+ Ethernet II: 14 Bytes

+ IP: 20 Bytes

+ UDP: 8 Bytes

+ RTP: 12 Bytes

----------------------------------

= Hearder=54 Bytes

# RTP: 12 Bytes

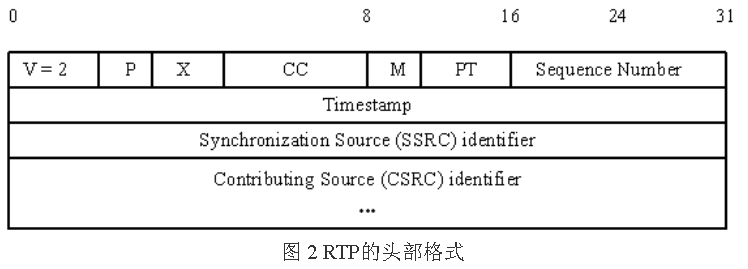
**包头分析：**

**//h: 80 21 9f a5 1c ba 5c 1b df db fe af**

//b: 10 0 0 0000 0 0100001 1001111110100101

//h: 1c ba 5c 1b

//h: df db fe af



# RTCP：SR/RR

RTCP提供5种报文类型，本文主要介绍SR和RR两种重要的报告。

SR(sender report:200)：发送者报告，提供64位NTP和32位timestamp(用于音视频同步)等。

RR(receiver report:201)：接收者报告，提供RTP包的丢包，时延等QOS统计。

BYE(203): 结束多媒体会话时发送BYE，通知释放。

SDES(202): 源描述信息

APP：APP应用信息

## SR: PT=200=0xC8

**包头分析：**

80c80006 //b: 10(V)0(P) 00000(RC) h:C8(PT=SR=0xC8=200) h:0006(L=6)

33a59a23 //SSRC

d0c8fc76 //NTP MSW: 单位为秒(s)，相对于1900年1月1日的秒数，

5c08c826 //NTP LSW: 单位大约是232 picoseconds(即1s=10的12次方picoseconds)，即把1s分隔成2的32次方分

d1273ff6 //RTP: 对应SSRC的采样时间戳

0007f1ed //packet count: 发送者包数目：即从开始传输到生成本次SR包的总RTP数据包数目。

28d7a654 //发送的字节(octets)数目：从开始传输到此生成本次SR包产生时该发送者在RTP数据包发送的字节总数(不包括头和填充)，

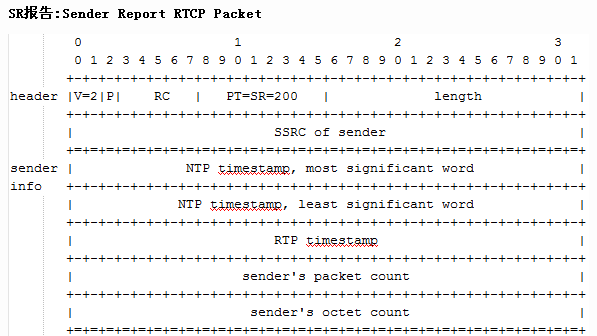
81ca0004 //0xca(SDES:202): Source Description, L=4

33a59a23 //Identifier: SSRC

01096c6f //h: 01(Type=CNAME) 09(L)

63616c68 //h: 6c6f63616c686f7374 = "localhost"

6f737400



## RR: PT=201=0xC9

**包头分析：**

81c90007 //b: 10(V)0(P) 00001(RC) h:C9(PT=RR=0xC9=201) h:0007(L=7)

4741f27c //SSRC

5a244d41 //SSRC\_1

00000141 //h: 00(丢包率) 000141(累计丢包数)

00047e23 //扩展最高序列号: 低16比特为从源SSRC\_n的最高接收序列号，高16比特用相应的序列号周期计数器扩展序列号

00000050 //到达间隔抖动: RTP数据包到达时刻统计方差的估计值，反映的是瞬时时刻的抖动。**测量单位同时间戳单位**

f639cc45 //last SR: 上次从SSRC\_n接收到的RTP包的NTP中间的32位

0005d348 //DLSR：32比特，从SSRC\_n接收到上次SR开始计时，到本次发送RR的时长，以1/65536秒为单位

81ca0004 //0xca(SDES:202): Source Description, L=4

4741f27c //Identifier: SSRC

01076c79 //h: 01(Type=CNAME) 07(L)

