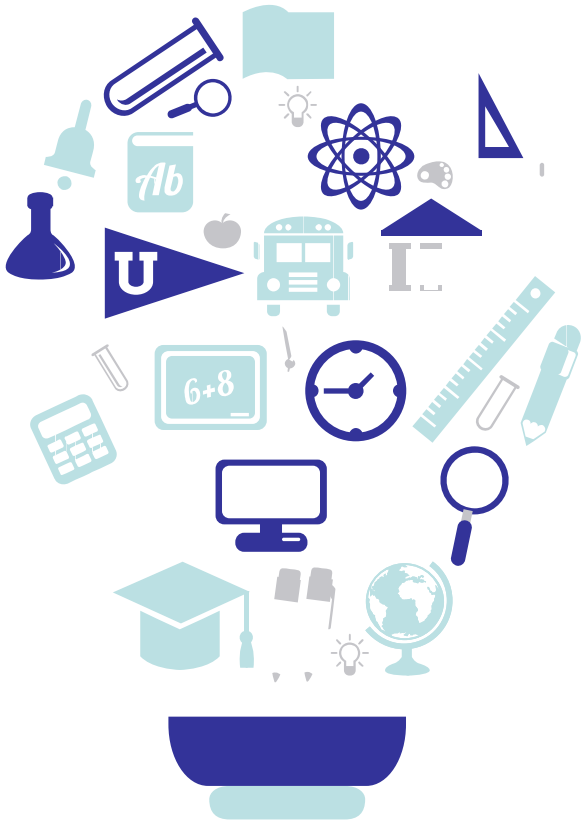
本课讲师：Darren老师

20-RTSP(中)-RTP(实时传输协议)



大纲

RTP 背景和概述

RTP封装协议

RTP关键技术

RTP协议运用

RTP打包

RTP协议背景及概述

- **流（Streaming）**是近年在Internet上出现的新概念，其定义非常广泛，主要是指通过网络传输多媒体数据的技术总称。

- 流式传输分为两种
  - ✓ 顺序流式传输（Progressive Streaming)
  - ✓ 实时流式传输（Real time Streaming)

- **实时流式传输**是实时传送，特别适合现场事件。“实时”是指在一个应用中数据的交付必须与数据的产生保持精确的时间关系，这需要相应的协议支持，这样RTP和RTCP就相应的出现了。

RTP协议背景及概述

- ✓ **RTP全名：Real-time Transport Protocol（实时传输协议）**。它是IETF提出的一个标准，对应的RFC文档为RFC3550。
- ✓ RFC3550定义了RTP，也定义了配套的实时传输控制协议RTCP（Real-time Transport Control Protocol）。
- ✓ RTP用来为IP网上的语音、图像、传真等多种需要实时传输的多媒体数据提供端到端的实时传输服务。
- ✓ RTP为Internet上端到端的实时传输提供时间信息和流同步，但并不保证服务质量，服务质量由RTCP来提供。

RTP协议背景及概述

- **RTP协议原理**：较简单，负责对流媒体数据进行封包并实现媒体流的实时传输，即它按照RTP数据包格式来封装流媒体数据，并利用与它绑定的协议进行数据包的传输。

- **RTCP原理**：向会话中的所有成员周期性地发送控制包来实现的，应用程序通过接收这些控制数据包，从中获取会话参与者的相关资料，以及网络状况、分组丢失概率等反馈信息，从而能够对服务质量进行控制或者对网络状况进行诊断。

RTP协议背景及概述

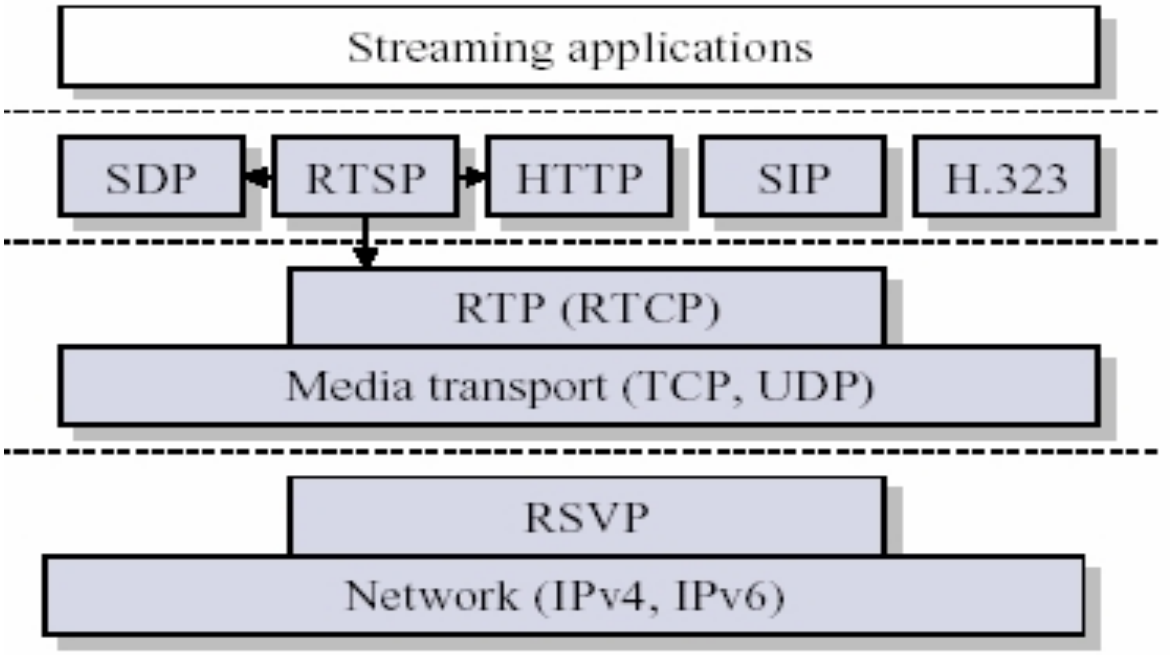
- ✓ RTP在端口号1025到65535之间选择一个未使用的偶数UDP端口号，而在同一次会话中的RTCP则使用下一个基数UDP端口号。
- ✓ 一般RTP和RTCP端口分开来的，但是**webrtc**用了同样的端口。
- ✓ jrtp base_port

- ✓ 默认端口号：
- RTP：5004
- RTCP：5005

RTP协议背景及概述

- 从下图可看出RTP被划分在传输层，它建立在UDP上。同UDP协议一样，为了实现其实时传输功能，RTP也有固定的封装形式。RTP用来为端到端的实时传输提供时间信息和流同步，但并不保证服务质量。服务质量由RTCP来提供。

