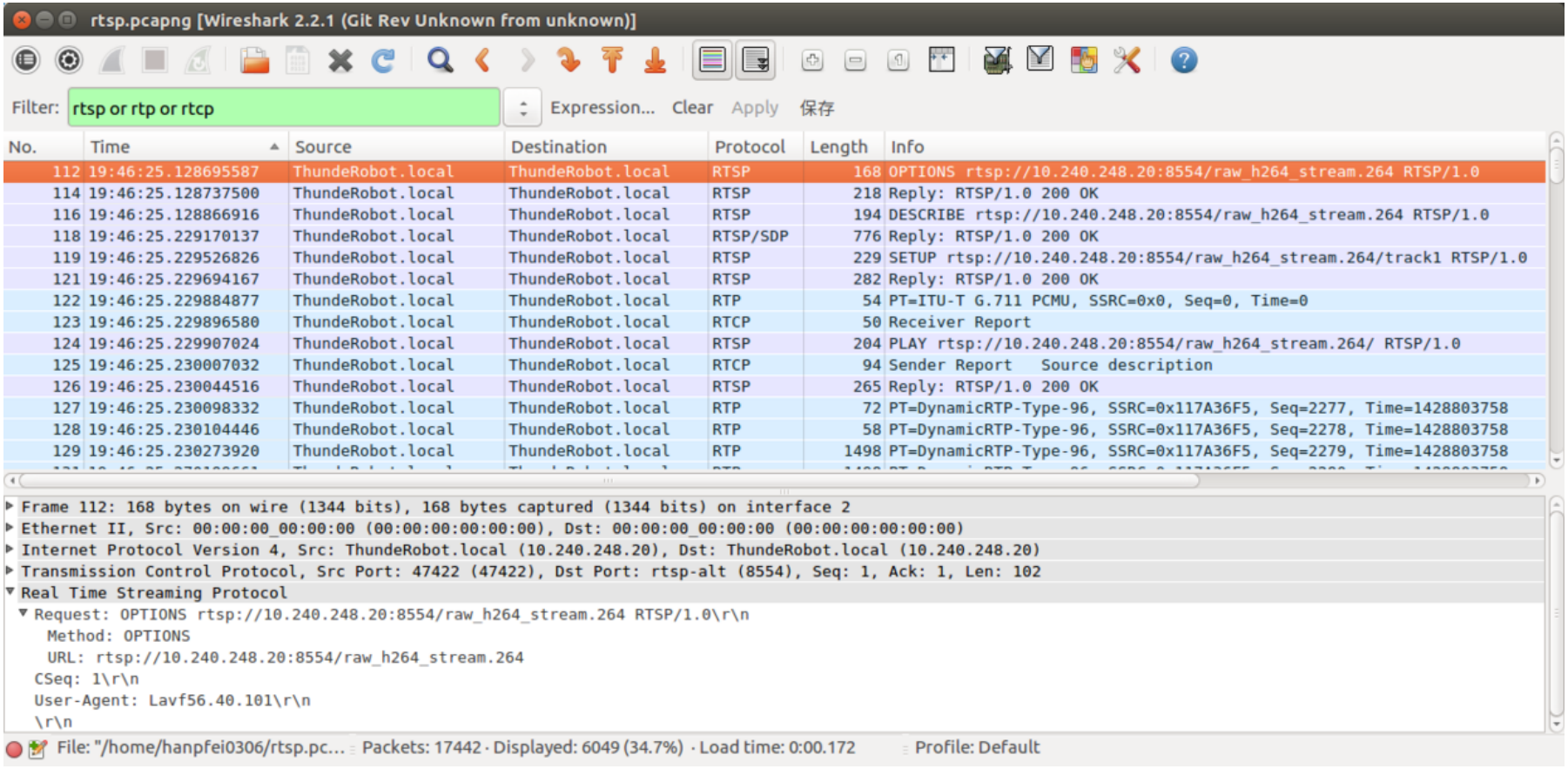
“边流边播”



# 20.3 实时流播协议

- (1) 用户通过Web浏览器向Web服务器发送HTTP GET消息，请求提供视听媒体，而Web服务器把描述媒体流的“演示描述(presentation description)”文件发送给Web浏览器
- (2) Web浏览器得到响应后打开媒体播放器，并将描述文件转发给媒体播放器
- (3) 媒体播放器向媒体服务器发送SETUP(设置)请求消息
- (4) 媒体播放器得到媒体服务器的响应后发送PLAY请求消息
- (5) 媒体服务器发送认可消息，并用RTP/RTCP向媒体播放器发送视听数据
- (6) 如果媒体播放器向媒体服务器发送暂停PAUSE请求，媒体服务器就暂停传输数据
- ……
- (n) 媒体播放器发送TEARDOWN请求，终止RTSP会话

# Wireshark 抓包分析 RTSP/RTP/RTCP




启动 Wireshark 抓包。然后通过 ffplay 请求 live555MediaServer 并播放 raw_h264_stream.264, 通过 Display Filter 过滤仅显示 RTSP/RTP/RTCP 包

