

JT同学

创作4年

暂无认证

105

2万+

91万+

46万+

原创

周排名

总排名

总排名

访问量

等级

5440

1861

1016

346

3945

积分

粉丝

获赞

评论

收藏

    

私信

关注

搜索主文章

热门文章

从零基础写一个RTSP服务器（一）RTSP协议讲解

57102

C++ deque的用法与示例

38004

Linux I2C驱动框架（超详细）

35491

我的开源项目-RtspServer

25054

从零基础写一个RTSP服务器（二）RTSP协议的实现

21860

最新评论

我的开源项目-RtspServer

kaifa3n: 我测试 /h264_rtp_server test.h264 也会这样的问题？请问解决了没？ m...
使用mp4v2封装H.264或mp4最简单示例
热心兔斯基: 应该是播放器的原因。换个播放器试试
使用mp4v2封装H.264或mp4最简单示例
ai604233436: h264的标准中，好像默认的设置的90KHz，所以我理解timeScale...
使用mp4v2封装H.264或mp4最简单示例
ai604233436: 有遇到，不知道原因呢。
深入学习Linux摄像头（二）v4l2驱动框架
qq_43745897: 请问往缓存区里面采集图像数据的帧率是多少呢？请问如果缓存区里...

您愿意向朋友推荐“博客详情页”吗？

强烈推荐

推荐

一般般

不推荐

强烈不推荐

最新文章

深入浅出MySQL事务（二）MVCC的实现原理

深入浅出MySQL事务（一）事务隔离

深入浅出MySQL索引（二）InnoDB存储引擎的索引

2020年 4篇

2019年 95篇

2018年 7篇

从零开始写一个RTSP服务器（二）RTSP协议的实现

JT同学

于 2019-08-10 12:58:58 发布

21863 收藏 145

分类专栏:

从零基础写一个RTSP服务器

文章标签:

rtsp

rtsp

音视频

流媒体

从零基础写一个RTS...

专栏收录该内容

187 订阅

10 篇文章

订阅专栏

从零开始写一个RTSP服务器系列

★我的开源项目-RtspServer

从零基础写一个RTSP服务器（一）RTSP协议讲解

从零基础写一个RTSP服务器（二）RTSP协议的实现

从零基础写一个RTSP服务器（三）RTP传输H.264

从零基础写一个RTSP服务器（四）一个传输H.264的RTSP服务器

从零基础写一个RTSP服务器（五）RTP传输AAC

从零基础写一个RTSP服务器（六）一个传输AAC的RTSP服务器

从零基础写一个RTSP服务器（七）多播传输RTP包

从零基础写一个RTSP服务器（八）一个多播的RTSP服务器

从零基础写一个RTSP服务器（九）一个RTP OVER RTSP/TCP的RTSP服务器

从零开始写一个RTSP服务器（二）RTSP协议的实现

文章目录

从零基础写一个RTSP服务器（二）RTSP协议的实现

写在前面

- 一、创建套接字
- 二、解析请求
- 三、OPTIONS响应
- 四、DESCRIBE响应
- 五、SETUP响应
- 六、PLAY响应
- 七、源码
- 八、测试

写在前面

此系列只追求求精，旨在学习RTSP协议的实现过程，不追求复杂完美，所以这里要实现的RTSP服务器为了简单，实现上同一时间只能有一个客户端，下面开始介绍实现过程

在写一个RTSP服务器之前，我们必须知道一个RTSP服务器最简单的包含两部分，一部分是RTSP的交互，一部分是RTP发送，本文先实现RTSP交互过程

一、创建套接字

想一下我们在C/C++输入rtsp://127.0.0.1:8554后发生了什么事？

在这种情况下，vc其实是一个rtsp客户端，当输入这个url后，vc知道目的IP为127.0.0.1，目的端口号为8554，这时vc会发起一个tcp连接取连接服务器，连接成功后就开始发送请求，服务端响应

所以我们要写一个rtsp服务器，第一步肯定是创建tcp服务器

首先创建tcp套接字，绑定端口，监听

创建套接字

```
1 /*
2  * 作者: _JT_
3  * 博客: https://blog.csdn.net/weixin_42462202
4  */
5 serverSocketfd = socket(AF_INET, SOCK_STREAM, 0);
6 setsockopt(serverSocketfd, SOL_SOCKET, SO_REUSEADDR, (const char*)&on, sizeof(on));
```

