原 RTMPDump（libRTMP）源代码分析 2：解析RTMP地址——RTMP_ParseURL()

2013年10月22日 21:04:57　阅读数：16626

==========

RTMPdump(libRTMP) 源代码分析系列文章：

RTMPdump 源代码分析 1： main()函数

RTMPDump (libRTMP) 源代码分析2：解析RTMP地址——RTMP_ParseURL()

RTMPdump (libRTMP) 源代码分析3： AMF编码

RTMPdump (libRTMP) 源代码分析4： 连接第一步——握手 (HandShake)

RTMPdump (libRTMP) 源代码分析5： 建立一个流媒体连接 (NetConnection部分)

RTMPdump (libRTMP) 源代码分析6： 建立一个流媒体连接 (NetStream部分 1)

RTMPdump (libRTMP) 源代码分析7： 建立一个流媒体连接 (NetStream部分 2)

RTMPdump (libRTMP) 源代码分析8： 发送消息 (Message)

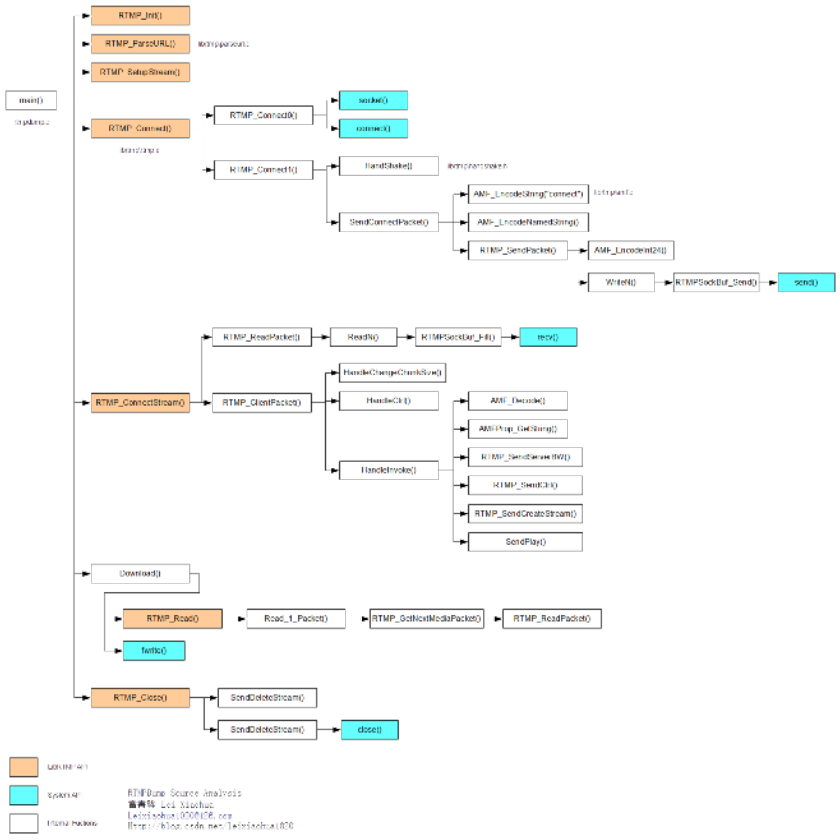
RTMPdump (libRTMP) 源代码分析9： 接收消息 (Message) (接收视音频数据)

RTMPdump (libRTMP) 源代码分析10： 处理各种消息 (Message)

==========

函数调用结构图

RTMPDump (libRTMP)的整体的函数调用结构图如下图所示。



单击查看大图

详细分析

之前分析了一下RTMPDump的Main()函数，其中获取RTMP流媒体数据很重要的前提是RTMP的URL的解析。如果没有这一步，那程序在强大也是白搭。现在来解析一

下这个函数吧：RTMP_ParseURL()。
下面首先回顾一下RTMP的URL的格式：

rtmp://localhost/vod/mp4:sample1_1500kbps.f4v

“://”之前的是使用的协议类型，可以是rtmp，rtmpt，rtmps等

之后是服务器地址

再之后是端口号（可以没有，默认1935）

在之后是application的名字，在这里是“vod”