首先是 RTSP 数据的交互，建立媒体传输会话，随后开始通过 RTP/RTCP 传输数据。

来自csdn hanpei's blog

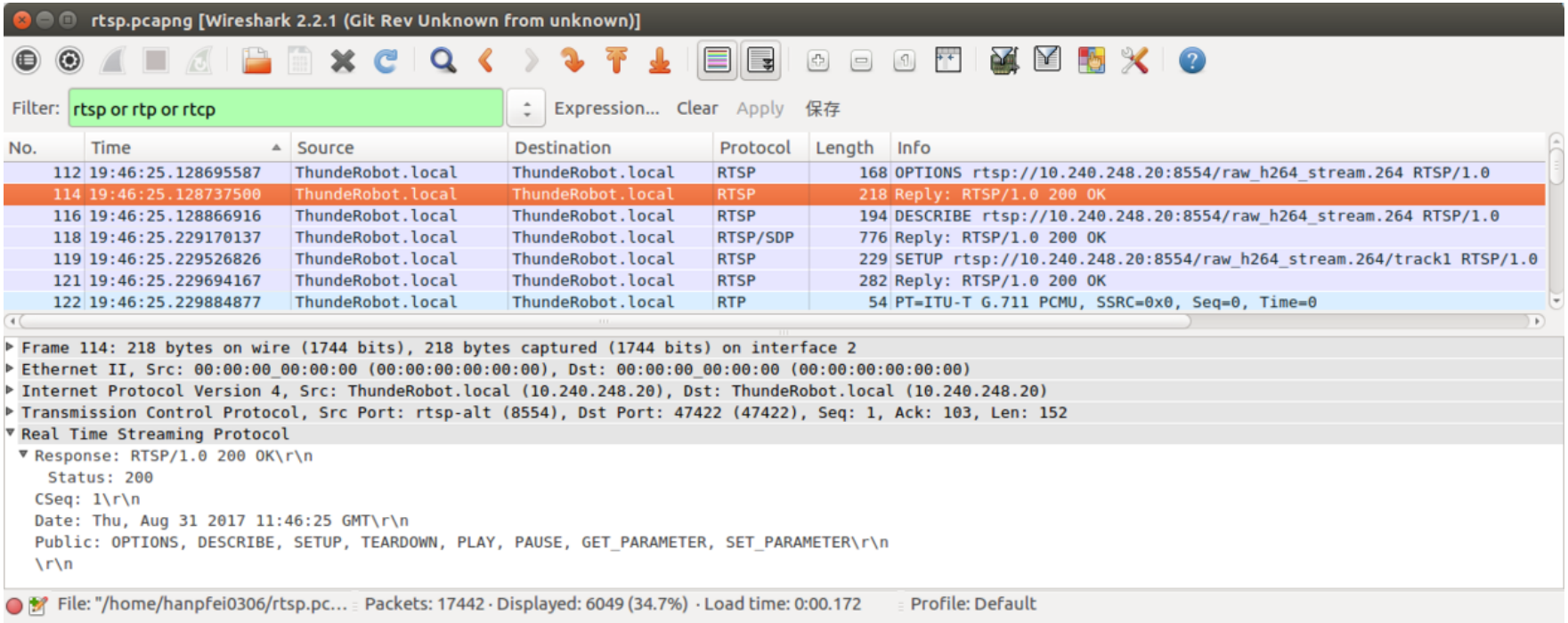
# Wireshark 抓包分析 RTSP/RTP/RTCP

## RTSP 具体定义的方法

| method | direction | object | requirement |
| --- | --- | --- | --- |
| DESCRIBE | C->S | P,S | recommended |
| ANNOUNCE | C->S, S->C | P,S | optional |
| GET_PARAMETER | C->S, S->C | P,S | optional |
| OPTIONS | C->S, S->C | P,S | required |
| | | | (S->C: optional) |
| PAUSE | C->S | P,S | recommended |
| PLAY | C->S | P,S | required |
| RECORD | C->S | P,S | optional |
| REDIRECT | S->C | P,S | optional |
| SETUP | C->S | S | required |
| SET_PARAMETER | C->S, S->C | P,S | optional |
| TEARDOWN | C->S | P,S | required |

# Wireshark 抓包分析 RTSP/RTP/RTCP

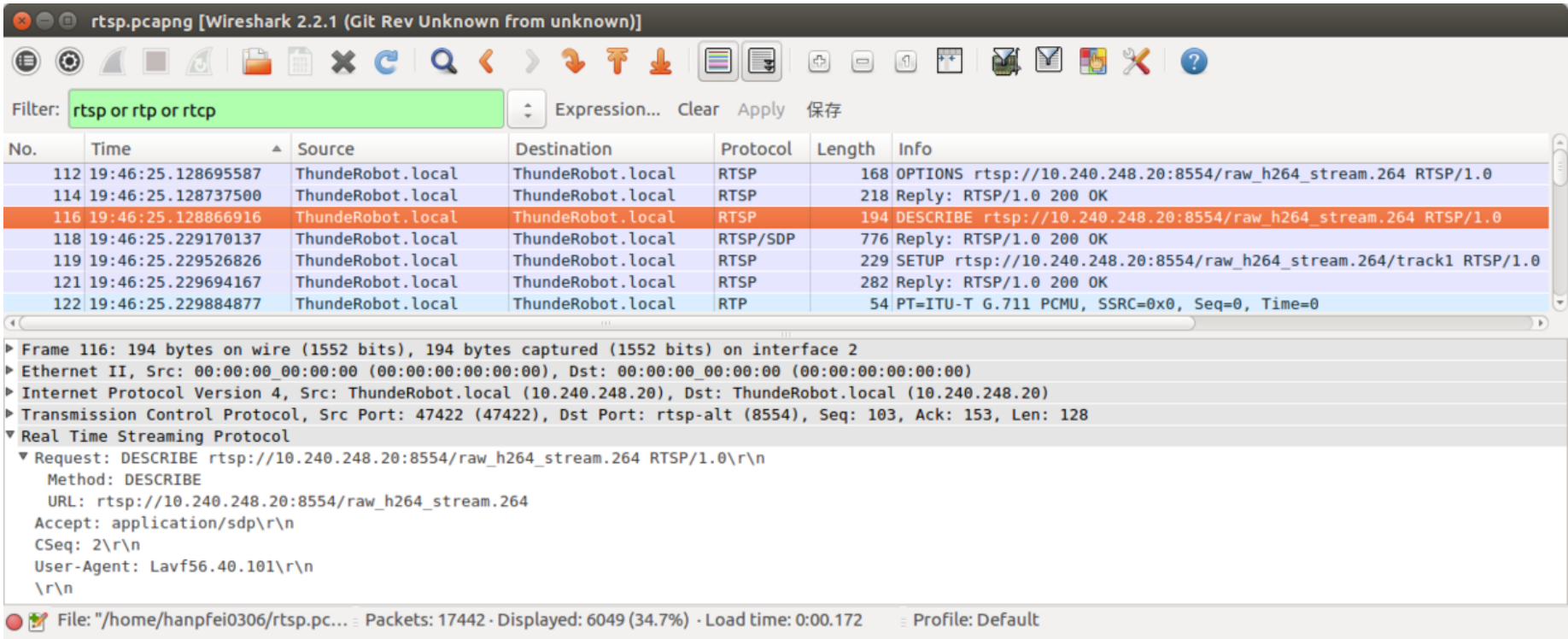
1.客户端首先向服务器发送了一个方法为 OPTIONS 的请求，如第 112 号包，携带有 URL，RTSP 版本号，User-Agent 等信息。




2.服务器将该 URL 支持的方法的列表返回给客户端，即OPTIONS, DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE, GET_PARAMETER, SET_PARAMETER