绑定地址和端口号

```
1 bind(serverSocketfd, (struct sockaddr *)&addr, sizeof(struct sockaddr))
```

这个示例绑定的地址是INADDR_ANY，端口号为8554

开始监听

```
1 listen(serverSocketfd, 10);
```

RTSP服务器传输音频数据和信息使用的是RTP和RTCP，所以我们要为RTP和RTCP创建UDP套接字，并绑定号端口

创建套接字

```
1 serverRtpSocketfd = createUdpSocket();
2 serverRtcpSocketfd = createUdpSocket();
```

绑定端口号

```
1 bindSocketAddr(serverRtpSocketfd, "0.0.0.0", SERVER_RTP_PORT);
2 bindSocketAddr(serverRtcpSocketfd, "0.0.0.0", SERVER_RTCP_PORT);
```

当创建好套接字还有绑定号端口后，就可以接收客户端请求了

开始accept等待客户端连接

```
1 clientfd = accept(serverSocketfd, (struct sockaddr *)&addr, &len);
```

二、解析请求

当rtsp客户端连接成功后就会开始发送请求，服务器这是需要接收客户端请求并开始解析，再采取相应得操作

请求的格式为（详细参考上一篇从零基础写一个RTSP服务器（一）不一样的RTSP协议讲解）

```
1 OPTIONS rtsp://127.0.0.1:8554/live RTSP/1.0\r\n
2 CSeq: 2\r\n
3 \r\n
4
5 DESCRIBE rtsp://127.0.0.1:8554/live RTSP/1.0\r\n
6 CSeq: 3\r\n
7 Accept: application/sdp\r\n
8 \r\n
9
10 SETUP rtsp://127.0.0.1:8554/live/track0 RTSP/1.0\r\n
11 CSeq: 4\r\n
12 Transport: RTP/AVP;unicast;client_port=54492-54493\r\n
13 \r\n
14
15 PLAY rtsp://127.0.0.1:8554/live RTSP/1.0\r\n
16 CSeq: 5\r\n
17 Session: 66334873\r\n
18 Range: npt=0.000-\r\n
19 \r\n
```

这里我们做得最简单，首先解析第一行得到方法，对于OPTIONS、DESCRIBE、PLAY、TEARDOWN我们只解析CSeq。对于SETUP，我们只解析client_port解析出来

所以我们要做的第一步就是解析请求中的信息

接收客户端数据

```
1 recvLen = recv(clientSocketfd, rBuf, BUF_MAX_SIZE, 0);
```

这里实现了一个简单得函数getLineFromBuf，从buf中读取一行(vn)

解析第一行请求得到方法

```
1 sscanf(line, "%s %s %s\r\n", method, url, version);
```

其次解析CSeq

```
1 sscanf(line, "CSeq: %d\r\n", &cseq)
```

目录

从零基础写一个RTSP服务器（二）RTS...

文章目录

写在前面

一、创建套接字

二、解析请求

三、OPTIONS响应

四、DESCRIBE响应

五、SETUP响应

六、PLAY响应

七、源码

八、测试

分类专栏

C/C++

1篇

STL源码剖析

18篇

MySQL

4篇

Linux内核

14篇

分布式

nginx

从零基础写一个RTSP服...

10篇

live555源码分析与应用

9篇

Linux驱动

19篇

如果方法是 `SETUP` 则再解析 `client_port`

```
1 if(!strcmp(method, "SETUP"))
2 {
3     sscanf(line, "Transport: RTP/AVP;unicast;client_port=%d-%d\r\n",
4             &clientRtpPort, &clientRtcpPort);
5 }
```

解析完请求命令后，接下来就是更具不同得方法做不同的响应了，如下

```
1 /*
2  * 作者: _3T_
3  * 博客: https://blog.csdn.net/weixin_42462282
4  */
5 if(!strcmp(method, "OPTIONS"))
6 {
7     handleCmd_OPTIONS();
8 }
9 else if(!strcmp(method, "DESCRIBE"))
10 {
11     handleCmd_DESCRIBE();
12 }
13 else if(!strcmp(method, "SETUP"))
14 {
15     handleCmd_SETUP();
16 }
17 else if(!strcmp(method, "PLAY"))
18 {
19     handleCmd_PLAY();
20 }
21 else if(!strcmp(method, "TEARDOWN"))
22 {
23     handleCmd_TEARDOWN();
24 }
```

