



http://ticktick.blog.51cto.com 【复制】 【订阅】

博客 | 图库 | 写博文 | 帮助

首页 | 嵌入式 | 软件编程 | Linux学习与应用 | Android开发 | 视频相关技术 | 工具及调试笔记 | 生活随笔 | 其他

tickTick 的BLOG

相关视频课程

更多



写留言

去学院学习

发消息

加好友链接

进家园 加好友



深入浅出之-JavaSE基础教程(共70课时)

114715人学习



PK技术实现网络安全(共28课时)

12996人学习



新版CCNP交换方向视频教程(共11课时)

134人学习

博主的更多文章>>

原创 ortp库使用入门

2010-07-07 20:05:26

标签: 示例 rtp 介绍 ortp 流媒体传输

原创作品，允许转载，转载时请务必以超链接形式标明文章 [原始出处](#)、作者信息和本声明。否则将追究法律责任。  
任。<http://ticktick.blog.51cto.com/823160/345642>

博客统计信息

51CTO推荐博客

用户名: tickTick

文章数: 92

评论数: 299

访问量: 599168

无忧币: 2187

博客积分: 2810

博客等级: 7

注册日期: 2009-06-13

热门专题

更多>>



Exchange Server 2013 服务器配置详解

阅读量: 1240



从零开始学高德地图JS API

阅读量: 1596



Exchange 2013SP1和O365混合部署详解

阅读量: 871

热门文章

c++构造函数详解

undefined reference问题..

c++异常处理机制示例及讲解

Doxygen详细介绍(三)(..

C++串口编程实例

入门视频采集与处理(学..

谈谈RTP传输中的负载类型..

ortp库使用入门

我们知道，RTP（Real-timeTransportProtocol）是用于Internet上针对多媒体数据流的一种传输协议，做流媒体传输方面的应用离不开RTP协议的实现及使用，为了更加快速地在项目中应用RTP协议实现流媒体的传输，我们一般会选择使用一些RTP库，例如使用c++语言编写的JRTPLIB库，网上关于RTP协议以及JRTPLIB库的介绍已经很多了，在此我也不再赘述，文本主要介绍实现了RTP协议的另一种开源库——ORTP库，这个库是纯使用c语言编写，由于我们的项目是基于Linux下的c语言编程，故我们选择了ortp作为我们的第三方库，在此我也对该库进行一个简单地介绍，希望对其他ortp的初学者有所帮助。

## 一、简介

ORTP是一个支持RTP以及RFC3550协议的库，有如下的特性：

- （1）使用C语言编写，可以工作于windows, Linux, 以及 Unix平台
- （2）实现了RFC3550协议，提供简单易用的API。支持多种配置，RFC3551为默认的配置。
- （3）支持单线程下的多个RTP会话，支持自适应抖动处理。
- （4）基于GPL版权声明。

ORTP可以在其官方网站上（[http://www.linphone.org/index.php/eng/code\\_review/ortp](http://www.linphone.org/index.php/eng/code_review/ortp)）下载，下载解压后得到ORTP的源码包和示例程序（tests）。其帮助文档在docs目录下，也可以在<http://download.savannah.gnu.org/releases/linphone/ortp/docs/>在线查看。

关于ORTP的资料并不多，主要是其源码、帮助文档以及示例程序，关于示例程序说明如下：  
rtpprecv.c 和rtppsend.c 展示了如何接收和发送单RTP数据流。  
mrtprecv.c mrtpsend.c 展示了如何同时接收和发送多个RTP数据流。

## 二、主要函数介绍

### rtp\_session\_init

函数原型: void rtp\_session\_init (RtpSession \* session, int mode)

函数功能: 执行rtp会话的一些必要的初始化工作

参数含义:

session: rtp会话结构体，含有一些rtp会话的基本信息

mode: 传输模式，有以下几种，决定本会话的一些特性。

RTP\_SESSION\_RECVONLY: 只进行rtp数据的接收

搜索BLOG文章

搜索

最近访客



Bill\_Hoo



卧听潇雨



ruiqi..