# Wireshark 抓包分析 RTSP/RTP/RTCP

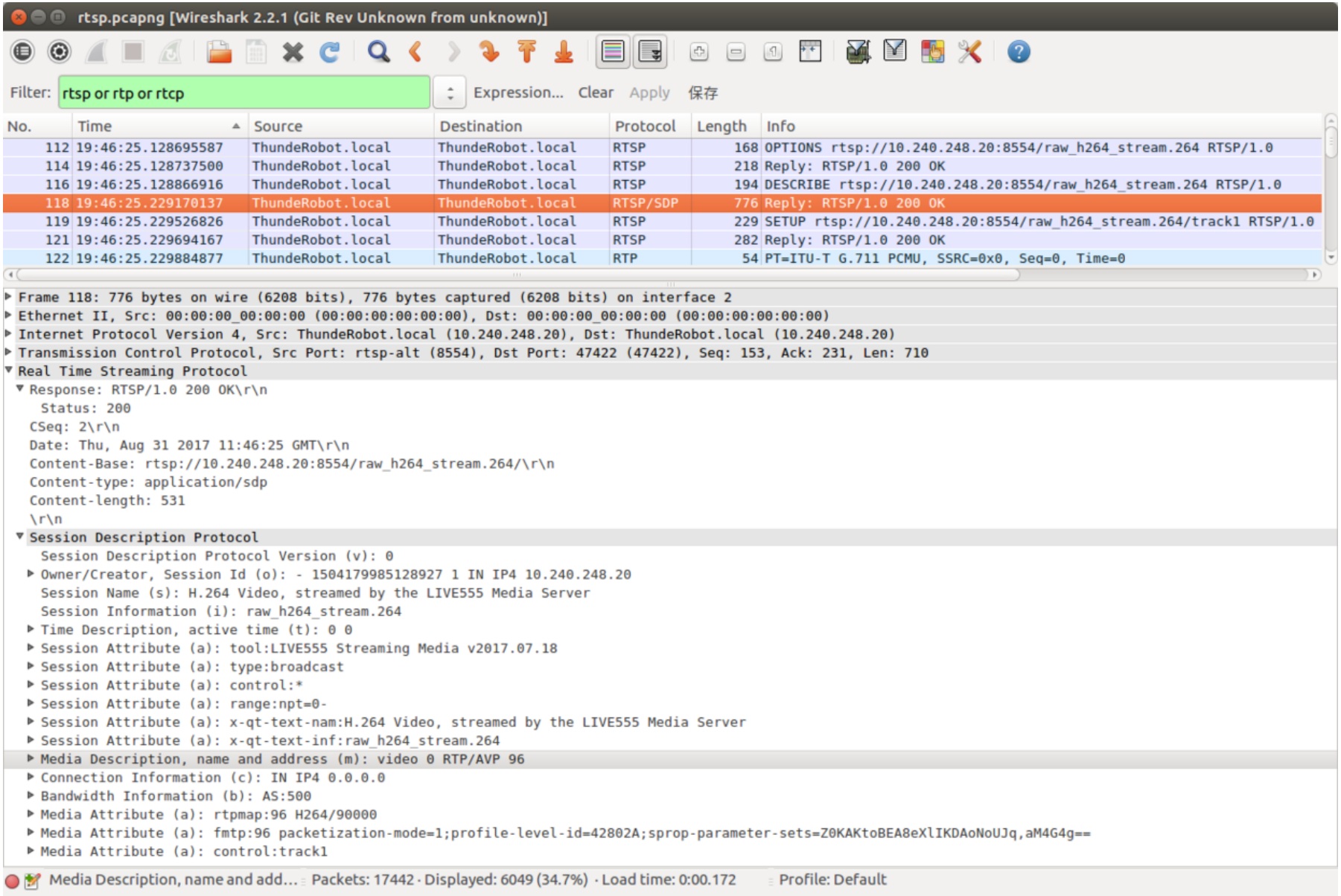
3.客户端向服务器发送了一个 DESCRIBE 请求，即第 116 号包




DESCRIBE 请求的 Accept 头部值为 application/sdp，表示客户端希望收到 SDP 格式的媒体表示。

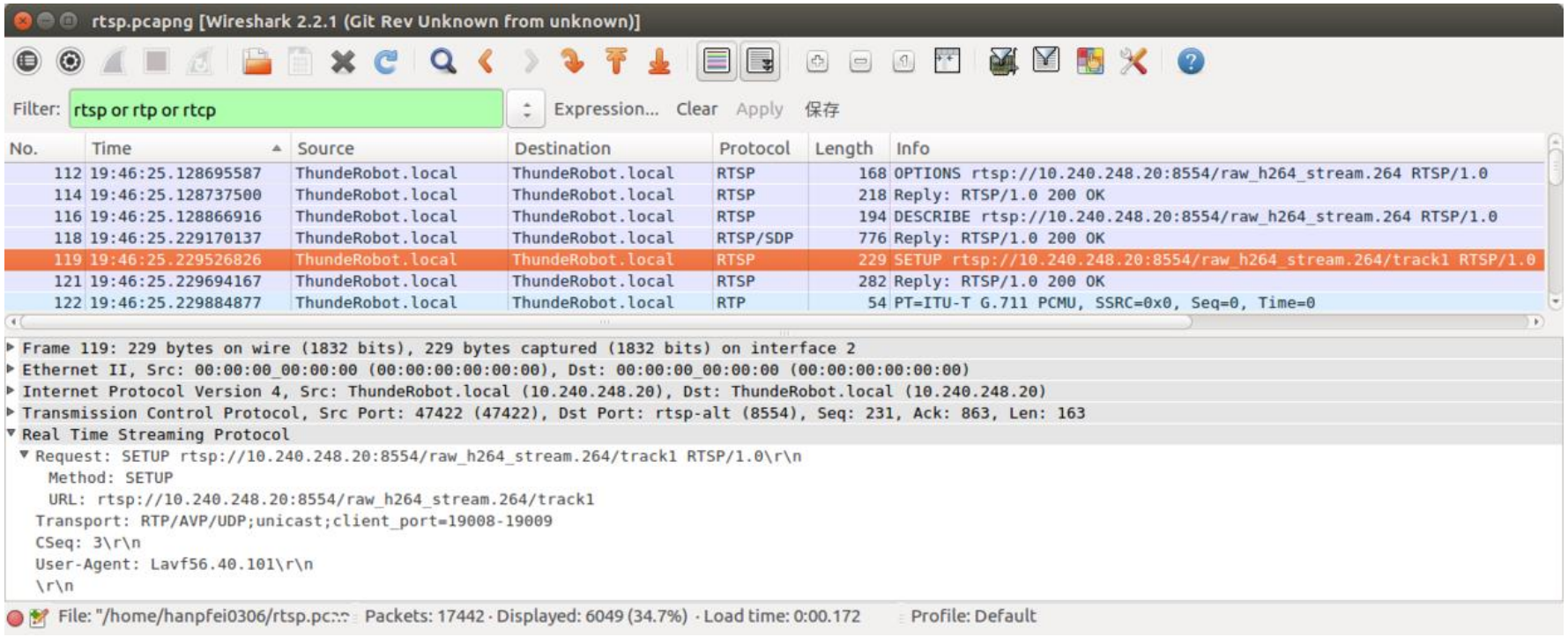