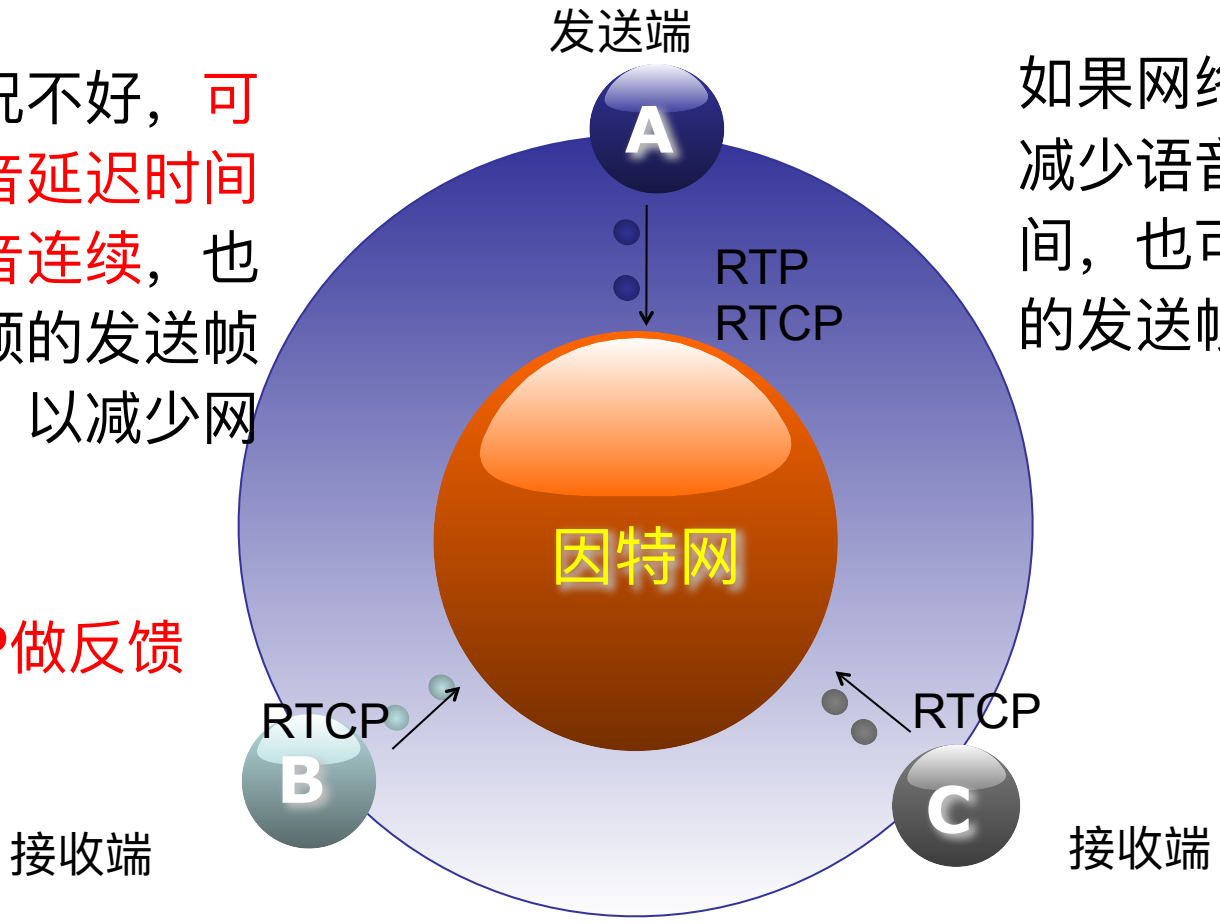
- H264 AAC



RTP协议背景及概述

若网络状况不好，可以增大语音延迟时间以保证语音连续，也可减少视频的发送帧率或质量，以减少网络的阻塞。

需要**RTCP**做反馈



如果网络情况好,可以减少语音的延迟时间，也可以增大视频的发送帧率或质量。

RTP/RTCP工作原理

RTP 背景和概述

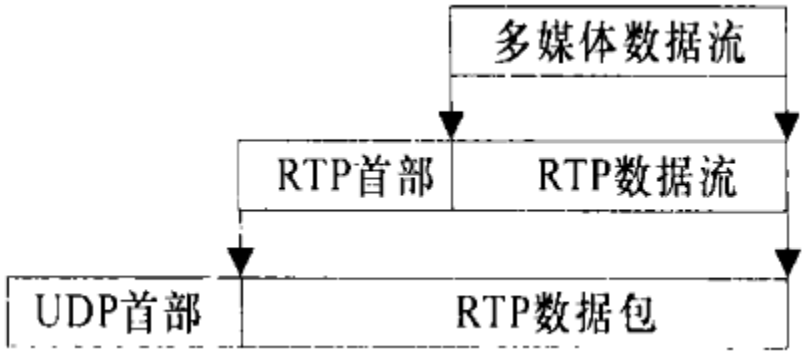
RTP封装协议

RTP关键技术

RTP协议运用

RTP打包

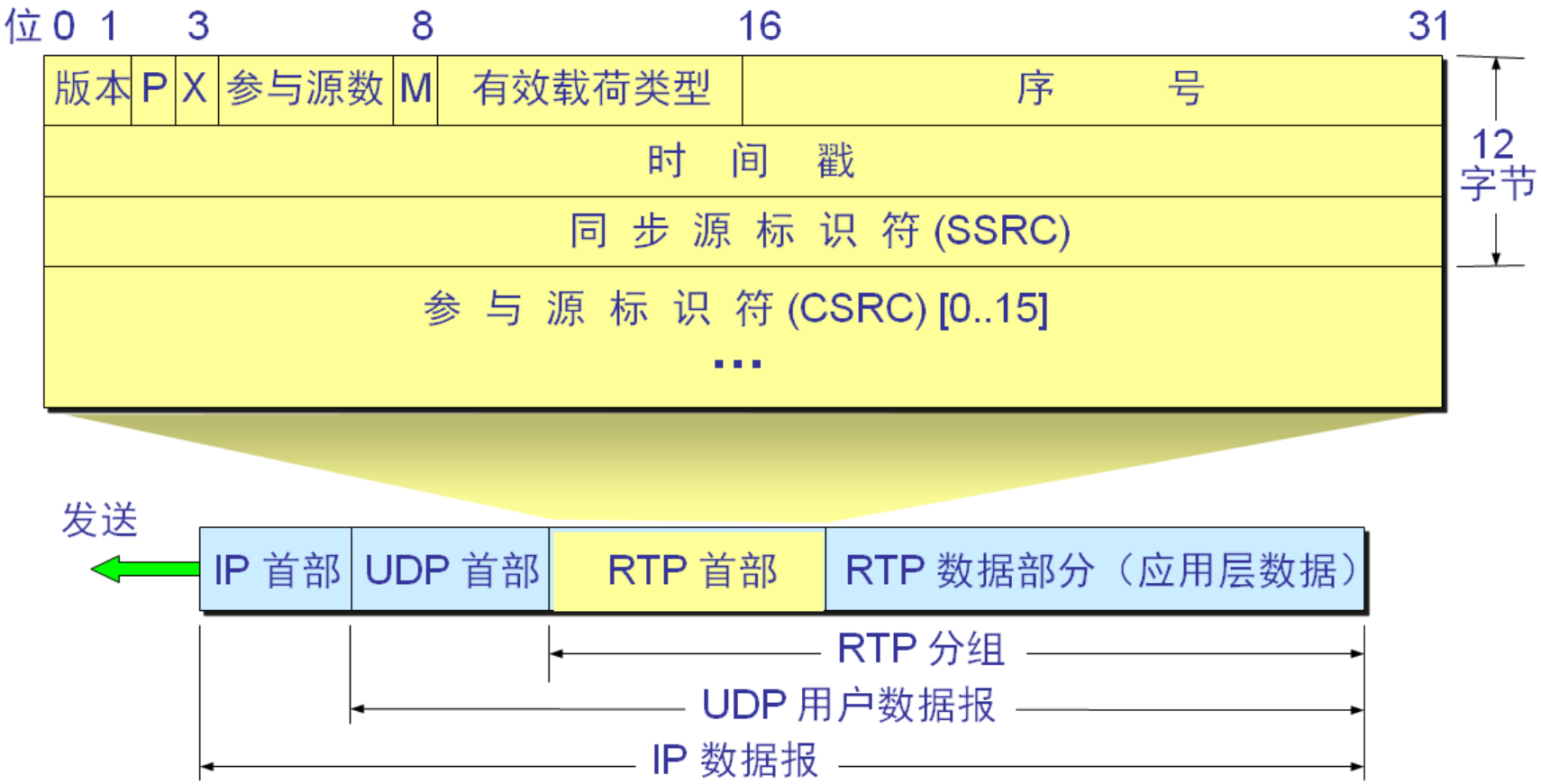
RTP协议封装



从开发者的角度看，RTP 应当是应用层的一部分。

RTP 最大的数据包（包含RTP头部） = MTU - UDP首部 – IP报文首部

RTP协议封装


位 0 1 3 8 16 31
版本
P
X
参与源数
M
有效载荷类型
序 号
时 间 戳
同 步 源 标 识 符 (SSRC)
参 与 源 标 识 符 (CSRC) [0..15]
...
12 字节
发送
IP 首部
UDP 首部
RTP 首部
RTP 数据部分（应用层数据）
RTP 分组
UDP 用户数据报
IP 数据报

RTP Header

1. 版本V：RTP协议的版本号，占2位，当前协议版本号为2

2. P：填充标志，占1位，如果P=1，则在该报文的尾部填充一个或多个额外的八位组，它们不是有效载荷的一部分。

3. X：扩展标志，占1位，如果X=1，则在RTP报头后跟有一个扩展报头

4. 参与源数CC：CSRC计数器，占4位，指示CSRC 标识符的个数

5. **M:** 标记，占**1**位，不同的有效载荷有不同的含义，对于视频，标记一帧的结束；对于音频，标记会话的开始。