三、OPTIONS响应

OPTIONS是客户端向服务器请求可用的方法，我们这里就向客户端回复我们当前可用的方法

```
1 sprintf(sBuf, "RTSP/1.0 200 OK\r\n"
2           "CSeq: %d\r\n"
3           "Public: OPTIONS, DESCRIBE, SETUP, PLAY\r\n"
4           "\r\n",
5           c s e q);
6 send(clientSockfd, sBuf, strlen(sBuf));
```

四、DESCRIBE响应

DESCRIBE是客户端向服务器请求媒体信息，这是服务器需要回复sdp描述文件，这个例子中的媒体是H.264

sdp文件生成

```
1 sprintf(sdp, "v=0\r\n"
2           "o=- 5555 1 IN IP4 %s\r\n"
3           "t=0 0\r\n"
4           "a=control:\r\n"
5           "m=video 0 RTP/AVP 96\r\n"
6           "a=rtpmap:96 H264/90000\r\n"
7           "a=control:track0\r\n",
8           time(NULL), localtime);
```

回复

```
1 sprintf(sBuf, "RTSP/1.0 200 OK\r\n"
2           "CSeq: %d\r\n"
3           "Content-Base: %s\r\n"
4           "Content-type: application/sdp\r\n"
5           "Content-length: %d\r\n\r\n"
6           "%s",
7           c s e q,
8           u r l,
9           strlen(sdp),
10          s d p);
11
12 send(clientSockfd, sBuf, strlen(sBuf));
```

五、SETUP响应

SETUP是客户端请求建立会话连接，并发送了客户端的RTP端口和RTCP端口，那么此时服务端需要回复服务端的RTP端口和RTCP端口

回复

```
1 sprintf(result, "RTSP/1.0 200 OK\r\n"
2           "CSeq: %d\r\n"
3           "Transport: RTP/AVP;unicast;client_port=%d-%d;server_port=%d-%d\r\n"
4           "Session: 66334873\r\n"
5           "\r\n",
6           c s e q,
7           clientRtpPort,
8           clientRtcpPort+1,
9           SERVER_RTP_PORT,
10          SERVER_RTCP_PORT);
11
12 send(clientSockfd, sBuf, strlen(sBuf));
```

其中session id是随便写的，只要保证在多个会话连接时唯一的就行

play响应之后就可以向客户端的RTP端口发送RTP包了

六、PLAY响应

PLAY时客户端向服务器请求播放，这时服务端回复完请求后就开始通过setup过程中创建的udp套接字发送RTP包

回复

```
1 sprintf(result, "RTSP/1.0 200 OK\r\n"
2           "CSeq: %d\r\n"
3           "Range: npt=0.000-\r\n"
4           "Session: 66334873; timeout=60\r\n\r\n",
5           c s e q);
6
7 send(clientSockfd, sBuf, strlen(sBuf));
```