4.服务器以一个 RTSP/SDP 包作为响应

# Wireshark 抓包分析 RTSP/RTP/RTCP



# Wireshark 抓包分析 RTSP/RTP/RTCP

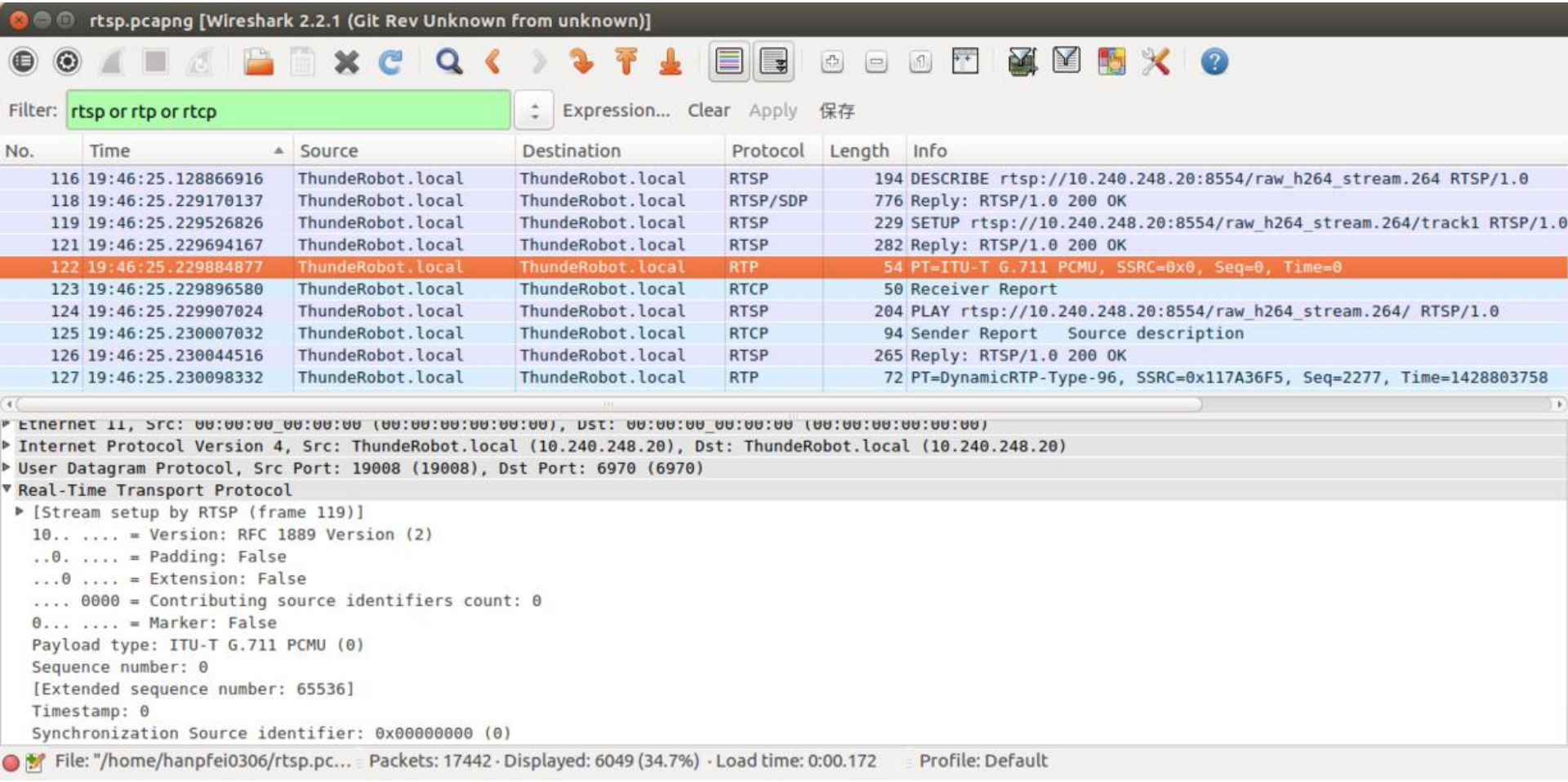
5. 客户端在收到服务器发来的 SDP 包之后，会选择两个端口，分别用于 RTP 和 RTCP 包的收发，并发送了一个 SETUP 请求用于建立媒体会话