6. **PT:** 有效荷载类型，占7位，用于说明RTP报文中有效载荷的类型，如GSM音频、JPEM图像等,在流媒体中大部分是用来区分音频流和视频流的，这样便于客户端进行解析。

RTP Header 音视频是独立的RTP通道

7. 序列号：占16位，用于标识发送者所发送的RTP报文的序列号，每发送一个报文，序列号增1。这个字段当下层的承载协议用UDP的时候，网络状况不好的时候可以用来检查丢包。同时出现网络抖动的情况可以用来对数据进行重新排序，序列号的**初始值是随机的**，同时音频包和视频包的sequence是分别记数的。

8. 时戳(Timestamp)：占32位，推荐使用90kHz 时钟频率。时戳反映了该RTP报文的第一个八位组的采样时刻。接收者使用时戳来计算延迟和延迟抖动，并进行同步控制。

9. **同步信源(SSRC)标识符**：占32位，用于标识同步信源。该标识符是随机选择的，参加同一视频会议的两个同步信源不能有相同的SSRC。

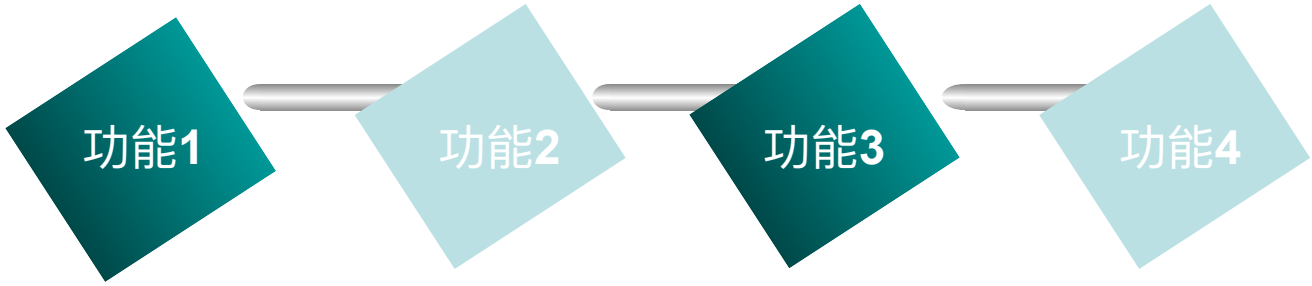
10. 特约信源(CSRC)标识符：每个CSRC标识符占32位，可以有0～15个。每个CSRC标识了包含在该RTP报文有效载荷中的所有特约信

RTP Header

- 有效载荷类型：7位，指出后面的RTP数据属于何种格式的应用。应用层可根据次类型进行处理。

  - 音频：AAC(99) G.726(100)
  - 视频：H264(96)

RTCP的主要功能



功能1	功能2	功能3	功能4
提供数据传输质量的反馈	为每个RTP源传输一个固定的识别符CNAME	知晓成员数目，控制包的发送速率	传输最小连接控制信息，如参加者标识

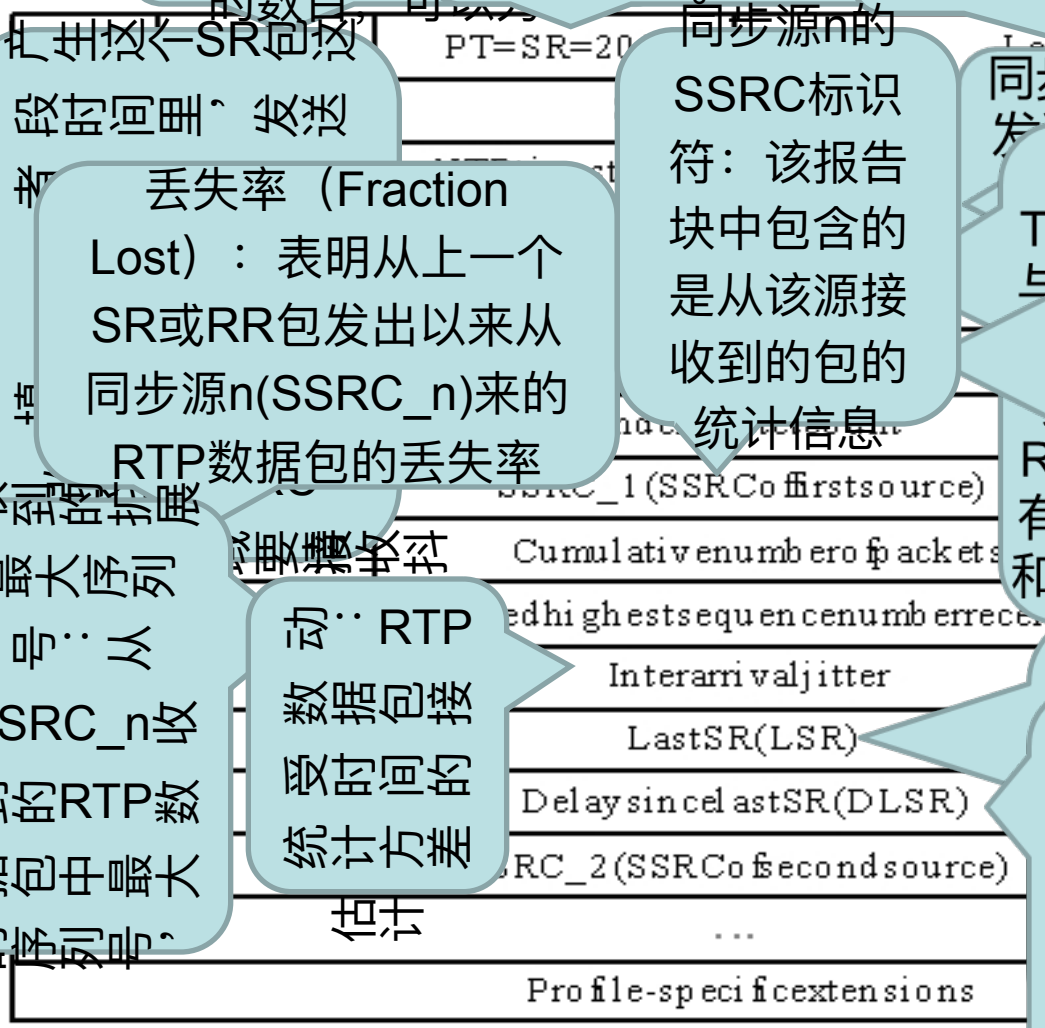
RTCP协议封装

- RTCP也是用UDP来传送的，但RTCP封装的仅仅是一些控制信息，因而分组很短，所以可以将多个RTCP分组封装在一个UDP包中。RTCP有如下**五种分组类型**：

