

猿创

猫诗人

于 2019-04-18 19:59:46 发布

4728

收藏 35

版权

分类专栏:

h264传输-编解码

流媒体 RTSP/RTMP/RTCP/RTMP H264

文章标签:

H264

h264传输-编解码

同时被 2 个专栏收录

3 订阅

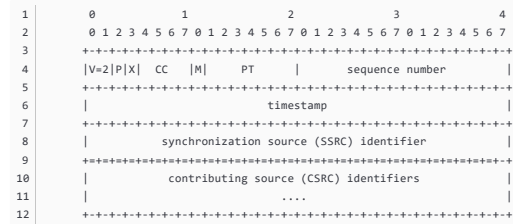
9 篇文章

订阅专栏

H264帧需要通过RTP协议进行传输，这其中就涉及到H264数据帧的封装、拆包和解包等过程。

### RTP协议格式

下面是 RFC 3550 中规定的 RTP 头的结构：



RTP协议头部共12字节，在每一个RTP packet中都会存在，头部之后就是RTP payload部分，也即是H264数据；这里只关注RTP payload部分。由于H264帧数据打包进入RTP packet时会把start code (00 00 00 01 或者00 00 01) 去掉，所以RTP payload部分第一字节就是H264 NALU Header部分；

### H264 NALU

H264帧是由一系列的NALU组成，NALU Header由一个字节组成,它的语法如下：



**F:** 1个比特. forbidden\_zero\_bit 在 H.264 规范中规定了这一位必须为 0.禁止位，0表示正常，1表示错误，一般都是0；

**NRI:** 2个比特. nal\_ref\_idc. 取00~11,表示这个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, 重要 nal\_unit\_type. 这个NALU单元的类型.简述如下：

1	Type	Packet	Type name
2			
3	0	undefined	-
4	1-23	NAL unit	Single NAL unit packet per H.264
5	24	STAP-A	Single-time aggregation packet
6	25	STAP-B	Single-time aggregation packet
7	26	MTAP16	Multi-time aggregation packet
8	27	MTAP24	Multi-time aggregation packet
9	28	FU-A	Fragmentation unit
10	29	FU-B	Fragmentation unit
11	30-31	undefined	

之前提到过，单个H264仅用1-23类型，24以及以后的用在RTP协议中 H264数据的头部，用于标识传输过程中H264的类型。

### H264打包过程

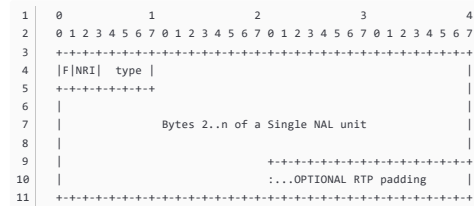
对于一个原始的 H.264 NALU 单元常由 [Start Code] [NALU Header][NALU Payload] 三部分组成. 其中 StartCode 用于标示这是一个NALU 单元的开始, 必须是 "00 00 00 01" 或 "00 0001", NALU 头仅一个字节, 其后都是NALU 单元内容.打包时去除 "00 0001" 或 "00 00 00 01" 的开始码, 把其他数据封装的 RTP 包即可.

根据H264 NALU的数据长度和MTU进行比较的结果，RTP有三种打包模式：单一NALU模式、组合封装模式、分片封装模式。

#### 1、单一NALU模式

对于NALU的长度小于MTU大小的包，一般采用单一NALU模式：

下面就是相应的格式示意：



如有一个 H.264 的 NALU 是这样的：

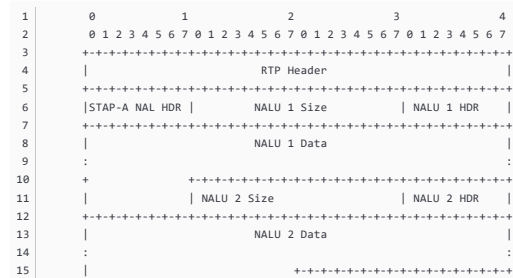
[00 0000 01 67 42 A0 1E 23 56 0E 2F ... ]