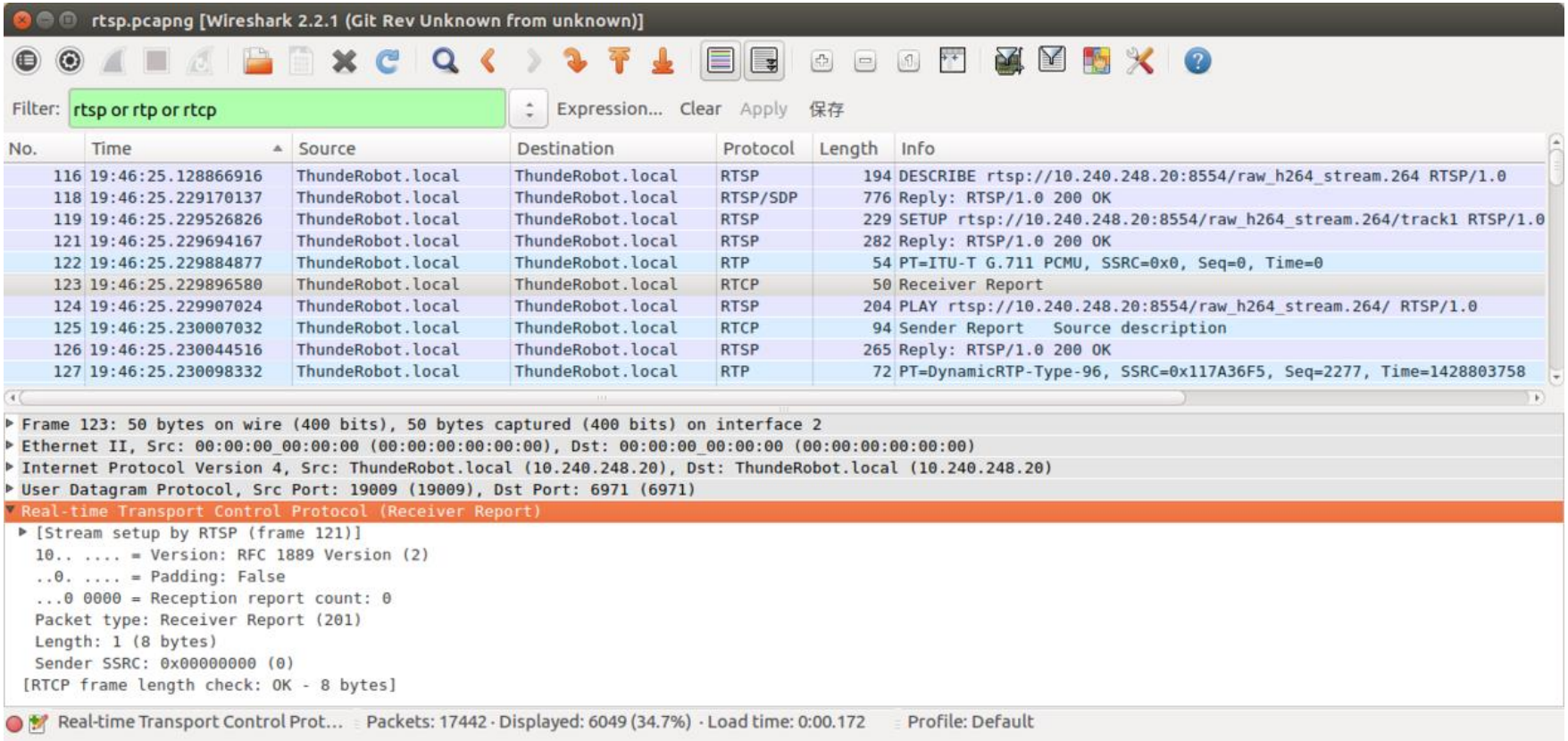

6. 服务器对 SETUP 请求做出了响应，如第121 号包，把它为媒体会话开启的用于收发 RTP、RTCP 包的端口，会话的标识符，超时时间等信息通知给客户端

# Wireshark 抓包分析 RTSP/RTP/RTCP

7. 客户端分别在 RTP 和 RTCP 的端口上，向服务器的 RTP 和 RTCP 端口上发送了两个包，如第 122 号包和第 123 号包

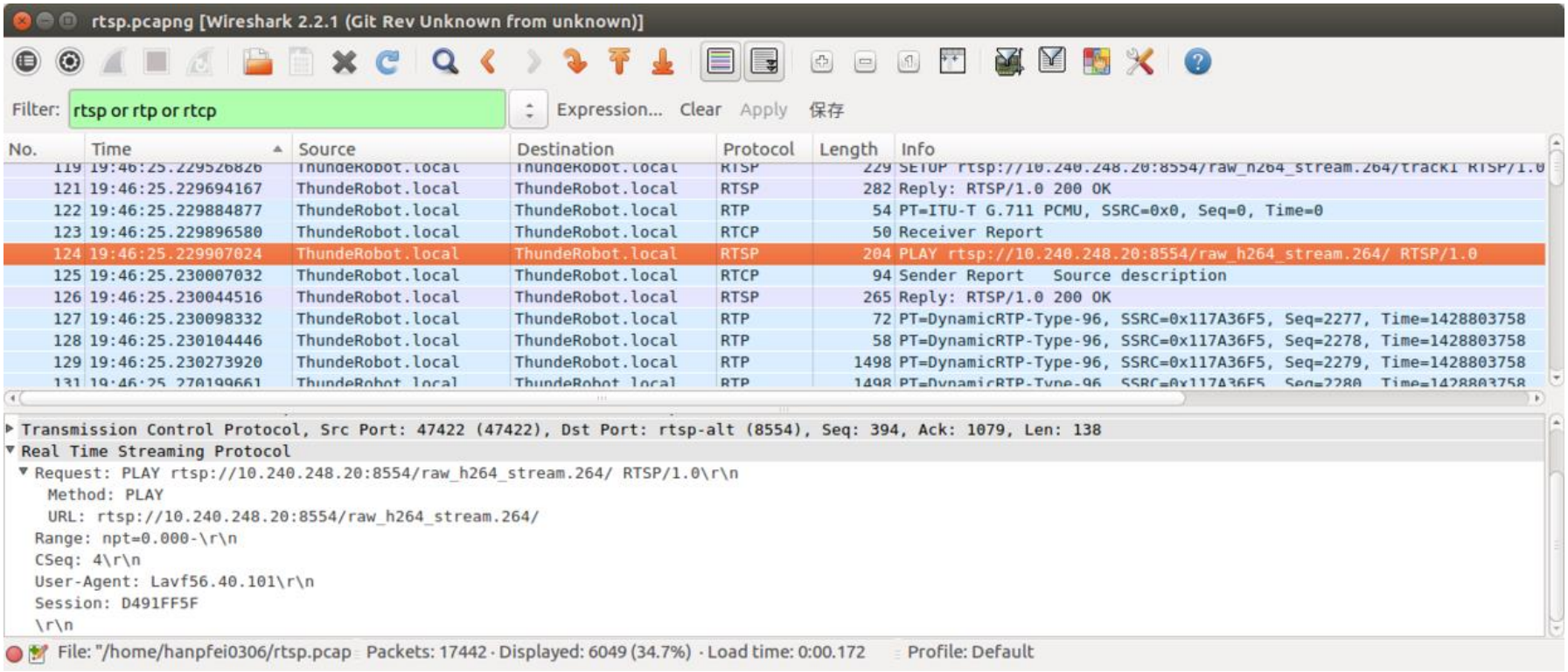


# Wireshark 抓包分析 RTSP/RTP/RTCP



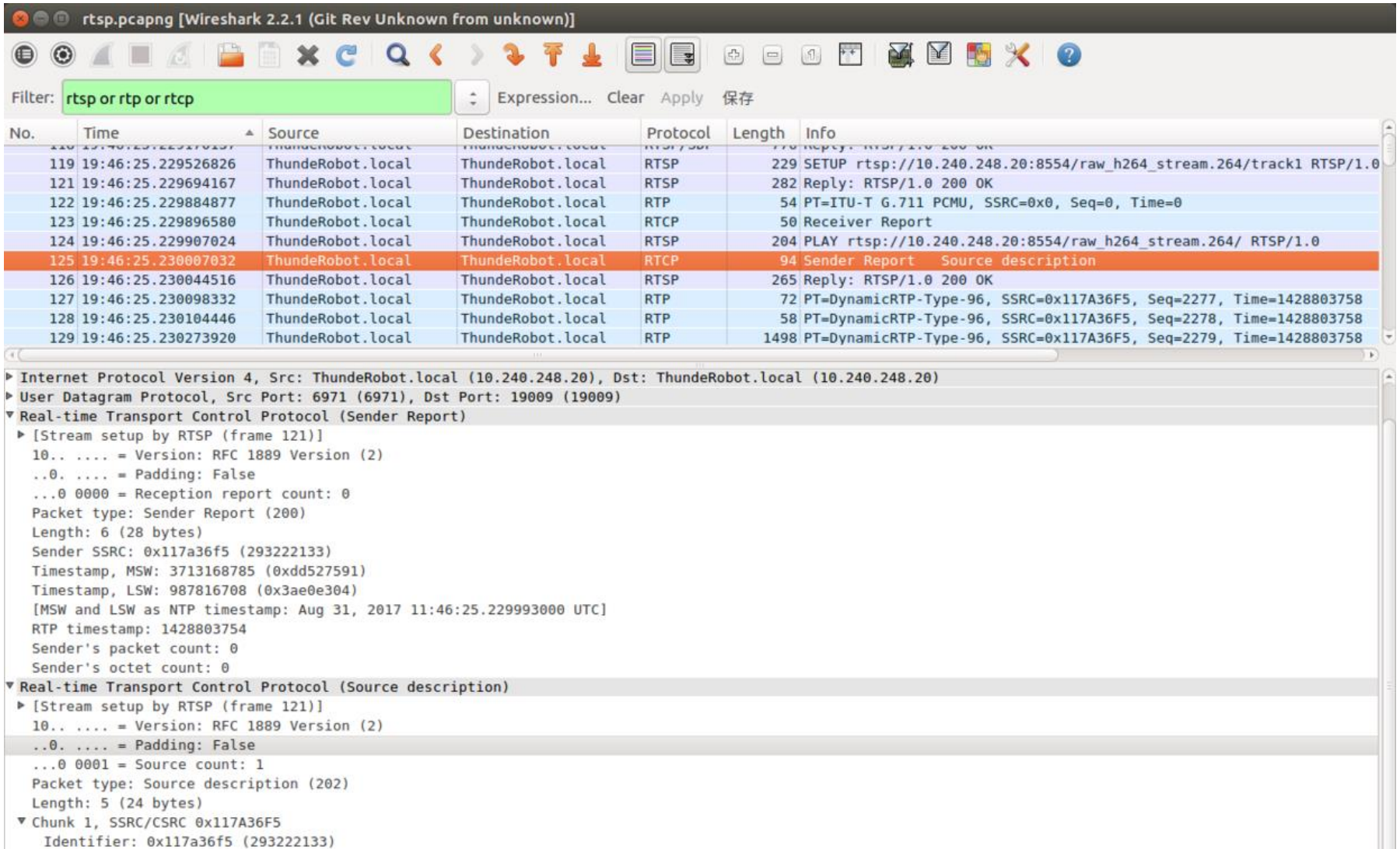
# Wireshark 抓包分析 RTSP/RTP/RTCP

8.客户端向服务器发送了一个 PLAY 请求，来启动播放



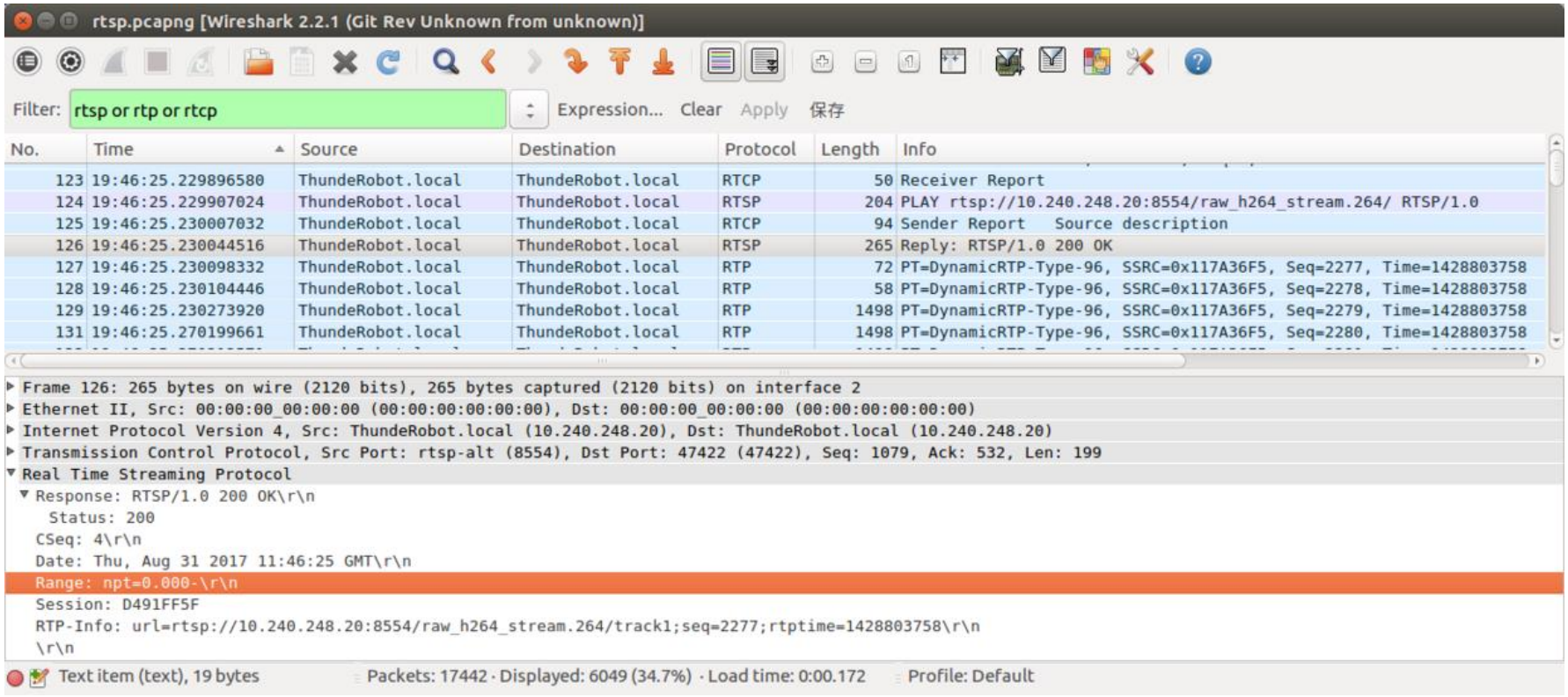