王者鹤兮



xhb29



OTExc..



benji..



ddl0601



睦文峰



majestli



8207531



草蜢

最新评论

niceheart: 回复 tickTick: 楼主可否给我一个..

tickTick: 回复 niceheart: 这个代码只是用..

niceheart: 请问楼主在项目是如何去用使用? ..

kangear: 你这个似乎是NDK编程的, Android系..

kkdluf: 太感谢楼主了

51CTO推荐博文

更多>>

【云图】如何创建云图（云存储）

性能分析工具-PerfView

RMI原理揭秘之安全通信

使用“忽略授权表”参数登录多实..

ASP.NET MVC学前篇之扩展方法、链..

【web开发】之常用上传下载代码

MHA高可用部署及测试

修改MFC OCX的GUID

基于Python的HTTPS协议模拟登陆+..

[android底层]jni中获取自定义的..

Eclipse常用的一些设置

友情链接

錦戩疏鍾版氮錦氫慙

肖骁的博客

Dian团队官方主页

RTP\_SESSION\_SENDOONLY：只进行rtp数据的发送

RTP\_SESSION\_SENDCV：可以进行rtp数据的接收和发送

执行的操作：

1. 设置rtp包缓冲队列的最大长度
2. 根据传输模式设置标志变量的值
3. 随机产生SSRC和同步源描述信息
4. 传入全局的av\_profile，即使用默认的profile配置
5. 初始化rtp包缓冲队列
6. 发送负载类型默认设置为0（pcmu音频），接收负载类型默认设置为-1（未定义）
7. 将session的其他成员的值均设置一个默认值。

## rtp\_session\_set\_scheduling\_mode

函数原型：**void rtp\_session\_set\_scheduling\_mode (RtpSession \* session, int yesno)**

函数功能：RtpScheduler管理多个session的调度和收发的控制，本函数设置是否使用该session调度管理功能。

参数含义：

**session：**rtp会话结构体

**yesno：**是否使用rtp session的系统调度功能

说明：

如果yesno为1，则表明使用系统的session调度管理功能，意味着可以使用以下功能：

1. 可以使用session\_set\_select在多个rtp会话之间进行选择，根据时间戳判定某个会话是否到达了收发的时间。
2. 可以使用rtp\_session\_set\_blocking\_mode()设置是否使用阻塞模式来进行rtp包的发送和接收。

如果yesno为0，则表明该会话不受系统管理和调度。

关于rtp session的管理和调度，由全局的变量**RtpScheduler \* \_\_ortp\_scheduler**来负责，该变量必须通过**ortp\_scheduler\_init()**来进行初始化操作。

## rtp\_session\_set\_blocking\_mode

函数原型：**void rtp\_session\_set\_blocking\_mode (RtpSession \* session, int yesno)**

函数功能：设置是否使用阻塞模式，

参数含义：

**session：**rtp会话结构体

**yesno：**是否使用阻塞模式

说明：

阻塞模式只有在scheduling mode被开启的情况下才能使用，本函数决定了rtp\_session\_recv\_with\_ts() 和 rtp\_session\_send\_with\_ts()两个函数的行为，如果启用了阻塞模式，则rtp\_session\_recv\_with\_ts()会一直阻塞直到接收RTP包的时间点到达（这个时间点由该函数参数中所定义的时间戳来决定），当接收完RTP数据包后，该函数才会返回。同样，rtp\_session\_send\_with\_ts()也会一直阻塞直到需要被发送的RTP包的时间点到达，发送结束后，函数才返回。

## rtp\_session\_signal\_connect

函数原型：**int rtp\_session\_signal\_connect (RtpSession \* session, const char \*signal, RtpCallback cb, unsigned long user\_data)**