752d7368 //CNAME Text: 0x6c 79 75 2d 73 68 61 = lyu-sha

61000000 //

# TS数据格式: 188 Bytes

TS关键字：

**47 40 00 1c 00 00 b0 0d; //PAT: 4000+1x**

**47 41 e0 1c 00 02 b0 12; //PMap: 41e0+1x**

**47 01 e1 2c b7 10 00 3f; //PCR: 01e1+2x**

**47 41 e1 1d 00 00 01 e0; //Unit-S: 41e1+1x**

**47 01 e1 1e ff ff d5 9f; //Unit-P: 01e1+1x**

**47 01 e1 3b 2f 00 ff ff; //Unit-E: 01e1+3x**

writeProgramAssociationTable()

{

//0x47 0x40 0x00 0x1? 0x00 0x00 0xb0 0x0d

/\*

static const uint8\_t kData[] = {

0x47,

0x40, 0x00, 0x10, 0x00, // b0100 0000 0000 0000 0001 ???? 0000 0000

0x00, 0xb0, 0x0d, 0x00, // b0000 0000 1011 0000 0000 1101 0000 0000

0x00, 0xc3, 0x00, 0x00, // b0000 0000 1100 0011 0000 0000 0000 0000

0x00, 0x01, 0xe1, 0xe0, // b0000 0000 0000 0001 1110 0001 1110 0000

0x00, 0x00, 0x00, 0x00 // b???? ???? ???? ???? ???? ???? ???? ????

};

\*/

0x47, (type)

? 0x4000,

b: 0(transport\_error\_indicator)

1(payload\_unit\_start\_indicator)

0(transport\_priority)

0 0000 0000 0000(0x0000)(PID)

0x1?,

b: 00(transport\_scrambling\_control)

01(adaptation\_field\_control)

????(continuity\_counter) //????=[0-16]

0x00, (skip)

? 0x00, (table\_id)

? 0xb00d,

b: 1(section\_syntax\_indicator)

0(must\_be\_zero)

11(reserved)

0000 0000 1101(0x00d)(section\_length, fixed)

? 0x0000, (transport\_stream\_id)

0xc3,

b: 11(reserved)

00 001(version\_number)

1(current\_next\_indicator)

0x00, (section\_number)

0x00, (last\_section\_number)

? 0x0001, (program\_number)

? 0xe1e0,

b: 111(reserved)

0 0001 1110 0000(0x01e0)(program\_map\_PID, 13 bits)

0x????????, //crc: Byte[5, 5+12]

}

writeProgramMap

{

//0x47 0x40 0x00 0x1? 0x00 0x00

//0x47 0x41 0xe0 0x1? 0x00 0x02

/\*

static const uint8\_t kData[] = {

0x47,

0x41, 0xe0, 0x10, 0x00, // b0100 0001 1110 0000 0001 ???? 0000 0000

0x02, 0xb0, 0x00, 0x00, // b0000 0010 1011 ???? ???? ???? 0000 0000

0x01, 0xc3, 0x00, 0x00, // b0000 0001 1100 0011 0000 0000 0000 0000

0xe0, 0x00, 0xf0, 0x00 // b111? ???? ???? ???? 1111 0000 0000 0000

};

\*/

0x47, (type)

? 0x41e0,

b: 0(transport\_error\_indicator)

1(payload\_unit\_start\_indicator)

0(transport\_priority)

0 0001 1110 0000(0x01e0)(PID, 13 bits)

0x1?,

b: 00(transport\_scrambling\_control)

01(adaptation\_field\_control)

????(continuity\_counter, 4bits) //?=[0-16]

0x00, (skip)

? 0x02, (table\_id)

? 0xb???,

b: 1(section\_syntax\_indicator)

0(must\_be\_zero)

11(reserved)

???? ???? ????(0x???)(section\_length, 12bits) //section\_length = 5 \* mSources.size() + 4 + 9;

? 0x0001, (program\_number)

0xc3,

b: 11(reserved)