类型	缩写表示	用途
200	SR（Sender Report）	发送端报告
201	RR（Receiver Report）	接收端报告
202	SDES（Source Description Items）	源点描述
203	BYE	结束传输
204	APP	特定应用

RTCP协

- 发



接收报告计数器（RC）：5

SR包中的接

的数值

包类型（PT）：8

比特，SR包是200。

长度域（Length）：16比特，其中存放的是该SR包以32比特为单位的总长度减

从开始发送包到产生这个SR包这段时间里，发送

丢失率（Fraction Lost）：表明从上一个SR或RR包发出以来从同步源n(SSRC_n)来的RTP数据包的丢失率

同步源n的SSRC标识符：该报告块中包含的是从该源接收到的包的统计信息

NTP

RTP

从开始发送包到产生这个SR包这段时间里，发送

累计的包丢失数目：从开始接收到SSRC_n的包

上次SR时间戳：

上次SR以来的延时：上次从SSRC_n收到SR包到发送本报告的延时。

RTP数据包接受的间的统计方差的估计

收到的最大序列号：从SSRC_n收到的RTP数据包中最大的序列号，

RTP 背景和概述

RTP封装协议

RTP关键技术

RTP协议运用

RTP打包

RTP协议关键技术

- 时间戳
- 时延
- 抖动
- 丢包率
- 会话和流两级分用

时间戳（1）

- **时间戳字段**是RTP首部中说明数据包时间的同步信息，是数据能以正确的时间顺序恢复的关键。

- **时间戳的值**给出了分组中数据的第一个字节的采样时间，要求发送方时间戳的时钟是连续、单调增长的（32bit），即使在没有数据输入或发送数据时也是如此。

- 在静默时，发送方不必发送数据，保持时间戳的增长，在接收端，由于接收到的数据分组的序号没有丢失，就知道没有发生数据丢失，而且只要比较前后分组的时间戳的差异，就可以确定输出的时间间隔。

10秒 20秒才有声音，然后发送声音。

时间戳（2）

- RTCP 中的 SR (Sender Report发送端报告）控制分组包含**NTP（网络时间）时间戳和RTP时间戳**可用于同步音视频媒体流。

- RTP时间戳是依据邻近的RTP数据包中的时间戳结合NTP时间差得到的。

- **公式表达**为：**RTP_tsi = tsi + NTPi - NTP'i**

- RTP_tsi表示RTCP中的RTP时间戳；tsi表示邻近的RTP包中的时间戳；NTPi表示RTCP的网络时间戳；NTP'i表示邻近的RTP包对应的网络时间戳；下标表示第i个源。

时间戳（3）

- 因此，**i和源j之间的相对时差**可以表示为：

（RTP_tsi – tsi）-(RTP_tsj - tsj) = (NTPi –NTP'i) - （NTPj—NTP'j）

- 由于NTP同步，差值可以反映出两个源的相对时差。因为要同步不同来源的媒体流，必须使得同步他们的绝对时间基准，而NTP时间戳正是这样的绝对时间基准。

- 应用RTP时间戳来保证同一来源的媒体流同步。

时延

- **影响时延的因素**有多个方面：
  - ✓ 编解码 （编码延迟，B帧数量，参考帧数量）
    - ✓ 丢一帧数据进去编码，是不是可以立即拿到一帧编码后的数据。
  - ✓ 网络
  - ✓ 防抖动缓冲 （接收缓存的大小）
  - ✓ 报文队列
  - ✓ ……

其中有些是**固定时延**，如编解码网络速率等；有些是变化的，如防抖动缓冲等，固定的时延可以通过改变编解码方式和提高网络速率来改变，而**变化时延**通常采用提高转发效率来提高。

抖动

- 到达时刻抖动J的定义：一对包中接收机相对发射机的时间跨度差值的平均偏差。

- 该值等于两个包相对传输时间的差值，相对传输时间是指包的RTP时间标志和到达时刻接收机时钟，以同一单位的差值.若Si是包i的RTP时间标志，Ri是包i以RTP时间标志单位的到达时刻值。对于两个包i和j，D可以表达为

$$D(i,j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$$

- 到达时刻抖动可以在收到从源SSRC_n来的每个数据包i后连续计算,利用该包和前一包i-1的偏差D。

- 根据公式J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16计算

丢包率

- **丢包率**是通过计算接收包数量和发送包数量的比率得到。
  - **Sequence number，前一seq + 1 == 当前seq**

- **流程**

- 发送方：每间隔一定时间读取每个发送通道的发包数量和数据长度，组成一个此通道的RTCP报文发送给接收方，同时将发送数据包计数清零。
- 接收方：收到RTCP包后，读取接收通道接收到的包数量，并计算出丢包率，通过一个RTCP接收汇报包发送给发送方，同时对接收数据包计数清零。

会话和流两级分化

- 一个RTP会话包括传给某个指定目的地对的所有通信量，发送方可能包括多个。而从同一个同步源发出的RTP分组序列称为流(Stream),**一个RTP会话可能包含多个RTP流**。

- 一个RTP分组在服务器端发送出去的时候总是要指定属于哪个会话和流，在接收时也需要进行**两级分用**，即**会话分用**和**流分用**。

- 只有当RTP使用同步源标识和分组类型把同一个流中的分组组合起来，才能够使用序列号和时间戳对分组进行排序和正确回放。

RTP 背景和概述

RTP封装协议

RTP关键技术

RTP协议运用

RTP打包

RTP协议应用方案——单播

在客户端与媒体服务器之间建立一个单独的数据通道，从一台服务器送出的每个数据包只能传送给一个客户端，这种传送方式称为**单播**。

- **优点**：便于控制和管理
- **缺点**：每个用户必须分别对媒体服务器发送单独的查询，而媒体服务器必须向每个用户发送所申请的数据包拷贝。这种巨大冗余造成服务器负担沉重，响应需要很长时间

RTP协议应用方案——广播

广播指的是用户被动地接收流。

在广播过程中，数据包的单独一个拷贝将发送给网络上的所有用户，客户端接收流，但不能控制流; 广播方式中资料包的单独一个拷贝将发送给网络上的所有用户, 而不管用户是否需要，会非常浪费网络带宽。