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 包如下：

[ RTPHeader ][ 67 42 A0 1E 23 56 0E 2F ]

#### 2、组合封装模式

当多个NALU长度特别小时，可以将多个NALU一起封装在一个RTP包中。这里就对应着NALU Type为24、25、26、27.下面就是相应的格式示意：



发布首篇文章，得大额流量券

点亮新秀勋章

分类专栏

流媒体 RTSP/RTMP/RTCP/RTMP H264

14 篇

android开发

20 篇

ble

4 篇

蓝牙

2 篇

java反射机制

1 篇

shell

1 篇

Camera开发

3 篇

java

11 篇

AudioRecorder

1 篇

jni开发

12 篇

ffmpeg-解码h264

1 篇

h264传输-编解码

9 篇

okhttp

13 篇

Android USB

3 篇

```
16 |                                     :...OPTIONAL RTP padding |
17 | +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

例如：如有一个 H.264 的 NALU 是这样的:

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

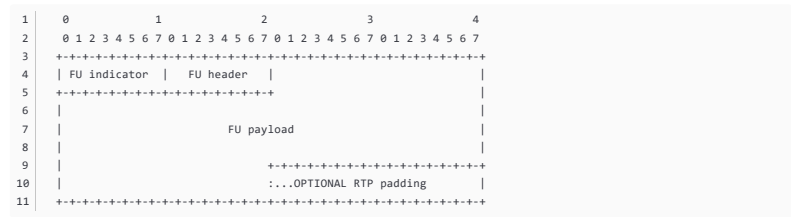
[00 00 00 01 68 42 12 2B 6A D6 FF E4...]

封装成RTP包，结果如下：

[RTP Header] [78(STAP-A头，一个字节)] [第一个NALU长度，两个字节] [67 42 A0 1E 23 56 0E 2F...][第二个NALU长度，两个字节] [68 42 12 2B 6A D6 FF E4...]

### 3、分片封装模式

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



FU indicator有以下格式:



FU指示字节的类型域Type=28表示FU-A。。NRI域的值必须根据原分片NAL单元的NRI域的值设置。

FU header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 取1-23的那个值，也就是原被分片的H264帧的原始NALU Type

NAL单元荷载类型定义见下表:

nal_unit_type	NAL类型	C
0	未使用	
1	不分区、非IDR的片	2,3,4
2	片分区A	2
3	片分区B	3
4	版分区C	4
5	IDR图像中的片	2,3
6	补充增强信息单元 (SEI)	5
7	序列参数集	0
8	图像参数集	1
9	分界符	6
10	序列结束	7
11	码流结束	8
12	填充	9
13..23	保留	
24..31	不保留	

分片模式举例：一帧H264为：

00 00 00 01 21 42 0A DE DF FF AD 5D ... (需要第一个RTP Packet)...A5 00 0F 08 56 78 DA ... (需要第二个RTP Packet) ...DE D6 D8 89 09 00 0F...(需要第三个RTP Packet)... 则按照分片封装模式，分为三个包，依次为：

第一个包：[RTP Header] [FU Indicator =3C ] [FU Header = 81] [21(去掉) 42 0A DE DF FF AD 5D ...][...]

第二个包：[RTP Header] [FU Indicator =3C ] [FU Header = 01] [A5 00 0F 08 56 78 DA ...][...]

第三个包：[RTP Header] [FU Indicator =3C ] [FU Header = 41] [DE D6 D8 89 09 00 0F...][...]

然后，总共三个RTP Packet发出去。

**解包过程：**

当接收端接收到一个的一个的RTP Packet，就需要将其中的H264数据取出，并还原原始的H264帧，交由解码器解码。

单一NALU模式和组合封装模式解包过程其实比较简单，主要看一下分片封装模式：

1、当接收到接收到FU-A类型的RTP Packet时，需要看FU Header中的S / E两位：

- 如果时10，表示时一帧数据的开头部分；
- 如果是00，表示一帧数据的中间部分，有可能多个00的分片；
- 如果是01，表示一帧数据的结尾。

2、把之前所有的数据都暂时保存、拼接成完成的一帧H264数据，然后加上start code：00 00 00 01即可，然后就可以交由解码器解码；

3、还原后的NAL头的八位是由FU indicator的前三位加FU header的后五位组成，即：

nal\_unit\_type = (fu\_indicator & 0xe0) | (fu\_header & 0xf)

接下来，看代码