开始发送数据

回复之后，就开始向客户端指定的RTP端口发送RTP包，如何发送RTP包，下篇文章再介绍

七、源码

```
1 /*
2  * 作者: _3T_
3  * 博客: https://blog.csdn.net/weixin_42462282
4  */
5
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <stdint.h>
9 #include <string.h>
10 #include <sys/types.h>
11 #include <sys/socket.h>
12 #include <sys/socket.h>
13 #include <netinet/in.h>
14 #include <arpa/inet.h>
15 #include <time.h>
16
17 #define SERVER_PORT 8554
18 #define SERVER_RTP_PORT 5552
19 #define SERVER_RTCP_PORT 5553
20 #define BUF_MAX_SIZE (1024*1024)
21
22 static int createTcpSocket()
23 {
24     int sockfd;
25     int on = 1;
26
27     sockfd = socket(AF_INET, SOCK_STREAM, 0);
28     if(sockfd < 0)
29         return -1;
30
31     setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, (const char*)&on, sizeof(on));
32
33     return sockfd;
34 }
35
```

```

36 static int createUdpSocket()
37 {
38     int sockfd;
39     int on = 1;
40
41     sockfd = socket(AF_INET, SOCK_DGRAM, 0);
42     if(sockfd < 0)
43         return -1;
44
45     setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, (const char*)&on, sizeof(on));
46
47     return sockfd;
48 }
49
50 static int bindSocketAddr(int sockfd, const char* ip, int port)
51 {
52     struct sockaddr_in addr;
53
54     addr.sin_family = AF_INET;
55     addr.sin_port = htons(port);
56     addr.sin_addr.s_addr = inet_addr(ip);
57
58     if(bind(sockfd, (struct sockaddr*)&addr, sizeof(struct sockaddr)) < 0)
59         return -1;
60
61     return 0;
62 }
63
64 static int acceptClient(int sockfd, char* ip, int* port)
65 {
66     int clientfd;
67     socklen_t len = 0;
68     struct sockaddr_in addr;
69
70     memset(&addr, 0, sizeof(addr));
71     len = sizeof(addr);
72
73     clientfd = accept(sockfd, (struct sockaddr*)&addr, &len);
74     if(clientfd < 0)
75         return -1;
76
77     strcpy(ip, inet_ntoa(addr.sin_addr));
78     *port = ntohs(addr.sin_port);
79
80     return clientfd;
81 }
82
83 static char* getlineFromBuf(char* buf, char* line)
84 {
85     while(*buf != '\n')
86     {
87         *line = *buf;
88         line++;
89         buf++;
90     }
91
92     *line = '\n';
93     ++line;
94     *line = '\0';
95
96     ++buf;
97     return buf;
98 }
99
100 static int handleCmd_OPTIONS(char* result, int cseq)
101 {
102     sprintf(result, "RTSP/1.0 200 OK\r\n"
103                 "CSeq: %d\r\n"
104                 "Public: OPTIONS, DESCRIBE, SETUP, PLAY\r\n"
105                 "\r\n";
106             cseq);
107
108     return 0;
109 }
110
111 static int handleCmd_DESCRIBE(char* result, int cseq, char* url)
112 {
113     char sdp[500];
114     char localIp[100];
115
116     sscanf(url, "rtsp://%[^:]:", localIp);
117
118     sprintf(sdp,
119             "o=- 501d 1 IN IP4 %s\r\n"
120             "t=0 0\r\n"
121             "a=control:\r\n"
122             "m=video 0 RTP/AVP 96\r\n"
123             "a=rtpmap:96 H264/90000\r\n"
124             "a=control:track0\r\n",
125             time(NULL), localIp);
126
127     sprintf(result, "RTSP/1.0 200 OK\r\nCSeq: %d\r\n"
128                 "Content-Base: %s\r\n"
129                 "Content-type: application/sdp\r\n"
130                 "Content-length: %d\r\n\r\n"
131                 "%s"
132                 "cseq, %d\r\n"
133                 "url, %s\r\n"
134                 "sdp");
135
136     return 0;
137 }
138
139 static int handleCmd_SETUP(char* result, int cseq, int clientRtpPort)
140 {
141     sprintf(result, "RTSP/1.0 200 OK\r\n"
142                 "CSeq: %d\r\n"
143                 "Transport: RTP/AVP;unicast;client_port=%d-%d;server_port=%d-%d\r\n"
144                 "Session: 66334873\r\n"
145                 "\r\n";
146             cseq,
147             clientRtpPort,
148             clientRtpPort+1,
149             SERVER_RTP_PORT,
150             SERVER_RTP_PORT);
151
152     return 0;
153 }
154
155 static int handleCmd_PLAY(char* result, int cseq)
156 {
157     sprintf(result, "RTSP/1.0 200 OK\r\n"
158                 "CSeq: %d\r\n"
159                 "Range: npt=0.000-∞\r\n"
160                 "Session: 66334873; timeout=60\r\n\r\n",
161             cseq);
162
163     return 0;
164 }
165
166 static void doClient(int clientSockfd, const char* clientIP, int clientPort,
167                    int serverRtpSockfd, int serverRtcpSockfd)
168 {
169     char method[40];
170     char url[100];
171     char version[40];
172     int cseq;
173     int clientRtpPort, clientRtcpPort;
174     char* rBufPtr;
175     char* rBuf = malloc(BUF_MAX_SIZE);
176     char* sBuf = malloc(BUF_MAX_SIZE);
177     char line[400];
178
179     while(1)
180     {
181         int recvlen;
182
183         recvlen = recv(clientSockfd, rBuf, BUF_MAX_SIZE, 0);
184         if(recvlen == 0)
185             goto out;
186
187         rBuf[recvlen] = '\0';
188         printf("-----C>S-----\n");
189         printf("%s", rBuf);
190
191         /* 解析方法 */
192         bufPtr = getlineFromBuf(rBuf, line);
193         if(sscanf(line, "%s %s %s\r\n", method, url, version) != 3)
194         {
195             printf("parse err\n");
196             goto out;
197         }
198     }
199 }