- **优点**：简单
- **缺点**：浪费网络带宽

RTP协议应用方案——组播（1）

组播技术构建的网络，允许路由器一次将数据包复制到多个通道上。采用组播方式，媒体服务器只需要发送一个信息包，所有发出请求的客户端即可同时收到连续数据流而无延时。

- 组播吸收了单播和广播两种发送方式的长处，克服了上述两种发送方式的弱点。

RTP协议应用方案——组播（2）

- **优点**：减少网络上传输的信息包的总量；网络利用效率大大提高，成本大为下降。

- **缺点**：当不同的用户同时点播同一个节目时，由于点播总有先后顺序，后点播的用户依照网络中同时点播此节目的其它用户的播放进度，这就造成当前用户极有可能从节目的中间开始看起,很难做到个性化。

RTP 背景和概述

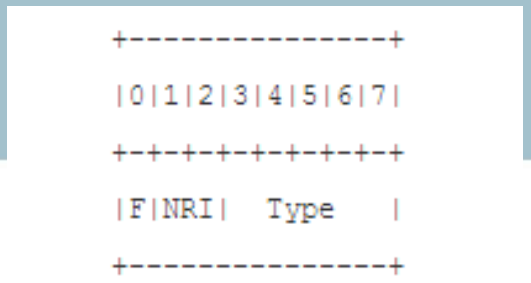
RTP封装协议

RTP关键技术

RTP协议运用

RTP打包 （H264/AAC）

网络抽象层单元类型 (NALU):



F: 1个比特. forbidden_zero_bit. 在 H.264 规范中规定了这一位必须为 0.
NRI: 2个比特. nal_ref_idc. 取00~11,似乎指示这个NALU的重要性,如00的NALU解码器可以丢弃它而不影响图像的回放.
Type: 5个比特. nal_unit_type. 这个NALU单元的类型,简述如下:

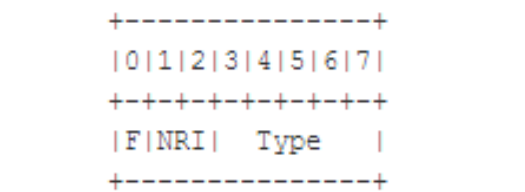
0	没有定义	
1-23	NAL单元	单个 NAL 单元包
24	STAP-A	单一时间的组合包
25	STAP-B	单一时间的组合包
26	MTAP16	多个时间的组合包
27	MTAP24	多个时间的组合包
28	FU-A	分片的单元
29	FU-B	分片的单元
30-31	没有定义	

nal_unit_type	NAL类型	nal_reference_bit
0	未使用	0
1	非IDR的片	此片属于参考帧，则不等于0，不属于参考帧，则等与0
2	片数据A分区	同上
3	片数据B分区	同上
4	片数据C分区	同上
5	IDR图像的片	5
6	补充增强信息单元（SEI）	0
7	序列参数集	非0
8	图像参数集	非0
9	分界符	0
10	序列结束	0
11	码流结束	0
12	填充	0
13..23	保留	0
24..31	不保留	0

1.3. Network Abstraction Layer Unit Types

RTP协议-H264打包模式

H264 RTP负载第一个字节的结构如下,它和H.264的NALU头结构一致，可以把它认为是**RTP H264负载类型**字节，完全是多增加的一个字节，不影响后面的NALU结构



这里的Type类型除1-23外还可取以下值：

24	STAP-A	单一时间的组合包
25	STAP-B	单一时间的组合包
26	MTAP16	多个时间的组合包
27	MTAP24	多个时间的组合包
28	FU-A	分片的单元
29	FU-B	分片的单元

如果使用1-23就是：单一NAL单元模式

https://tools.ietf.org/html/rfc3984

Table 1. Summary of NAL unit types and their payload structures

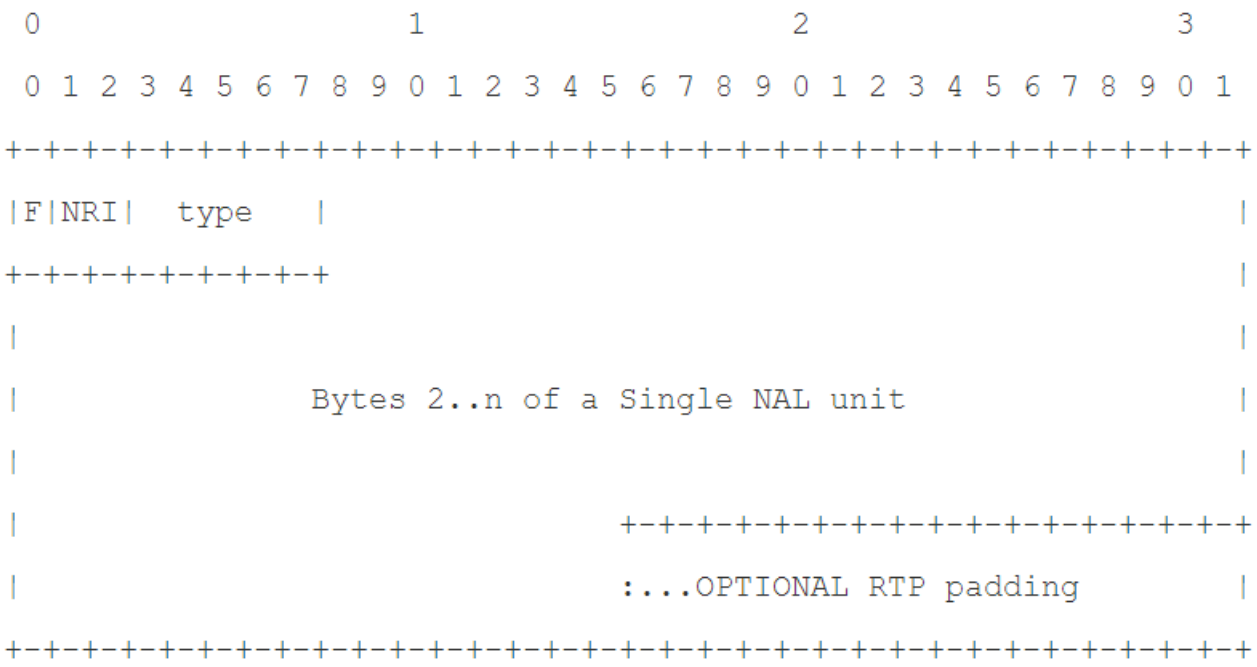
H264 RTP打包模式

本次主要讲解三种打包模式：

1. 单一NAL单元方式 (每个NAL一个RTP包)
2. 组合包模式（多个NAL一个RTP包）
3. 分包模式（1个NAL多个RTP包）

H264 RTP打包-单一NAL单元模式

•对于 NALU 的长度小于 MTU 大小的包, 一般采用单一 NAL 单元模式. 对于一个原始的 H.264 NALU 单元常由 ~~[Start Code]~~ [NALU Header] [NALU Payload] 三部分组成, 其中 Start Code 用于标示这是一个 NALU 单元的开始, 必须是 "00 00 00 01" 或 "00 00 01", NALU 头仅一个字节, 其后都是 NALU 单元内容. 打包时去除 "00 00 01" 或 "00 00 00 01" 的开始码, 把其他数据封包的 RTP 包即可.



H264 RTP打包-单一NAL单元模式

•**例：**

如有一个 H.264 的 NALU 是这样的:
[00 00 00 01 **67 42 A0 1E 23 56 0E 2F ...**]
这是一个序列参数集 NAL 单元. [00 00 00 01] 是四个字节的开始码, 67 是 NALU 头, 42 开始的数据是 NALU 内容.

封装成 RTP 包将如下:
[RTP Header] [67 42 A0 1E 23 56 0E 2F ...]
即只要去掉 4 个字节的开始码就可以了.