```
1 //This method is used to get the NAL unit from the RTP packet
2 //Then translate the NAL unit to the decoder
3 public void H264PacketRecombine(byte[] data,int length) {
4     nalType = data[0] & 0x1F;
5
6     switch (nalType) {
7         //Single-timeaggregation packet
8         case NAL_UNIT_TYPE_STAP_A:
9             break;
10
11         //Single-timeaggregation packet
12         case NAL_UNIT_TYPE_STAP_B:
13             break;
14
15         //Multi-time aggregationpacket
16         case NAL_UNIT_TYPE_MTAP16:
17             break;
18
19         //Multi-time aggregationpacket
20         case NAL_UNIT_TYPE_MTAP24:
21             break;
22
23         //Fragmentationunit
24         case NAL_UNIT_TYPE_FU_A:
25             packFlag = data[1] & 0xC0;
26             switch (packFlag) {
27                 //NAL Unit start packet
28                 case 0x80://一帧的开头
29                     NALUnit[4] = (byte) ((data[0] & 0xE0) | (data[1] & 0x1F));
30                     System.arraycopy(data, 2, NALUnit, 5, length - 2);
31                     tmpLen = length + 5 - 2;
32                     break;
33                 //NAL Unit middle packet
34                 case 0x00:
35                     System.arraycopy(data, 2, NALUnit, tmpLen, length - 2);
36                     tmpLen +=length - 2;
37                     break;
38                 //NAL Unit end packet
39                 case 0x40:
40                     System.arraycopy(data, 2, NALUnit, tmpLen, length - 2);
41                     tmpLen +=length - 2;
42
43                     if (SquareApplication.getInstance().getH264Stream()!=null) {
44                         SquareApplication.getInstance().getH264Stream().decodeH264Stream(Arrays.copyOfRange(
45                             data, 2, length - 2), NALUnit, tmpLen);
46                         tmpLen =0;
47                     }
48                     break;
49             }
50             break;
51
52         //Fragmentationunit
53         case NAL_UNIT_TYPE_FU_B:
54             break;
55
56         //Single NAL unit per packet
57         default:
58             if (SquareApplication.getInstance().getH264Stream()!=null) {