```

```
200 /* 解析序号号 */
201 bufPtr = getLineFromBuf(bufPtr, line);
202 if(sscanf(line, "%Seq: %d\n", &cseq) != 1)
203 {
204     printf("parse err\n");
205     goto out;
206 }
207
208 /* 如果是SETUP, 那么就用解析client_port */
209 if(!strcmp(method, "SETUP"))
210 {
211     while(1)
212     {
213         bufPtr = getLineFromBuf(bufPtr, line);
214         if(!strcmp(line, "Transport:", strlen("Transport:")))
215         {
216             sscanf(line, "Transport: RTP(AVP;unicast;client_gort=%d-%d\n",
217                 &clientRtpPort, &clientRtcpPort);
218             break;
219         }
220     }
221 }
222
223 if(!strcmp(method, "OPTIONS"))
224 {
225     if(handleCmd_OPTIONS(sBuf, cseq))
226     {
227         printf("failed to handle options\n");
228         goto out;
229     }
230 }
231 else if(!strcmp(method, "DESCRIBE"))
232 {
233     if(handleCmd_DESCRIBE(sBuf, cseq, url))
234     {
235         printf("failed to handle describe\n");
236         goto out;
237     }
238 }
239 else if(!strcmp(method, "SETUP"))
240 {
241     if(handleCmd_SETUP(sBuf, cseq, clientRtpPort))
242     {
243         printf("failed to handle setup\n");
244         goto out;
245     }
246 }
247 else if(!strcmp(method, "PLAY"))
248 {
249     if(handleCmd_PLAY(sBuf, cseq))
250     {
251         printf("failed to handle play\n");
252         goto out;
253     }
254 }
255 {
256     goto out;
257 }
258
259 printf("-----S>C-----\n");
260 printf("Xs", sBuf);
261 send(clientSocket, sBuf, strlen(sBuf), 0);
262 }
263 }
264 out:
265 close(clientSocket);
266 free(rBuf);
267 free(sBuf);
268 }
269
270 int main(int argc, char* argv[])
271 {
272     int serverSocket;
273     int serverRtpSocket, serverRtcpSocket;
274     int ret;
275
276     serverSocket = createTcpSocket();
277     if(serverSocket < 0)
278     {
279         printf("failed to create tcp socket\n");
280         return -1;
281     }
282
283     ret = bindSocketAddr(serverSocket, "0.0.0.0", SERVER_PORT);
284     if(ret < 0)
285     {
286         printf("failed to bind addr\n");
287         return -1;
288     }
289
290     ret = listen(serverSocket, 10);
291     if(ret < 0)
292     {
293         printf("failed to listen\n");
294         return -1;
295     }
296
297     serverRtpSocket = createUdpSocket();
298     serverRtcpSocket = createUdpSocket();
299     if(serverRtpSocket < 0 || serverRtcpSocket < 0)
300     {
301         printf("failed to create udp socket\n");
302         return -1;
303     }
304
305     if(bindSocketAddr(serverRtpSocket, "0.0.0.0", SERVER_RTP_PORT) < 0 ||
306        bindSocketAddr(serverRtcpSocket, "0.0.0.0", SERVER_RTCP_PORT) < 0)
307     {
308         printf("failed to bind addr\n");
309         return -1;
310     }
311
312     printf("rtsp://127.0.0.1:%d\n", SERVER_PORT);
313
314     while(1)
315     {
316         int clientSocket;
317         char clientIp[40];
318         int clientPort;
319
320         clientSocket = acceptClient(serverSocket, clientIp, &clientPort);
321         if(clientSocket < 0)
322         {
323             printf("failed to accept client\n");
324             return -1;
325         }
326
327         printf("accept client;client ip:%s,client port:%d\n", clientIp, clientPort);
328
329         doClient(clientSocket, clientIp, clientPort, serverRtpSocket, serverRtcpSocket);
330     }
331
332     return 0;
333 }
```