# Wireshark 抓包分析 RTSP/RTP/RTCP

9.服务器向客户端发送了一个 RTCP 包，把 RTP 时间戳，服务器的 SSRC，服务器的 CNAME 等信息发送给客户端



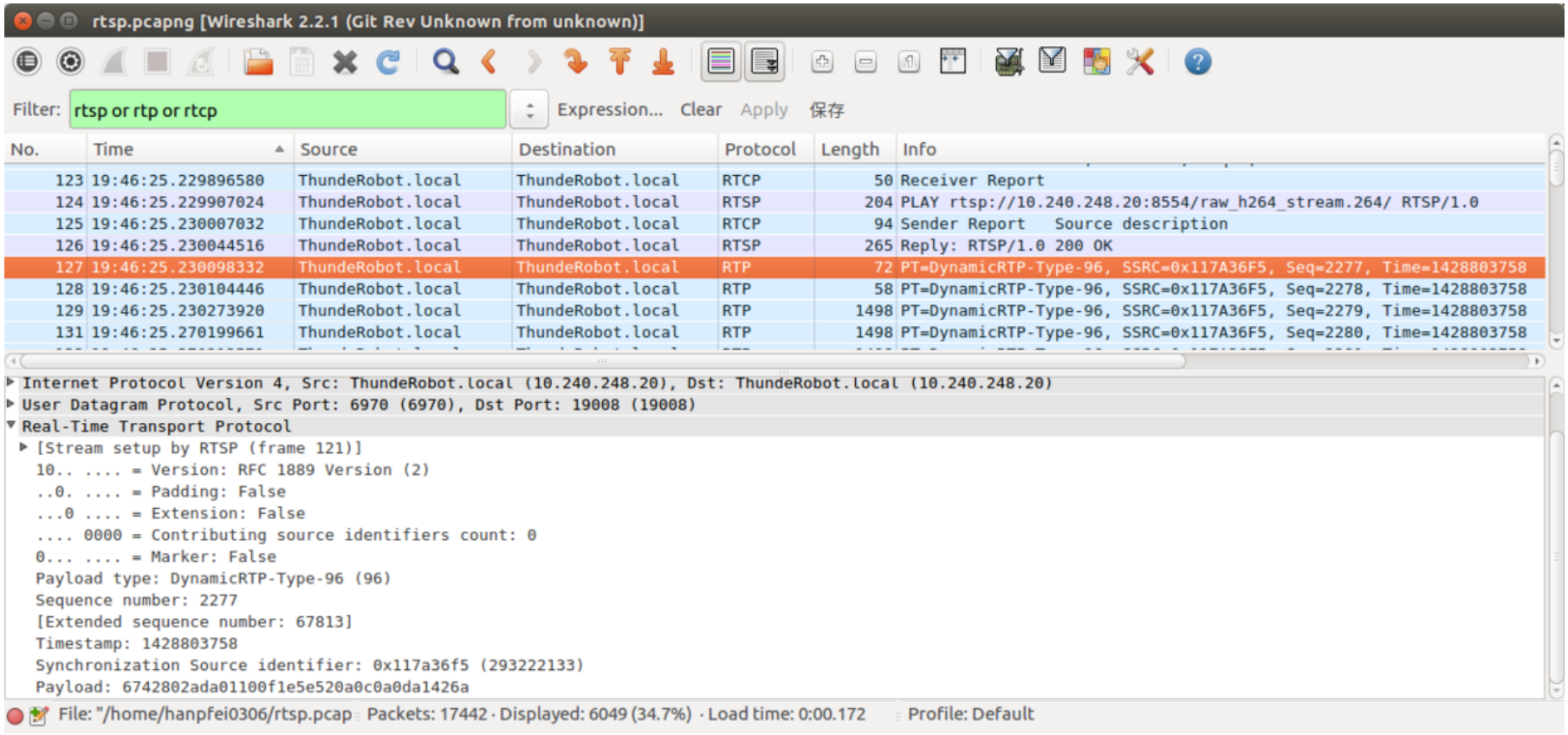
# Wireshark 抓包分析 RTSP/RTP/RTCP

10. 服务器发送 PLAY 请求的响应，其中包含 RTP 包的初始序列号，RTP 时间等重要信息。至此媒体会话最终建立完成。

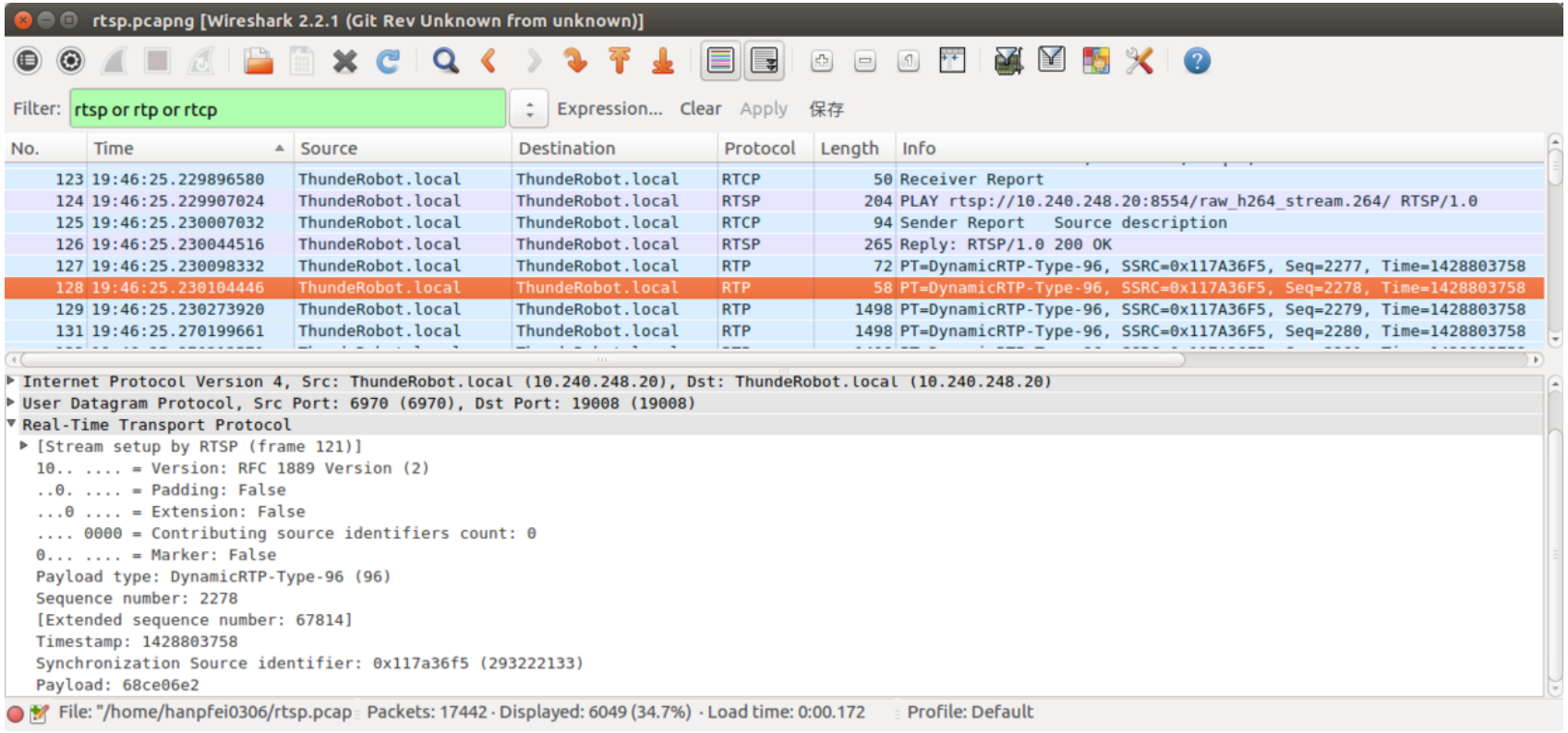


# Wireshark 抓包分析 RTSP/RTP/RTCP

11. 通过 RTP/RTCP 发送流媒体数据。



# Wireshark 抓包分析 RTSP/RTP/RTCP

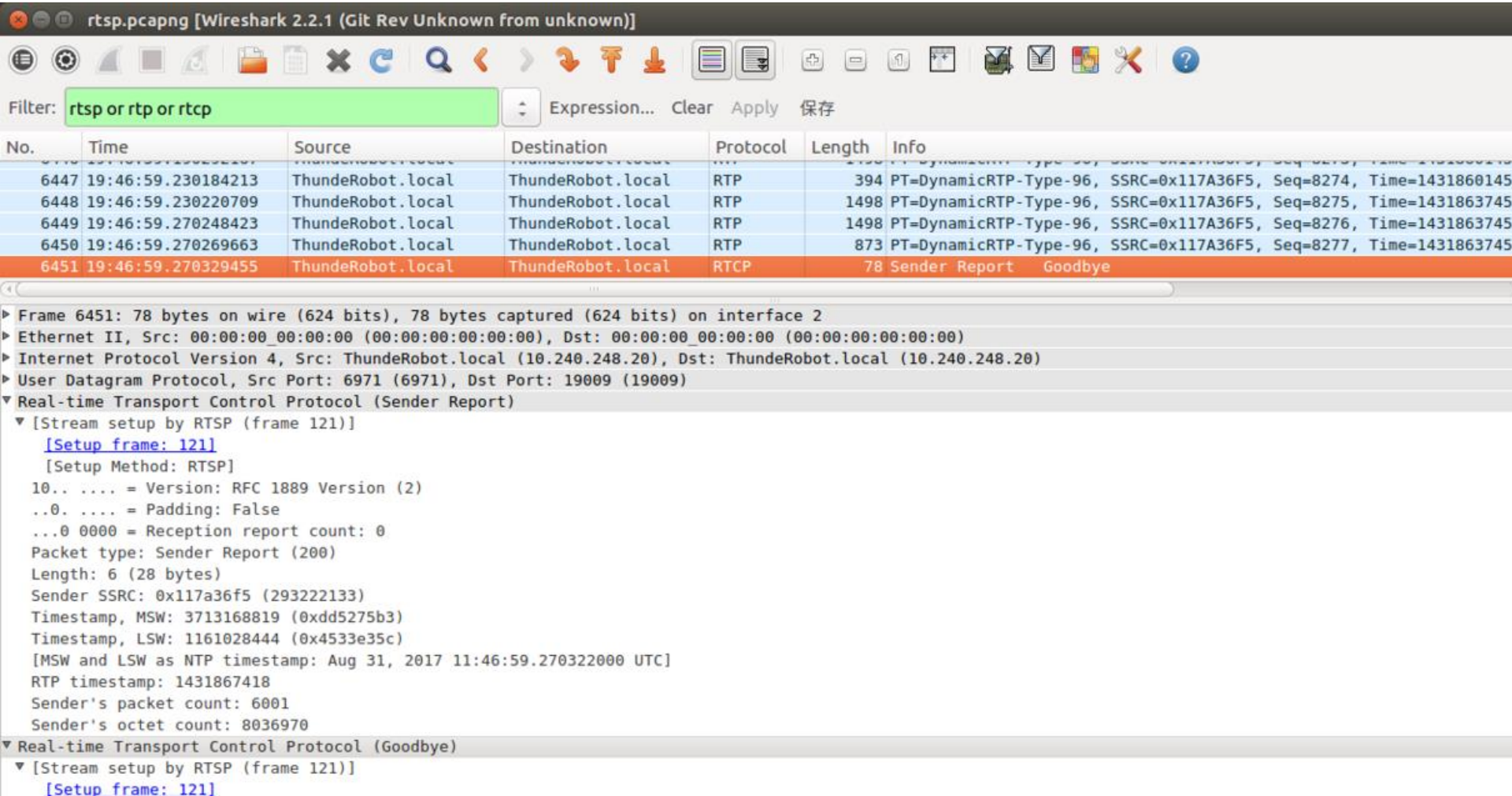



127，128号包的内容与 H.264 视频文件的前两个 NALU 的内容完全吻合。