函数功能：本函数提供一种方式，用于通知应用程序各种可能发生的RTP事件（信号）。可能通过注册回调函数的形式来实现本功能。

参数含义：

**session：**rtp会话结构体

**signal：**信号的名称

**cb：**回调函数

**user\_data：**传递给回调函数的数据

返回值：0表示成功，-EOPNOTSUPP表示信号名称不存在，-1表示回调函数绑定错误

说明：

信号的名称必须是以下字符串中的一种：

"ssrc\_changed"：数据流的同步源标识改变

"payload\_type\_changed"：数据流的负载类型改变

"telephone-event\_packet"：telephone-event RTP包(RFC2833)被接收

"telephone-event"：telephone event 发生

"network\_error"：网络错误产生，传递给回调函数的是描述错误的字符串（const char \*型）或者错误码（int型）

"timestamp\_jump"：接收到的数据包发生了时间戳的跳跃。

要取消事件（信号）的监听，可以使用下面这个函数

```
int rtp_session_signal_disconnect_by_callback ( RtpSession * session, const char * signal_name, RtpCallback cb )
```

## rtp\_session\_set\_local\_addr

函数原型：`int rtp_session_set_local_addr( RtpSession * session, const char * addr,int port)`

函数功能：设置本地rtp数据监听地址

参数含义：

**session**：rtp会话结构体

**addr**：本地IP地址，例如127.0.0.1，如果为NULL，则系统分配0.0.0.0

**port**：监听端口，如果设置为-1，则系统为其自动分配端口

返回值：0表示成功

说明：

如果是RTP\_SESSION\_SENDOONLY（只发送）型会话，则不需要进行本设置，而必须设置

`rtp_session_set_remote_addr()` 来设置远程目的地址。

如果采用了系统自动分配监听端口，则可以通过`int rtp_session_get_local_port(const RtpSession *session)` 来获取系统分配的监听端口号。

## rtp\_session\_set\_remote\_addr

函数原型：`int rtp_session_set_remote_addr (RtpSession * session, const char * addr, int port)`

函数功能：设置RTP发送的目的地址

参数含义：

**session**：rtp会话结构体

**addr**：目的IP地址

**port**：目的地址的监听端口号

返回值：0表示成功

## rtp\_session\_set\_send\_payload\_type

函数原型：`int rtp_session_set_send_payload_type (RtpSession * session, int paytype)`

函数功能：设置RTP发送数据的负载类型

参数含义：

**session**：rtp会话结构体

**paytype**：负载类型

返回值：0表示成功，-1表示负载未定义

说明：

负载类型在`payloadtype.h`文件中有详细的定义，RTP接收端有着类似的负载类型设置函数，`int`

`rtp_session_set_rcv_payload_type ( RtpSession * session, int paytype )`，注意，发送的负载类型必须与接收的负载类型一致才能正常完成收发。

## rtp\_session\_send\_with\_ts

函数原型: `int rtp_session_send_with_ts (RtpSession * session, const char * buffer, int len, uint32_t userts)`

函数功能: 发送RTP数据包

参数含义:

**session:** rtp会话结构体

**buffer:** 需要发送的RTP数据的缓冲区

**len:** 需要发送的RTP数据的长度

**userts:** 本RTP数据包的时间戳

返回值: 成功发送到网络中的字节数

说明:

发送RTP数据需要自己管理时间戳的递增, 每调用一次本函数, 请根据实际情况对userts进行递增, 具体递增的规则见RTP协议中的说明。

例如: 如果发送的是采样率为90000Hz的视频数据包, 每秒25帧, 则时间戳的增量为:  $90000/25 = 3600$

时间戳的起始值为随机值, 建议设置为0。

## rtp\_session\_rcv\_with\_ts

函数原型: `int rtp_session_rcv_with_ts (RtpSession * session, char * buffer, int len, uint32_t time, int * have_more)`

函数功能: 接收RTP数据包

参数含义:

**session:** rtp会话结构体

**buffer:** 存放接收的RTP数据的缓冲区

**len:** 期望接收的RTP数据的长度

**time:** 期望接收的RTP数据的时间戳

**have\_more:** 标识接收缓冲区是否还有数据没有传递完。当用户给出的缓冲区不够大时, 为了标识缓冲区数据未取完, 则have\_more指向的数据为1, 期望用户以同样的时间戳再次调用本函数; 否则为0, 标识取完。

## rtp\_session\_destroy

【原型】: `void rtp_session_destroy(RtpSession *session)`

【功能】: 摧毁rtp会话对象, 释放资源

【参数】: session已经创建的RTP会话对象

## 三、程序示例

下面, 我简单地通过程序演示了怎么使用ortp进行rtp数据包的发送, 接收端的程序待以后有时间再整理出来吧。

注: 示例代码我已经整理出来了, 见博文: 《[ortp编程示例代码](#)》

```
01.  //////////////////////////////////////
02.  /// COPYRIGHT NOTICE
03.  /// Copyright (c) 2009, 华中科技大学ticktick Group
04.  /// All rights reserved.
05.  ///
06.  /// @file    ortpSend.c
07.  /// @brief   ortpSend的测试
08.  ///
09.  /// 本文件示例使用ortp库进行rtp数据包的发送
10.  ///
11.  /// @version 1.0
12.  /// @author  tickTick
13.  /// @date    2010/07/07
14.  /// @E-mail  lujun.hust@gmail.com
15.  ///
16.  /// 修订说明: 创建文件
17.  //////////////////////////////////////
18.
19.  #include <ortp/ortp.h>
20.  #include <signal.h>
21.  #include <stdlib.h>
22.
23.  #ifndef _WIN32
24.  #include <sys/types.h>
25.  #include <sys/time.h>
26.  #include <stdio.h>
27.  #endif
28.
29.  // 时间戳增量
```

```

30. #define TIME_STAMP_INC 160
31. #define BYTES_PER_COUNT 65535

01. // 时间戳
02. uint32_t g_user_ts;
03.
04. /** 初始化
05. *
06. * 主要用于对ortp以及其它参数进行初始化
07. * @param: char * ipStr 目的端IP地址描述串
08. * @param: iint port 目的端RTP监听端口
09. * @return: RtpSession * 返回指向RtpSession对象的指针, 如果为NULL, 则初始化失败
10. * @note:
11. */
12. RtpSession * rtpInit(char * ipStr, int port)
13. {
14.     // Rtp会话对象
15.     RtpSession *session;
16.     char *ssrc;
17.
18.     // 时间戳初始化
19.     g_user_ts = 0;
20.
21.     // ortp的一些基本初始化操作
22.     ortp_init();
23.     ortp_scheduler_init();
24.     // 创建新的rtp会话对象
25.     session=rtp_session_new(RTP_SESSION_SENDFONLY);
26.
27.     rtp_session_set_scheduling_mode(session,1);
28.     rtp_session_set_blocking_mode(session,1);
29.     // 设置远程RTP客户端的IP和监听端口 ( 即本rtp数据包的发送目的地址 )
30.     rtp_session_set_remote_addr(session, ipStr, port);
31.     // 设置负载类型
32.     rtp_session_set_payload_type(session,0);
33.
34.     // 获取同步源标识
35.     ssrc=getenv( "SSRC");
36.     if (ssrc!=NULL)
37.     {
38.         printf("using SSRC=%i.\n", atoi(ssrc));
39.         rtp_session_set_ssrc(session, atoi(ssrc));
40.     }
41.
42.     return session;
43. }
44.
45. /** 发送rtp数据包
46. *
47. * 主要用于发送rtp数据包
48. * @param: RtpSession *session RTP会话对象的指针
49. * @param: const char *buffer 要发送的数据的缓冲区地址
50. * @param: int len 要发送的数据长度
51. * @return: int 实际发送的数据包数目
52. * @note: 如果要发送的数据包长度大于BYTES_PER_COUNT, 本函数内部会进行分包处理
53. */
54. int rtpSend(RtpSession *session, const char *buffer, int len)
55. {
56.     int curOffset = 0;
57.     int sendBytes = 0;
58.     int clockslide=500;
59.     // 发送包的个数
60.     int sendCount = 0;
61.
62.     ortp_message("send data len %i\n ", len);
63.
64.     // 是否全部发送完毕
65.     while(curOffset < len )
66.     {
67.         // 如果需要发送的数据长度小于等于阈值, 则直接发送
68.         if( len <= BYTES_PER_COUNT )
69.         {
70.             sendBytes = len;
71.         }
72.         else
73.         {
74.             // 如果当前偏移 + 阈值 小于等于 总长度, 则发送阈值大小的数据
75.             if( curOffset + BYTES_PER_COUNT <= len )
76.             {
77.                 sendBytes = BYTES_PER_COUNT;
78.             }
79.             // 否则就发送剩余长度的数据
80.             else
81.             {
82.                 sendBytes = len - curOffset;
83.             }
84.         }
85.     }
86.
87.     ortp_message("send data bytes %i\n ", sendBytes);
88.
89.     rtp_session_send_with_ts(session, (char *) (buffer+curOffset), sendBytes, g_user_ts);
90.
91.     // 累加
92.     sendCount ++;
93.     curOffset += sendBytes;
94.     g_user_ts += TIME_STAMP_INC;