59                 System.arraycopy(data, 0, BaseNALUnit, 4, length);
60                 SquareApplication.getInstance().getH264Stream().decodeH264Stream(Arrays.copyOfRange(
61                     data, 0, length - 4), BaseNALUnit, 4);
62                 tmpLen =0;
63             }
64             break;
65         }
66     }
67 }
68 }
```

接下来就是使用MediaCodec进行硬解码。

阅读终点，创作起航，您可以撰写心得或摘录文章要点写篇博文。去创作 > ×

 猿诗人 关注

 3   35   1  专栏目录

h264基础及rtsp分包解包 machh的专栏 · 1万+  
一、h264基础概念SODB: 数据比特串 - -> 最原始的编码数据RBSP: 原始字节序列载荷 - -> 在SODB的后面填加了结尾比特 (RBSP trailing bits ...)

论文研究-基于RTP的H.264视频传输系统研究.pdf 08-15  
基于RTP的H.264视频传输系统研究, 熊伟, 杨春金, 本文通过研究实时传输协议RTP/RTCP协议和H.264码流的层次结构, 在此基础上, 深入研究了适合...

1 条评论 qq\_26602805 热评 请问rtsp包里面有padding位, 请问我需要加入进去吗? 写评论

音视频传输\_RTP协议详解和H.264打包方案\_rtp格式\_音视频开发老舅的博客... 9-3  
但是结合流媒体传输的特点,我们发现UDP更适合传输,所以我们把保证传输速度即快又正确这件事交给了上层的RTP和RTCP协议.这样我们开发者是可以来...

流媒体专家(10)rtp传输H264\_rtp到h264\_奇妙之二的博客-CSDN博... 9-3  
H.264的规范. 首先要明确,RTP包的格式是绝不会变的.永远都是RTP头+RTP载荷: RTP...

RTSP服务器: RTP打包传输H264码流 weixin\_43796767的博客 · 1612  
H264码流与RTP协议分析 1) H264码流由一个NALU组成, 每个NALU前面都有起始码 (00 00 00 01或00 00 01), 充当分隔符作用. 每个NALU有1字...

rtp发送h264码流 最新发布 qq\_42161913的博客 · 58  
rtp发送h264码流

H264解码之RTP流解析\_h264\_rtp\_SunkingYang的博客 8-23  
即我们所使用的RTP+H264一般都是一个RTP包中最多只有一个NALU包.在nal\_unit\_type的定义中(见表一).0到23是给H264用的.24~31未使用.在rtp打包时...

H264基础及RTP分包解包\_rtp\_h264\_xiaopangcame的博客 8-19  
RFC3984是H.264的baseline码流在RTP方式下传输的规范.这里只讨论FU-A分包方式. H264的码流结构 1、单个NAL包单元: 12字节的RTP头后面的就是...

wireshark从RTP包中提取出H264裸流数据脚本 10-24  
wireshark从RTP包中提取出H264裸流数据脚本 wireshark RTP H264 H265

wireshark-rtp-h264-extractor 04-28  
rtp\_h264\_extractor 将RTP h.264有效负载传输到原始h.264文件 (\* .h264) 根据RFC3984将RTP的H264有效内容分解为NALU, 并将其从<sourceip>写入...

RTP协议详解(荷载H264)\_rtp报文结构\_Sparkl的博客 8-25  
荷载格式定义三个不同的基本荷载结构,接收者可以通过RTP荷载的第一个字节后5位识别荷载结构. 1、单个NAL单元包:荷载中只包含一个NAL单元. NAL...

H264码流RTP封装方式详解 热门推荐 技术控的笔记 · 5万+  
在流媒体传输领域,经常使用的传输协议是RTP/RTCP, 本文将对RTP对H264进行封装的过程进行详解

RTP(荷载H264码流)基础知识 u014415274的博客 · 4167  
一、H264介绍 h264是一种视频压缩标准, 经过压缩的帧分为I,P,B帧: 帧: 关键帧, 采用帧内压缩技术, 自身可以通过视频解压算法解压成一张单独的完...



猿诗人

码龄12年

暂无认证

62

5万+

180万+

15万+



原创

周排名

总排名

访问

等级

1893

49

58

16

281

积分

粉丝

获赞

评论

收藏



私信

关注