0	没有定义	
1-23	NAL单元	单个 NAL 单元包
24	STAP-A	单一时间的组合包
25	STAP-B	单一时间的组合包
26	MTAP16	多个时间的组合包
27	MTAP24	多个时间的组合包
28	FU-A	分片的单元
29	FU-B	分片的单元
30-31	没有定义	

H264 RTP打包 - 组合打包模式 - STAP-A模式

•其次, 当 NALU 的长度特别小时, 可以把几个 NALU 单元封在一个 RTP 包中.

Type	Packet
24	STAP-A
25	STAP-B
26	MTAP16
27	MTAP24

表4. STAPs和MTAPs的类型域

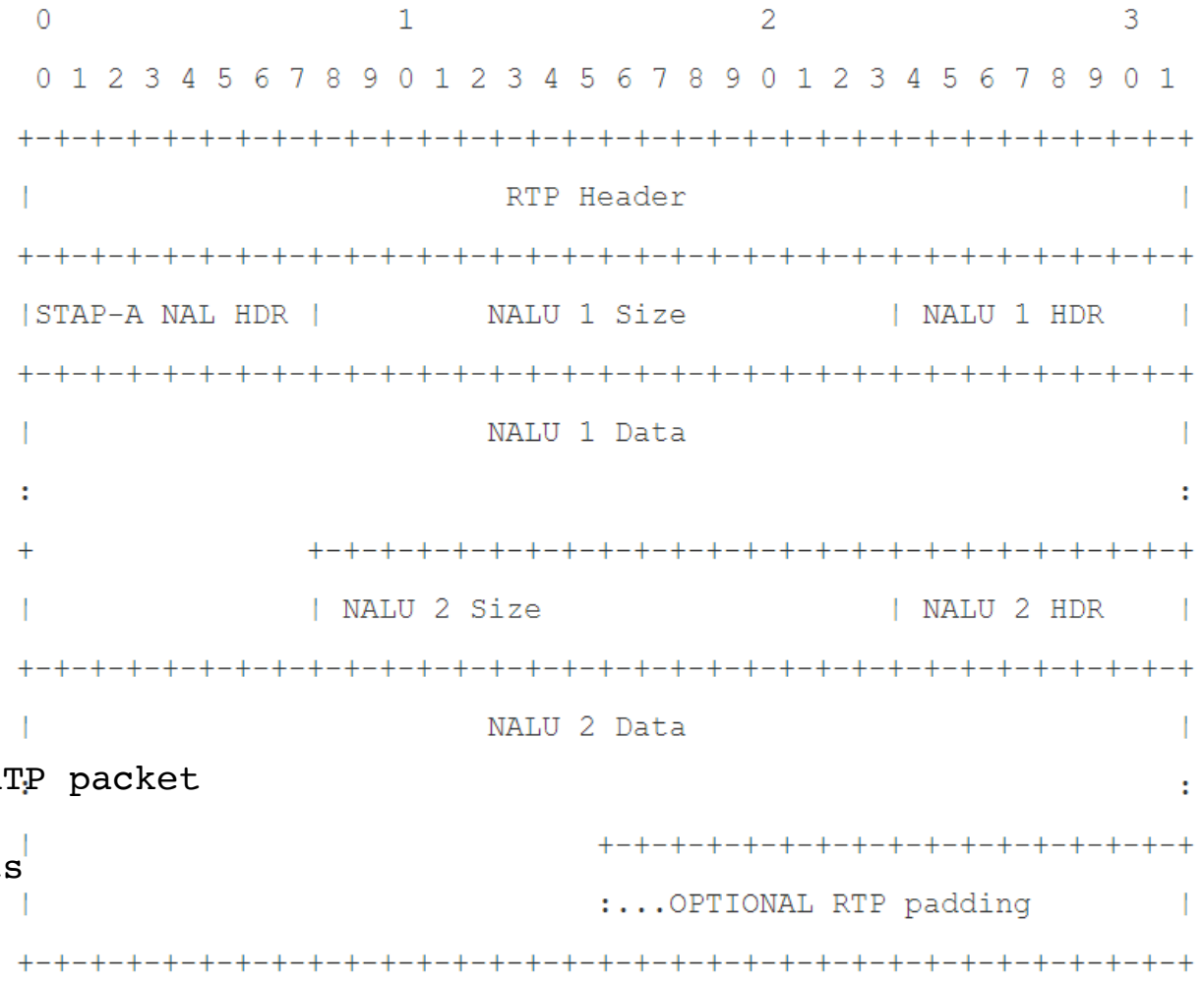


Figure 7. An example of an RTP packet including an STAP-A and two single-time aggregation units

H264 RTP打包 - 组合打包模式 - STAP-A模式 例

如有一个 H.264 的 NALU 是这样的:

[00 00 00 01 67 42 A0 1E 23 56 0E 2F ...]

[00 00 00 01 68 42 B0 12 58 6A D4 FF ...]

封装成 RTP 包将如下:

[RTP Header] [78 (STAP-A头，占用1个字节)] [第一个NALU长度 (占用两个字节)] [67 42 A0 1E 23 56 0E 2F] [第二个NALU长度 (占用两个字节)] [68 42 B0 12 58 6A D4 FF ...]

NALU 1 HDR

67和68属于

0x78 = 01111000, 低5bit即是十进制24

优点：网络传输效率高；

缺点：多帧缓存后再发送容易产生延迟

H264 RTP打包 - FU-A(分包)

当NALU的长度超过MTU时,就必须对NALU单元进行分片封包.也称为Fragmentation Units (FUs)。

本荷载类型允许分片一个NAL单元到几个RTP包中。下图 表示FU-A的RTP荷载格式。FU-A由1字节的FU indicator，1字节的FU header，和分片单元荷载组成。

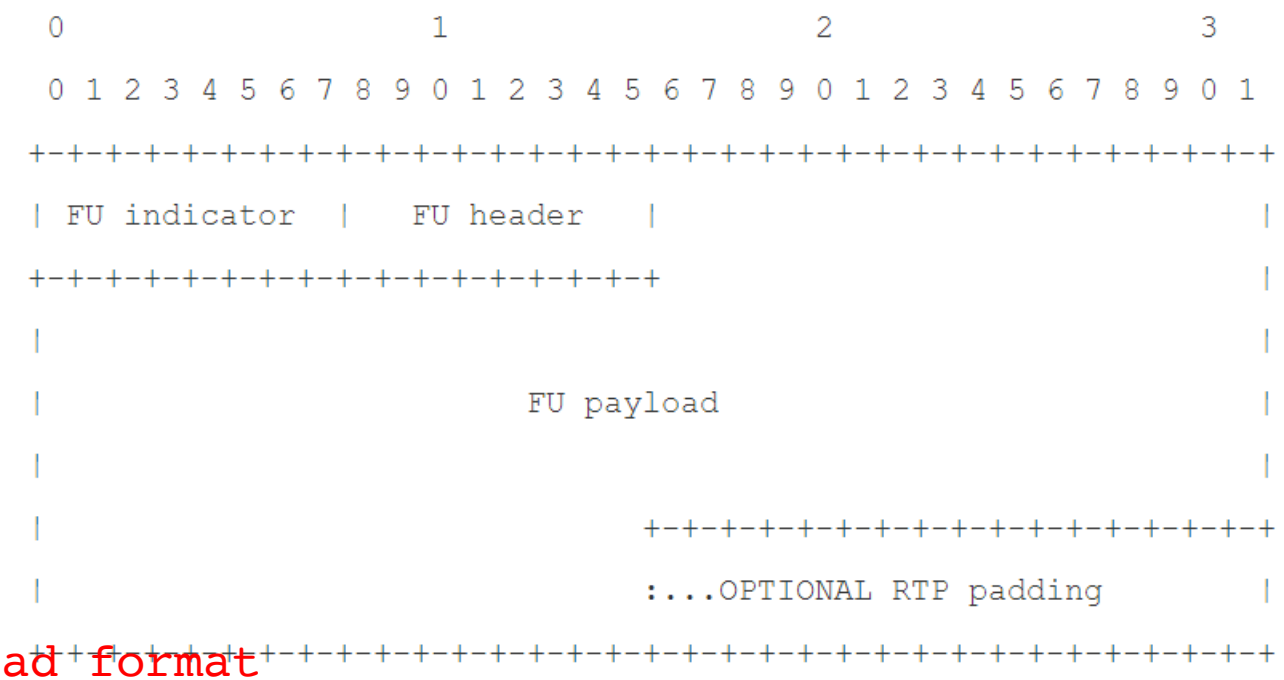
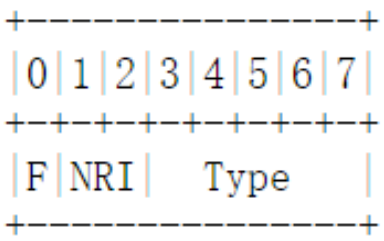