八、测试

编译运行源码，打开vc，输入rtsp://127.0.0.1:8554，点击开始播放，可以看到控制台会打印出交互过程，或是用Wireshark抓包

本篇文章到这里结束，至此完成了RTSP协议的交互部分，在PLAY之后并没有开始发送RTP包，所以暂时还看不到视频，究竟如何发送RTP包，请看下一篇文章

一些rtsp实现的开源代码 2013-07-10 22:25 / 工业 编程 / 共 414字 / 字号 小 中 大 / 暂无评论 * live.com C/S C++ http://www.live555.com * darwin ...

RTSP分析与实现原理 06-01 本课程是RTSPONVIFvG828181系列教程之一，通过分析RTSP协议指令交互过程，结合RTSP媒体库（由Implem完全自主实现的RTSP RTP/RTCP媒体库...

rtsp端口号_最详细的流媒体传输协议rtsp协议详解_知乎... 10-9 流媒体传输协议rtsp协议详解 参阅 RTSP协议详解和分析从零开始写一个RTSP服务器(一)RTSP协议讲解关于RTSP RTP RTCP协议的深刻初步介绍 rtsp...

RTSP协议的一些分析(三)——简单的rtsp服务器的实现_风... 6-14 RTSP服务器有两个部分组成一个处理RTSP的交互一个处理RTP数据的传输,本文主要实现RTSP服务的交互过程。从PTSP协议介绍中我们可以了解到,当rtsp...

从零开始写一个RTSP服务器(三) RTP传输H.264 27 同学的博文 175+ 从零开始写一个RTSP服务器(一) 不一样的RTSP协议讲解 从零开始写一个RTSP服务器(二) RTSP协议的实现 从零...

linux下rtsp推流软件,Toybrick-开源社区-人工智能-RK3399Pro入门教程 (10) RTSP推流介绍... 351 你啊,视频无法打开提示如下: [root@localhost IPCamera]# ./RTSPServer"rtsp://192.168.1.100:554/stream" from the file "test.264"Play this stream using the ...

RTSP协议详解及实例分析_king_weng的博客_rtsp 参数 9-22 安全 RTSP使用网络安全机制,独立于传输 RTSP可使用不可靠数据报协议(UDP),可靠数据报协议(RDP),如要实现应用级可靠,可使用可靠流协议,多服务...

