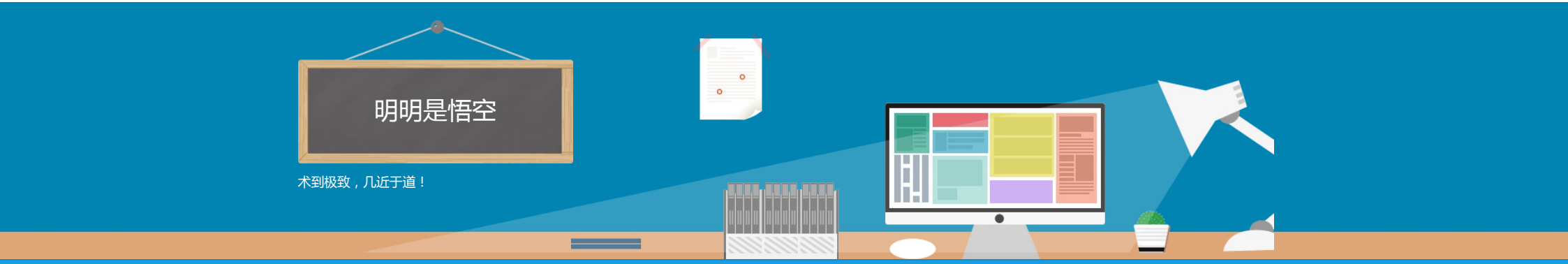

从零开始写一个发送h264的rtsp服务器(下)

转自：http://blog.csdn.net/jychen105/article/details/47012099

一、H264是如何通过rtsp发送的

简单来说，H264就是通过打包到rtp协议的数据部分发送出去的。

H264打包成rtp数据包有三种方式

- 单一封包模式
- 组合封包模式
- 分片模式

  要想弄明白这三种打包方式，必须先弄清楚h264的组成结构，或者叫组成单元。

二、H264结构单元

H264数据流最基本的结构单元叫nalu单元。

H264的nalu单元组成：

```
[start code] + [nalu header] + [nalu paload]
```

- start code: 可以为 001 或者 0001， 也就是点3个字节或者4个字节
- nalu header: 占一个字节

- nalu paload： 长度不定

每一帧画面拥有一个或多个nalu单元，每个nalu单元以start code进行分离

nalu header

nalu header占一个字节，它又分了三个部分：F，NRI，TYPE

```
7 6-5 4-0
F NRI TYPE
```

- F：一般为0
- NRI：指示nalu单元的重要性，不同编码器编出来的H264数据不同
- TYPE：nalu类型

TYPE类型：

类型	定义
0	未定义
1-23	NAL单元
24	STAP-A 单一时间组合包
25	STAP-B 单一时间组合包
26	MTAP-16 多个时间组合包
27	MTAP-24 多个时间组合包
28	FU-A 分片
29	FU-B 分片
30-31	未定义

特别注意的是7这SPS，8为PPS，发送SDP协议包时需进行base64编码
25,26,27,29这四种类型基本不会出现

三、H264的RTP打包

前面说过有三种打包方式：单一封包模式，组合封包模式，分片模式

打包原则

单一封包模式： nalu单元长度小于MTU长度（通常是1500，live555定义的是2400）用此种方式封包
组合封包模式： nalu单元实在太小，多个nalu长度和都小于MTU长度
分片模式：nalu单元长度大于MTU长度

以上封包模式是按nalu长度来分的，同时也完全符合nalu单元中TYPE类型来分。
TYPE1-23 单一封包，24组合封包， 28分片封包

打包细则

单一封包：

```
[RTP header] + [nalu header] + [nalu payload]
```

组合封包：

```
[RTP header]+[STAP-A头(1字节，低5位为24)] +
[第1个nalu长度(2字节)] + [第1个nalu header] + [第1个nalu payload]+
[第2个nalu长度(2字节)] + [第2个nalu header] + [第2个nalu payload]+
```

```
[第N个nalu长度(2字节)] + [第N个nalu header] + [第N个nalu payload]
```

分片模式封包：

此时要切分成多个RTP包

```
[RTP header]+[FU Indicator(1字节)]+[FU header(1字节)]+[部分nalu payload]
```

FU Indicator

```
7 6-5 4-0
F NRI TYPE
```

F, NRI为nalu中的F，NRI
TYPE：固定为28

FU header

```
7 6 5 4-0
S E R TYPE
```

S：开始标志(start)