```

```
95.
96.         // 发送一定数据包后休眠一会
97.         if (sendCount%10==0)
98.         {
99.             usleep(20000);
100.        }
101.    }
102.    return 0;
103. }
104.
105. /** 结束ortp的发送, 释放资源
106.  *
107.  * @param: RtpSession *session RTP会话对象的指针
108.  * @return: 0表示成功
109.  * @note:
110.  */
111. int rtpExit(RtpSession *session)
112. {
113.     g_user_ts = 0;
114.
115.     rtp_session_destroy(session);
116.     ortp_exit();
117.     ortp_global_stats_display();
118.
119.     return 0;
120. }
121.
122. // 主函数, 进行测试
123. int main()
124. {
125.     // 待发送的数据缓冲区
126.     char * pBuffer = "123445356234134234532523654323413453425236244123425234";
127.
128.     RtpSession * pRtpSession = NULL;
129.     // 向 (192.201.0.51, 8000) 目的地址发送rtp包
130.     pRtpSession = rtpInit("192.201.0.51",8000);
131.     if(pRtpSession==NULL)
132.     {
133.         printf("error rtpInit");
134.         return 0;
135.     }
136.
137.     // 循环发送
138.     while(1)
139.     {
140.         if( rtpSend(pRtpSession,pBuffer,20) != 0)
141.         {
142.             printf("error rtpInit");
143.             break;
144.         }
145.         usleep(10000);
146.         printf("sleep");
147.     }
148.
149.     // 退出
150.     rtpExit(pRtpSession);
151.
152.     return 0;
153. }
```

本文出自“对影成三人”博客, 请务必保留此出处<http://ticktick.blog.51cto.com/823160/345642>

分享至:

4

收藏 +

ajaxhe、jiangroron、underlight 4人 了这篇文章

类别: 软件编程 | 阅读(13535) | 评论(16) | [返回博主主页](#) | [返回博客首页](#)

[上一篇 技术培训的过程中,你们学到了什么](#) [下一篇 谈谈RTP传输中的负载类型和时间戳](#)



关注51CTO博客微信  
获得每日精选推荐文章  
微信号:blog51cto



关注51CTO官方微信  
我们提供不一样的东西  
微信号: weixin51cto

#### 相关文章

[RTP与RTCP协议介绍](#)

[Visio 2010各版本介绍](#)

[新一代无线Mesh接入技术介绍及其应用说明](#)


[服务器介绍系列-软件篇](#)

[实时传输协议RTP \( Real-Time Transport Prot..](#)

#### 文章评论

<< 1 2 >> 页数 ( 1/2 )