从零开始写一个RTSP服务器(六)一个传输AAC的RTSP服务器 10-7

*我的开源项目-RtpServer 从零开始写一个RTSP服务器(一)RTSP协议讲解 从零开始写一个RTSP服务器(二)RTSP协议的实现 从零开始写一个RTSP服...		
rtsp协议 /rtsp 开源服务器之live555	weixin_39955700的博客	2
Live555是一个实现了RTSP协议的开源流媒体框架, Live555包含RTSP服务器端的实现以及 RTSP客户端的实现。 Live555可以将若干种格式的流媒体文件或...		418
RTSP服务器: RTSP协议实现 (与Vc交互)	weixin_43796767的博客	418
RTSP协议介绍, 请参考: RTSP传输过程分析 协议实现过程: 1) 创建TCP连接, 与VLC客户端连接。 工作流程: 创建tcp套接字, 绑定端口, 监听 2...		2...
从头写rtsp服务器-RTSP协议的解析_achow4969的博客		10-11
1.首先服务器收到客户端连接请求,生产 一个RtcpClientConnection对象_RtcpClientConnection定义详见从头写rtsp服务器-模块的划分 int rtsp...v_accept(ne...		
从零开始写一个RTSP服务器 (一) RTSP协议讲解 热门推荐	JT同学的博客	574
从零开始写一个RTSP服务器系列 从零开始写一个RTSP服务器 (一) 不一样的RTSP协议讲解 从零开始写一个RTSP服务器 (二) RTP传输H.264(待写)
RTSP协议的实现	一叶知秋	603
写在前面 目前正在学习RTSP协议。偶然间发现在这篇文章非常好, 故转载学习使用: RTSP协议的实现 本文所有代码均来自此篇文章。在此基础上做了...		...
一个RtpServer的设计与实现和RTSP2.0简介	weixin_33824363的博客	267
一个RtpServer的设计与实现和RTSP2.0简介	前段时间着手实现了一个RTSP Server, 能够正常实现多播RTSP流的直播播放。因项目需要, 只做了...	...
Fast-RTPS、RTPS标准的实现_zip		09-18
Fast-RTPS、RTPS标准的实现 eProxima快速 RTPS system - config - 快速 RTPS 是 RTPS (实时发布订阅服务器) 协议的一个 C 实现。它通过对象管理组...		...
流媒体开发之开源项目 rtsp服务器-何工的eseasy rtsp服务器	weixin_33845881的博客	87
http://www.esayrtsp.org/ 转载于 https://www.cnblogs.com/pengkunfan/p/4360772.html		
搭建RTSP流媒体服务器的三种方式	fang_lovest_yang的博客	4339
主要用于测试目的。 系统是windows, 使用的是docker desktop 3.5.1 1. rtsp-simple-server 官网: https://github.com/aler9/rtsp-simple-server (1) 下载...		...
RTSPServer开源库测试	程序猿的专栏	677
前2年搞过一个C端的RTSPServer项目。原项目中使用EasyPcCamera, 这个项目目前已经改名为EasyRTSPServer, 提供了一套完整的直播推送方案。这...		...
从零开始写一个RTSP服务器 (五) RTP传输AAC	JT同学的博客	8058
从零开始写一个RTSP服务器系列 从零开始写一个RTSP服务器 (一) 不一样的RTSP协议讲解 从零开始写一个RTSP服务器 (二) RTP协议的实现 从事...		...
首视频学习之rtsp推送学习1 (rtspserver开源库example运行及流程梳理)	yun6853992的博客	1740
工作需要实现一个rtsp的推送拉流业务流程。对开源项目rtspserver进行学习与理解。 参考系列rtspserve的文章: 我的开源项目-RtpServer_JT同学的博...		...
RTCP协议介绍	先自制, 不成功	4000
 RTCP需要 实时传输控制协议(R eal-t ime C ontrol P rotocol , RTCP) 与RTP 共同定义在1996 年提出的RFC 1889 中, 是和 RTP
rtsp服务器: C语言实现rtsp服务器 最新发布	xiaocing132的专栏	751
class RtcpServer { public: RtcpServer(void); ~RtcpServer(void); public: int start_server(unsigned short port); /*RTSP OVER HTTP*/ int http_cmd_serve...		...
开源RTSP 流媒体服务器	coder	174
Columbia University rtspd - Supports serving of .au files via RTSP for Solaris, FreeBSD 3.x, Linux, Windows NT, Darwin Streaming Server - The open-...		...
我的开源项目-RtpServer	JT同学的博客	274
我的开源项目-RtpServer 文章目录我的开源项目-RtpServer一、项目介绍二、功能介绍三、开发环境四、使用方法4.1 传输音视频文件4.2 采集V4L2摄...		...
开源的rtsp实现	weixin_33924220的博客	437
开源的rtsp实现	2006-5-4 1:40	最近在搞rt...
*** 感谢您的支持 ***		
-- by BeagleTam		
“相关推荐”对你有帮助么?		
😄 非常有帮助 😊 有帮助 😐 一般 😞 没帮助 😡 非常没帮助		
©2022 CSDN 皮肤主题: 数字20 设计师: CSDN官方博客 返回顶部		
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 京公网安备11010802020076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 不良信息举报中心 家长监护 网络110报警服务 中国互联网络信息中心 Chrome商店下载 知乎 哔哩哔哩 版权与免责声明 版权申诉 出版物许可证 营业执照 ©1999-2022北京创新乐知网络技术有限公司		
🔍 JT 同学 关注	👍 59 🗨 145 ⭐ 16 📄 16 📁 16	专栏目录