Figure 14. RTP payload format for FU-A

H264 RTP打包 - FU-A(分包) - FU indicator

FU indicator(指示)字节有以下格式：

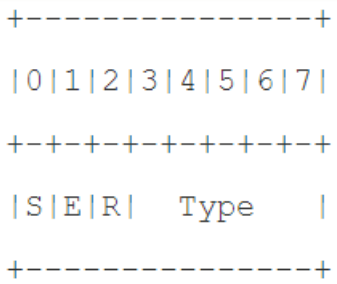


FU指示字节的类型域的28，29分别表示FU-A和FU-B。F的使用在5.3描述。NRI域的值必须根据分片NAL单元的NRI域的值设置。

注意：这是第一个字节FU indicator，NRI为 帧重要程度 00 可以丢 ，11不能丢。F一般设置0。

H264 RTP打包 -FU-A （分包）-FU header

FU header(头)的格式如下：



S: 1 bit

当设置成1,开始位指示分片NAL单元的开始。当跟随的FU荷载不是分片NAL单元荷载的开始，开始位设为0。

E: 1 bit

当设置成1, 结束位指示分片NAL单元的结束，即, 荷载的最后字节也是分片NAL单元的最后一个字节。当跟随的FU荷载不是分片NAL单元的最后分片，结束位设置为0。

R: 1 bit

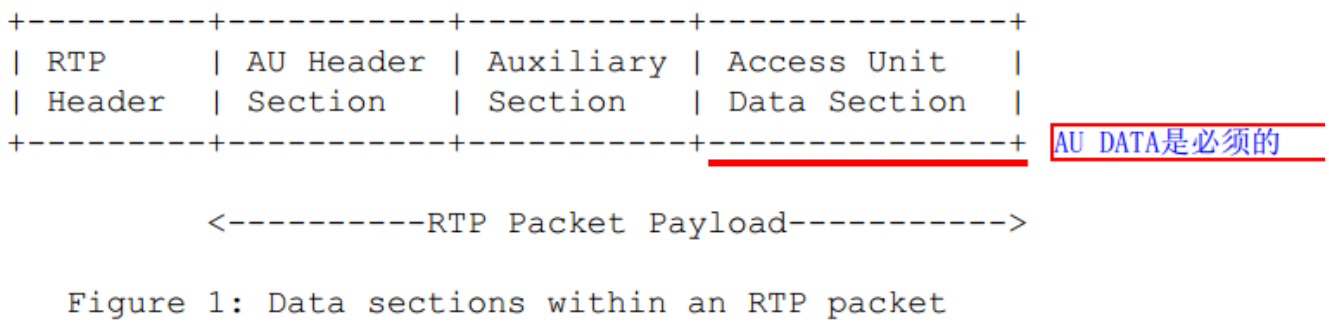
保留位必须设置为0，接收者必须忽略该位。

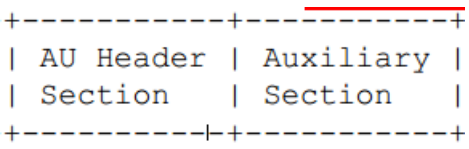
Type: 5 bits

NAL单元荷载类型定义在[ITU-T H.264建议书]的表7-1. (实际)