从 RTSP 的 OPTIONS 请求开始，到首个视频数据 NALU 开始发送，经过了总共大概 102 ms 的时间，媒体会话完全建立完成。

# Wireshark 抓包分析 RTSP/RTP/RTCP

12.视频数据经过一端时间的稳定传输，最终以服务器向客户端发送的一个 RTCP BYE 包而结束，如第 6451 号包：



# Wireshark 抓包分析 RTSP/RTP/RTCP

总结一下这个过程：

1. 客户端首先向服务器发送一个方法为 OPTIONS 的请求，了解服务器为 URL 提供了哪些方法的支持。

2. 服务器将该 URL 支持的方法的列表返回给客户端。

3. 客户端向服务器发送了一个 DESCRIBE 请求，提取由所请求的 URL 标识的表示或媒体对象的描述信息。

4. 服务器通过 SDP 包，告知流媒体数据传输所用的协议，以及流媒体本身的一些信息。

5. 客户端在收到服务器发来的 SDP 包之后，会选择两个端口，分别用于 RTP 和 RTCP 包的收发，并发送了一个 SETUP 请求用于建立媒体会话。

6. 服务器发回 SETUP 响应，把它为媒体会话开启的用于收发 RTP、RTCP 包的端口，会话的标识符，超时时间等信息通知给客户端

# Wireshark 抓包分析 RTSP/RTP/RTCP

7. 客户端分别在 RTP 和 RTCP 的端口上，向服务器的 RTP 和 RTCP 端口上发送了两个包。

8. 客户端向服务器发送一个 PLAY 请求，来启动播放。

9. 服务器向客户端发送一个 RTCP 包，把 RTP 时间戳，服务器的 SSRC，服务器的 CNAME 等信息发送给客户端。

10. 服务器发送 PLAY 请求的响应，其中包含 RTP 包的初始序列号，RTP 时间等重要信息。至此媒体会话最终建立完成。

11. 通过 RTP/RTCP 发送流媒体数据。

12. 服务器向客户端发送一个 RTCP BYE 包结束会话。

# END

## 第20章 多媒体的传输