[1楼]	 [匿名]firstlight	<a href="#">回复</a>
	贴了好多代码呢。。。 ortp包含rtcp控制不？ 推荐jrtpplib，这个纯Cpp实现的，包含rtcp	2010-07-08 09:06:31
<hr/>		
[2楼]	楼主  tickTick	<a href="#">回复</a>
	回复 firstlight:[1楼] 恩，ortp应该是有rtcp控制的，不能我还没用到，上个项目用的就是jrtpplib，很好很强大，但这个项目是用c语言开发，所以没有用jrtpplib库.....	2010-07-08 09:51:07
<hr/>		
[3楼]	 [匿名]_ant_	<a href="#">回复</a>
	实在太感谢了~~！	2010-10-11 16:28:22
<hr/>		
[4楼]	 [匿名]51CTO游客	<a href="#">回复</a>
	代码写得真漂亮！	2010-12-03 19:28:48
<hr/>		
[5楼]	 jiangroron	<a href="#">回复</a>
	如果我想每次发送一帧视频数据，大约有3000多字节，只要调用一次rtp_session_send_with_ts()函数吗？是否需要将这一帧数据根据MTU的值划分为N段，再调用N次rtp_session_send_with_ts()函数？	2011-03-01 14:17:55
<hr/>		
[6楼]	楼主  tickTick	<a href="#">回复</a>
	回复 jiangroron:[5楼] 理论上应该进行划分，直接调用rtp_session_send_with_ts()的话，rtp模块应该会报错，你可以试试。	2011-03-01 15:06:03
<hr/>		
[7楼]	 jiangroron	<a href="#">回复</a>
	请问用ORTP发送一个视频帧时，当到了一帧的末尾时，那个包的RTP包头需要自己设置标记位M为1吗？如果需要，是通过哪个函数设置标志位M的值呢？谢谢！	2011-03-02 19:58:21
<hr/>		
[8楼]	楼主  tickTick	<a href="#">回复</a>
	回复 jiangroron:[7楼] 我在使用它的时候没有设置过M位，应该是不需要的。	2011-03-03 18:36:21
<hr/>		
[9楼]	 [匿名]shang	<a href="#">回复</a>
	楼主我测试了你的代码，在linux下，虽然感觉上传输了，但是通过抓包工具，并没有抓到到包，这个你能解释下吗，我怀疑是ip地址的格式问题，用socket编程的时候，地址都得通过inet_aton转换一下，你解决这个问题了吗	2011-05-09 17:53:49
<hr/>		
[10楼]	楼主  tickTick	<a href="#">回复</a>
	回复 shang:[9楼] ortp关于设置ip地址的函数是rtp_session_set_remote_addr，该函数的文档是这么写的：*@addr: a local IP address in the xxx.xxx.xxx.xxx form.	2011-05-11 12:35:32
<hr/>		
	所以说，不需要通过inet_aton转换。你使用127.0.0.1这个IP试试。	
<hr/>		
[11楼]	 [匿名]Mr.liu	<a href="#">回复</a>
	你好，我把ORTP协议移植到了vxworks上面了，但是现在我写完应用程序后发现，rtp_session_rtp_rcv好像是个死循环似的，收到数据就出不来了，不知道博主知道是什么原因不？	2011-07-26 16:41:15
<hr/>		

[12楼]楼主  tickTick 回复

2011-07-27 08:42:53

回复 Mr.liu:[11楼]  
没有rtp\_session\_rtp\_rcv这个函数吧？你是说：rtp\_session\_rcv\_with\_ts吧？rtp\_session\_rcv\_with\_ts函数需要填入一个时间戳，该时间戳如果填错了的话，该函数会一直等待你填入的错误的时间戳的rtp数据包过来，可能就会导致一直出不来。

[13楼]  [匿名]Mr liu 回复

2011-07-27 10:40:22

这个时间戳 我看例子上面都写的是0 所以我填写的也是0

[14楼]楼主  tickTick 回复

2011-07-27 16:07:28

回复 Mr liu:[13楼]  
ortp自带的例子(src/tests/)不是用的0啊，推荐看我的另一篇文章：<http://ticktick.blog.51cto.com/823160/350142>

[15楼]  underalight 回复

2012-03-29 14:45:42

仔细学习一下把

<< 1 2  >> 页数 ( 1/2 )

**发表评论**      [2014 WOT全球软件技术峰会【火热抢票中】](#)

昵 称： [登录](#) [快速注册](#)

验证码： 请点击后输入验证码 [博客过2级，无需填写验证码](#)

内 容：