H264视频传输、编解码----RTP协议对H264数据帧打包、打包、解决了你的问题么? 可以写篇文章记录加深印象噢~

写文章

- 热门文章
- H264视频传输、编解码----RTSP协议 · 17111

Android变录音边转换为mp3格式的声音---libmp3lame库的使用 · 10529

H264视频传输、编解码----H264数据结构 · 9261

Android BLE项目中相应的超时处理机制 · 8624

Android USB设备通信-读写操作 · 8343

最新评论

Android JNI开发--集成第三方SO

CSDN-Ada助手: 非常感谢CSDN博主的分享, 非常实用的一篇博客。我觉得下一篇博...

基于RTMP协议的音视频传输----FLV格式

怕什么真理无穷: 兄弟, 写的不错, 有H265的吗?

Android应用监听来电、短信等方法

帮诺: 你好, 我想问下为什么那么多报错 该怎么解决 xml 注册了组件 然后 抽象那个...

Android USB设备通信--读写操作

Coder.Chen+: 你这个lock是哪里来的

Android BLE 通信处理过程---串行通信

day\_moon: 总结很到位, 唯一缺陷就是没demo

您愿意向朋友推荐“博客详情页”吗?

强烈不推荐

不推荐

一般般

推荐

强烈推荐

最新文章

Android JNI开发--集成第三方SO

Android JNI开发--基本内容

H265数据结构与码流分析

2020年 3篇

2018年 3篇

2016年 12篇

2019年 26篇

2017年 26篇

使用RTP包加载H264码流数据

在音视频通话中我们通常使用 RTP 协议包 承载音视频码流数据, 例如用户摄像头采集图像后进行编码成帧, 再将帧数据拆分到 RTP 协议包后发送到流媒...

基于RTP协议的H.264视频传输系统: 实现

实现的原理: 基于RTP协议的H.264视频传输系统: 原理 相关文章: 【1】RTP协议分析 【2】jrtplib简介 【3】Qt调用jrtplib实现单播、多播和广播 【4】...

h264流, 拆出来帧与其余帧, 并能重新组装, 完整工程

可以提取h264流中的帧和其他类型帧, 并可以重新组装起来, 还原出原来的h264流, 使用了x264中代码, 参考了网络上二位仁兄的代码

H264裸数据打包RTP方法和实验.pdf

主要讲了H264数据格式, RTP数据格式, 以及如何将H264的NALU打包为RTP。 本文的实验在VLC播放器进行了验证, 能够将本地文件推送到VLC进行展...

H264-encode-and-decode.rar\_H264实时编码\_h264传输

\bin 目录下是已编译成功的编解码器程序以及相应说明 \videocod目录下是H.264视频解码器程序码。 \lencod目录下是H.264视频编码器程序码。...\rtcpdump ...

RFC3984协议-H.264(RTP荷载格式).pdf

RFC3984协议-H.264(RTP荷载格式).pdf

音视频传输: RTP协议详解和H.264打包方案

问题背景: 前面讲解了PS、TS、FLV这三种媒体封装格式, 现在新开一个系列讲解下传输协议, 这里面会包含RTP、RTSP、HLS、RTMP等。当然最复...

H264码流分析和打包RTP过程

对摄像头采集的每一帧视频需要进行编码, 由于视频中存在空间和时间的冗余, 需要用算法来去除这些冗余。H264是专门去除这些冗余的算法, 我们把这...

网络摄像机 (IPC) 开发 (7) : RTP协议解析 (H264码流)

一、RTP (实时传输协议) RTP全名是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 (RFC1...

H264 RTP包解析

预备 视频: 由一副副连续的图像构成, 由于数据量比较大, 因此为了节省带宽以及存储, 就需要进行必要的压缩与解压缩, 也就是编解码。 h264裸码流...

ffmpeg5.0解码rtp协议传输的h264码流

要使用FFmpeg 5.0解码RTP协议传输的H.264码流, 需要使用以下命令: `` ffmpeg -i rtp://address:port -vcodec copy -f h264 - `` 其中, address是RTP...

“相关推荐” 对你有帮助么?

非常没帮助

没帮助

一般

有帮助

非常有帮助

关于我们 招贤纳士 商务合作 寻求报道 400-660-0108 kefu@csdn.net 在线客服 工作时间 8:30-22:00

公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 家长监护 网络110报警服务 中国互联网举报中心 Chrome商店下载 账号管理规范 版权与免贵声明 版权申诉 出版物许可证 营业执照 ©1999-2023北京创新乐知网络技术有限公司