E：结束标志(end)

R：保留（reserve）

TYPE: nalu中的TYPE

标志	S	E	R
分片开始	1	0	0
分片中间	0	0	0
分片结束	0	1	0

RTP包头的填充

```
typedef struct
{
    /* byte 0 */
    unsigned char csrc_len:4;  /* CC expect 0 */
    unsigned char extension:1; /* X  expect 1, see RTP_OP below */
    unsigned char padding:1;   /* P  expect 0 */
    unsigned char version:2;   /* V  expect 2 */

    /* byte 1 */
    unsigned char payload:7;   /* PT  RTP_PAYLOAD_RTSP */
    unsigned char marker:1;    /* M   expect 1 */

    /* byte 2,3 */
    unsigned short seq_no;     /*sequence number*/

    /* byte 4-7 */
    unsigned  long timestamp;  /*time stamp*/

    /* byte 8-11 */
    unsigned long ssrc;        /* stream number is used here. */
} RTPHeader;/*12 bytes*/
```

各项值填充：

符号	位数	定义	数值
V	2bit	版本号	2

符号	位数	定义	数值
P	1bit	填充位	0
X	1bit	扩展位	0
CC	4bit	CSRC数目	0
M	1bit	标志位	单一封包为1; 分片封包最后一个包为1，其余为0;
PT	7bit	载荷类型	96(h264为96)
SeqNum	16bit	序列号	每发一个包加1
Timestamp	32bit	时间戳	单一封包 +采样率，h264为3600; 分片封包第一个加采样率，后续不变
SSRC	32bit	同步源标识	任意指定，标准是一个MD5算法值，未明
CSRC	0bit	贡献源列表	CC为0，所以此项没有

组合模式的M跟Timestamp未调查清楚，但是可以讨巧，打包的时候不采用组合模式，采用单一模式。

标签: rtsp

明明是悟空
关注 - 4
粉丝 - 203
+加关注

« 上一篇： 从零开始写一个发送h264的rtsp服务器(上)
» 下一篇： centos7安装debuginfo

posted @ 2018-03-19 17:28 明明是悟空 阅读(3006) 评论(1) 编辑 收藏 举报

· [翻译].NET 6 中的 dotnet monitor
· .NET Core 如何配置 TLS Cipher（套件）？
· 记一次 .NET 某智能服装智造系统 内存泄漏分析
· 大学毕业三年的一些经历与思考

最新新闻：
· 泡泡玛特越来越“重” 潮玩需要新故事（2021-12-09 12:11）
· “1.5个”村里人眼里的“张同学”（2021-12-09 12:00）
· “塌房”or过气，都是玲娜贝儿的结局（2021-12-09 11:50）
· 苹果出事了，十年来最严重的一次！（2021-12-09 11:35）
· B面宁德时代：超级工程背后的造富运动（2021-12-09 11:20）
» 更多新闻...

公告

昵称： 明明是悟空
园龄： 9年7个月
粉丝： 203
关注： 4
+加关注

< 2021年12月 >

日	一	二	三	四	五	六
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

搜索

找找看
谷歌搜索

我的标签

Linux(135)

随笔档案

阅读排行榜

评论排行榜

推荐排行榜

3. JAVA 的wait(), notify()与synchronized同步机制(5)

4. jsp放在web-inf下的注意事项(5)

5. 表现层(jsp)、持久层(类似dao)、业务层（逻辑层、service层）、模型（javabean）、控制层（action）(5)

最新评论

1. Re:关于socket——SO_SNDBUF and SO_RECVBUF

"SO_"前缀是指 "套接字选项"，所以是的，这些是每个套接字缓冲区的设置。通常有系统范围内的默认值和最大值。 SO_RCVBUF更容易理解：它是内核分配的缓冲区的大小，用来保存从网络上到达的数据和拥...

--成熟里的秋天

2. Re:sk_buff封装和解封装网络数据包的过程详解

学习了

--咖啡猫二世

3. Re:netfilter的钩子——数据包在内核态得捕获、修改和转发

打扰了，请问您的代码是基于哪一版本的linux内核呀

--RiverGone

4. Re:总结一下数据库的 一对多、多对一、一对一、多对多 关系

通俗易懂，学习了，谢谢大佬们啊

--别说我太单纯

5. Re:总结一下数据库的 一对多、多对一、一对一、多对多 关系

Mark.2021.5.17

--别说我太单纯