注意：这是第二个字节，用来表示开始结束和NAL帧的类型。

AAC RTP打包 – RTP 封装格式



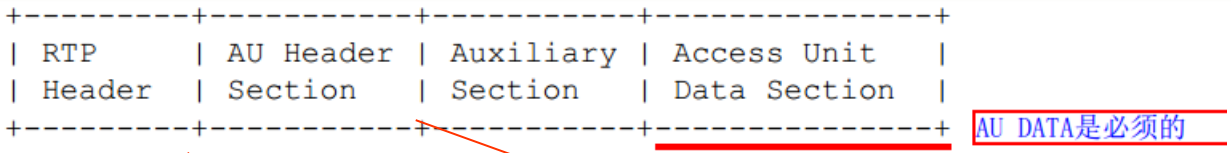


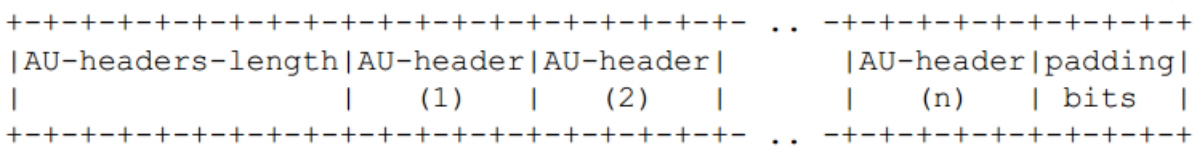
这两部分可以为空

RTP header和 Access Unit data：必须存在

2.11. Global Structure of Payload Format

AAC RTP打包 – AU Header section





AU-headers-length：占2个字节，**单位为bits**，表示所有AU-header + padding bits占用的位数。

每个AU-header对应一个AU data

3.2.1. The AU Header Section

AAC RTP打包 – The fields in the AU-header

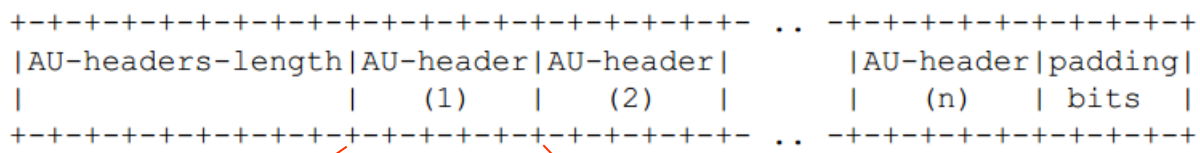
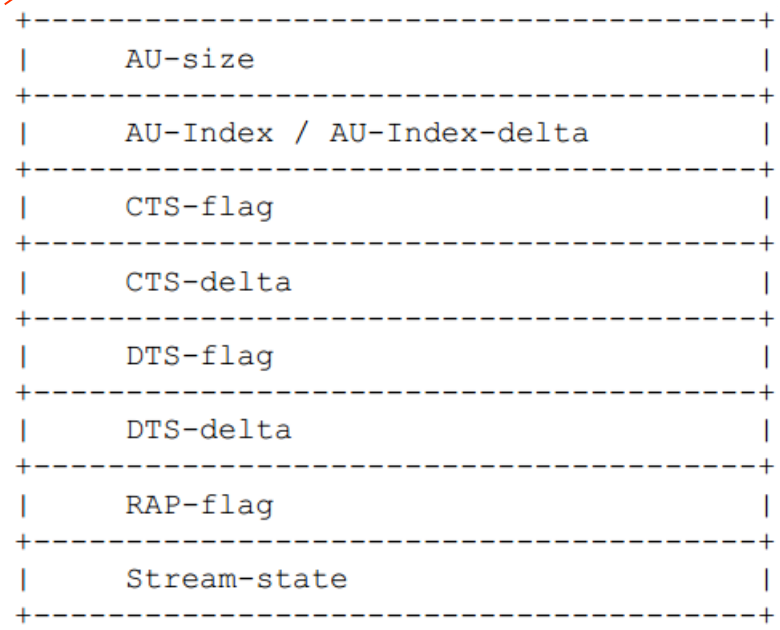


Figure 2: The AU Header Section



AU-size: Indicates the size in octets of the associated Access Unit in the Access Unit Data Section in the same RTP packet。
比如AAC出去ADTS后的长度。

3.2.1. The AU Header Section

AAC RTP打包 – The Access Unit Data Section

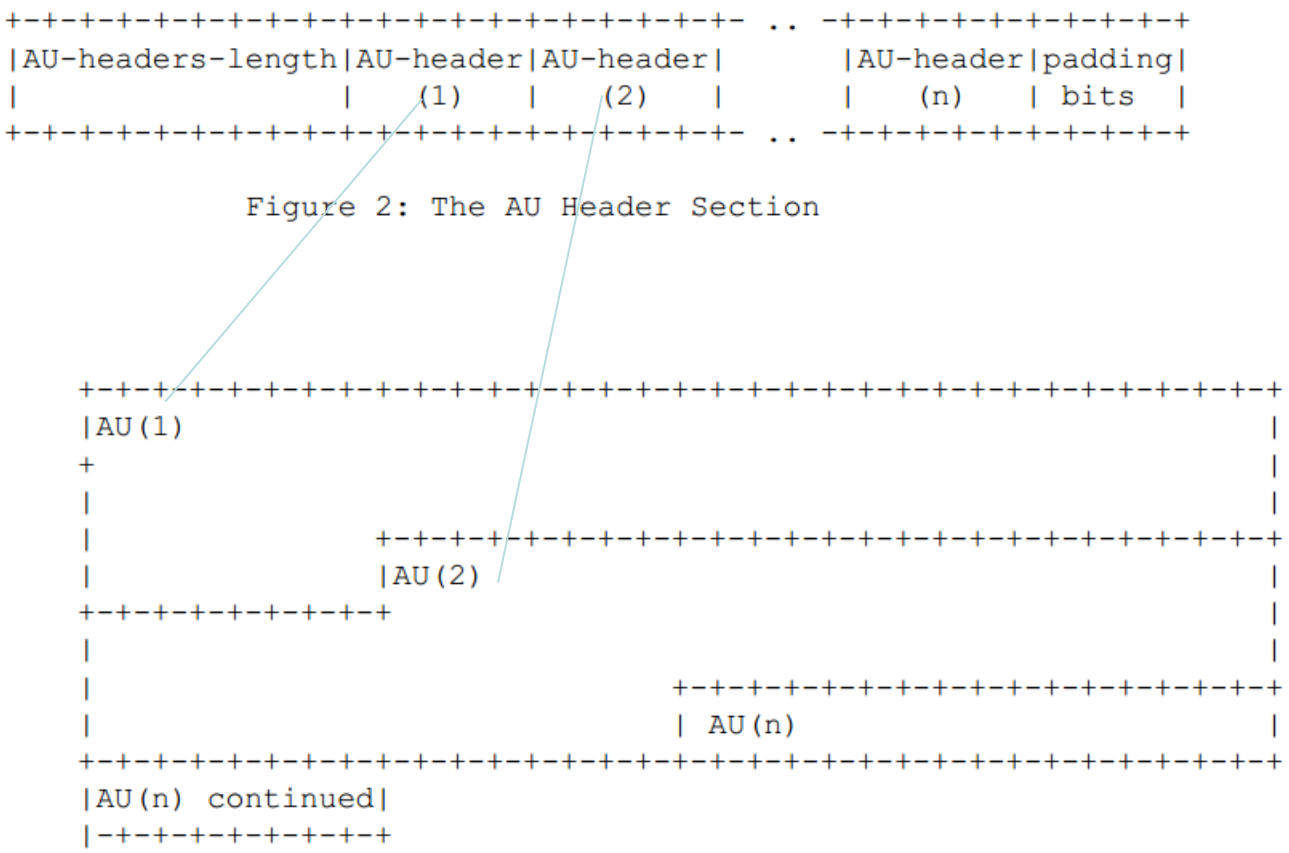


Figure 2: The AU Header Section

AAC RTP打包 – 打包模式

主要讲：

（1）单帧每包模式；

（2）分片模式（单帧多包）；

（3）多帧每包模式；

AAC RTP打包 – 单帧每包

单帧每包：每个RTP包只传输一帧数据

（1）AU header length 2字节，填充如下：

```
uint8_t au_header_length[2];
au_header_length[0] = 0x0;
au_header_length[0] = 0x10;
```

（2）AU header 主要填充au size，2字节（使用高13bit）

```
au_header[0] = (buf_size & 0x1fe0) >> 5;     // 存储高8位
au_header[1] = (buf_size & 0x1f) << 3;       // 存储低5位
```

（3）每个RTP header的mark = 1

AAC RTP打包 – 分片模式

当单帧数据大于最大传输时，需要分片传输，audio分片比较简单，和单帧每包模式区别不大：

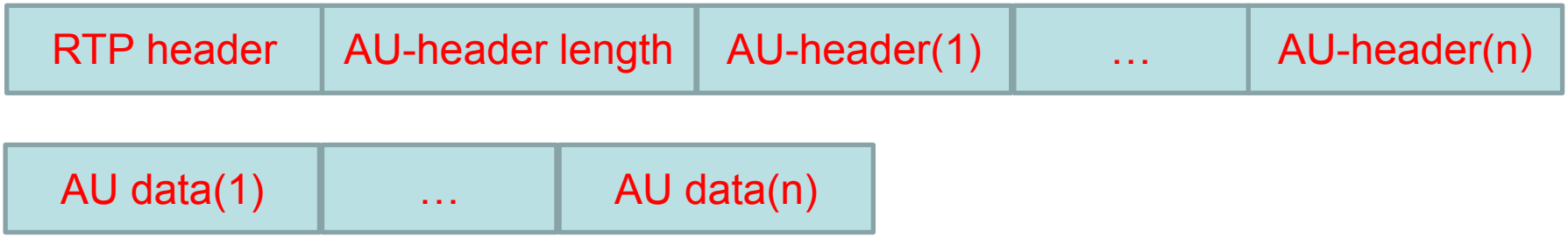
（1）时间戳保持不变；

（2）序号改变；

（3）最后一包RTP header标记mark = 1

3.2.3.1. Fragmentation

AAC RTP打包 –多帧每包模式



多帧拼在一个包进行传输，则相应延迟也变大。

FEC 前向纠正 +10%

+20%