00 001(version\_number)

1(current\_next\_indicator)

0x00, (section\_number)

0x00, (last\_section\_number)

? 0xe000, (program\_number)

b: 111(reserved)

? ???? ???? ????(0x????)(PCR\_PID, 13bits) //kPCR\_PID = 0x1e1

? 0xf000,

b: 1111(reserved)

0000 0000 0000(0x000)(program\_info\_length, 12 bits)

0x??, (stream\_type) //AAC:0x0f, AVC:0x1b

0xe000,

b: 111(reserved)

? ???? ???? ????(elementary\_PID, 13bits) //ES\_PID = 0x1e0 + i + 1;

0xf000,

b: 1111(reserved)

0000 0000 0000(ES\_info\_length, 12bits)

0x00000000, //crc: Byte[5, 5+12+mSources.size()\*5]

}

writePCRUnit

{

//0x47 0x40 0x00 0x1? 0x00 0x00 //PAT

//0x47 0x41 0xe0 0x1? 0x00 0x02 //PMap

//0x47 0x01 0xe1 0x2? 0xB7 0x10 //PCR

0x47,

0x0000 | PID, //PID = 0x1e0 + sourceIndex + 1;

0x20 | [0,16], //Adaptation only: 0x20 | continuity\_counter:[0, 16]

0x00 | [Size], //Adaptation Field Length: TS\_PACKET\_SIZE[188] - 5,

0x10, //Adaptation flags: PCR present

0x000000000000 | PCR, //PCR: 6Bytes , int64\_t PCR = (timeUs \* 27ll);

}

writeAccessUnit

{

//0x47 0x40 0x00 0x1? 0x00 0x00 //PAT

//0x47 0x41 0xe0 0x1? 0x00 0x02 //PMap

//0x47 0x01 0xe1 0x2? //PCR

//0x47 0x41 0xe1 0x1? //Unit

//0x47 0x41 0xe1 0x3? //Unit-end

47 40 0 1c 0 0 b0 d //PAT: 4000+1x

47 41 e0 1c 0 2 b0 12 //PMap: 41e0+1x

47 1 e1 2c b7 10 0 3f //PCR: 01e1+2x

47 41 e1 1d 0 0 1 e0 //Unit-S: 41e1+1x

47 1 e1 1e ff ff d5 9f //Unit-P: 01e1+1x

47 1 e1 3b 2f 0 ff ff //Unit-E: 01e1+3x

47 1 e1 2x b7 10 0 37 //PCR

47 41 e1 1x 0 0 1 e0 //Unit-start

47 1 e1 1x 78 be e 56 //Unit-part

47 1 e1 3x 41 0 ff ff //Unit-end

0x47,

0x4000 | PID, //PID = 0x1e0 + sourceIndex + 1;

(padding ? 0x30 : 0x10) | continuity\_counter

//bool padding = (accessUnit->size() < (188 - 18));

//(padding ? 0x30 : 0x10) | continuity\_counter;

if padding==true then ptr+=paddingSize

{

0x00+(paddingSize-1),

0x00

....

}

0x00 00 01 stream\_id, //stream\_id= streamType == 0x0f ? 0xc0 : 0xe0;

0x0000 | PES\_packet\_length, //PES\_packet\_length=accessUnit->size() + 8

0x84

0x80

0x05

0x2000000000 | PTS, //PTS = (timeUs \* 9ll) / 100ll;

copy mAVCCodecSpecificData, size=mAVCCodecSpecificData->size(); //16个包之间插入一次mAVCCodecSpecificData数据

copy accessUnit, size= sizeLeft or accessUnit->size(); //sizeLeft = buffer->data() + buffer->size() - ptr

--------------

0x47

0x0000 | PID, //PID = 0x1e0 + sourceIndex + 1;

(lastAccessUnit ? 0x30 : 0x10) | continuity\_counter //lastAccessUnit = ((accessUnit->size() - offset) < 184);

if lastAccessUnit==true then ptr+=paddingSize

{

0x00+(paddingSize-1),

0x00

....

}

copy accessUnit, size= sizeLeft or accessUnit->size(); //sizeLeft = buffer->data() + buffer->size() - ptr

}