最后是流媒体文件路径。

关于URL就不多说了，可以查看相关文档，下面贴上注释后的代码（整个parseurl.c）：

```cpp
/*
 * 本文件主要包含了对输入URL的解析
 */
#include "stdafx.h"
#include <stdlib.h>
#include <string.h>

#include <assert.h>
#include <ctype.h>

#include "rtmp_sys.h"
#include "log.h"

/*解析URL，得到协议名称(protocol)，主机名称(host)，应用程序名称(app)
 *
 */
int RTMP_ParseURL(const char *url, int *protocol, AVal *host, unsigned int *port,
    AVal *playpath, AVal *app)
{
    char *p, *end, *col, *ques, *slash;

    RTMP_Log(RTMP_LOGDEBUG, "Parsing...");

    *protocol = RTMP_PROTOCOL_RTMP;
    *port = 0;
    playpath->av_len = 0;
    playpath->av_val = NULL;
    app->av_len = 0;
    app->av_val = NULL;

    /* 字符串解析 */
    /* 查找“://”  */
    //函数原型：char *strstr(char *str1, char *str2);
    //功能：找出str2字符串在str1字符串中第一次出现的位置（不包括str2的串结束符）。
    //返回值：返回该位置的指针，如找不到，返回空指针。
    p = strstr((char *)url, "://");
    if(!p) {
        RTMP_Log(RTMP_LOGERROR, "RTMP URL: No :// in url!");
        return FALSE;
    }
    {
    //指针相减，返回“://”之前字符串长度len
    int len = (int)(p-url);
    //获取使用的协议
    //通过比较字符串的方法
    if(len == 4 && strncasecmp(url, "rtmp", 4)==0)
        *protocol = RTMP_PROTOCOL_RTMP;
    else if(len == 5 && strncasecmp(url, "rtmpt", 5)==0)
        *protocol = RTMP_PROTOCOL_RTMPT;
    else if(len == 5 && strncasecmp(url, "rtmps", 5)==0)
            *protocol = RTMP_PROTOCOL_RTMPS;
    else if(len == 5 && strncasecmp(url, "rtmpe", 5)==0)
            *protocol = RTMP_PROTOCOL_RTMPE;
    else if(len == 5 && strncasecmp(url, "rtmfp", 5)==0)
            *protocol = RTMP_PROTOCOL_RTMFP;
    else if(len == 6 && strncasecmp(url, "rtmpte", 6)==0)
            *protocol = RTMP_PROTOCOL_RTMPTE;
    else if(len == 6 && strncasecmp(url, "rtmpts", 6)==0)
            *protocol = RTMP_PROTOCOL_RTMPTS;
    else {
        RTMP_Log(RTMP_LOGWARNING, "Unknown protocol!\n");
        goto parsehost;
    }
    }

    RTMP_Log(RTMP_LOGDEBUG, "Parsed protocol: %d", *protocol);

parsehost:
    //获取主机名称
    //跳过“://”
```

```
    p+=3;

    /* 检查一下主机名 */
    if(*p==0) {
        RTMP_Log(RTMP_LOGWARNING, "No hostname in URL!");
        return FALSE;
    }
    //原型：char *strchr(const char *s,char c);
    //功能：查找字符串s中首次出现字符c的位置
    //说明：返回首次出现c的位置的指针，如果s中不存在c则返回NULL。
    end   = p + strlen(p);//指向结尾的指针
    col   = strchr(p, ':');//指向冒号（第一个）的指针
    ques  = strchr(p, '?');//指向问号（第一个）的指针
    slash = strchr(p, '/');//指向斜杠（第一个）的指针

    {
    int hostlen;
    if(slash)
        hostlen = slash - p;
    else
        hostlen = end - p;
    if(col && col -p < hostlen)
        hostlen = col - p;

    if(hostlen < 256) {
        host->av_val = p;
        host->av_len = hostlen;
        RTMP_Log(RTMP_LOGDEBUG, "Parsed host    : %.*s", hostlen, host->av_val);
    } else {
        RTMP_Log(RTMP_LOGWARNING, "Hostname exceeds 255 characters!");
    }

    p+=hostlen;
    }

    /* 获取端口号 */
    if(*p == ':') {
        unsigned int p2;
        p++;
        p2 = atoi(p);
        if(p2 > 65535) {
            RTMP_Log(RTMP_LOGWARNING, "Invalid port number!");
        } else {
            *port = p2;
        }
    }

    if(!slash) {
        RTMP_Log(RTMP_LOGWARNING, "No application or playpath in URL!");
        return TRUE;
    }
    p = slash+1;

    {
    /* 获取应用程序(application)
     *
     * rtmp://host[:port]/app[/appinstance][/...]
     * application = app[/appinstance]
     */

    char *slash2, *slash3 = NULL;//指向第二个斜杠，第三个斜杠的指针
    int applen, appnamelen;

    slash2 = strchr(p, '/');//指向第二个斜杠
    if(slash2)
        slash3 = strchr(slash2+1, '/');//指向第三个斜杠，注意slash2之所以+1是因为让其后移一位

    applen = end-p; /* ondemand, pass all parameters as app */
    appnamelen = applen; /* ondemand length */

    if(ques && strstr(p, "slist=")) { /* whatever it is, the '?' and slist= means we need to use everything as app and parse plapath
om slist= */
        appnamelen = ques-p;
    }
    else if(strncmp(p, "ondemand/", 9)==0) {
                /* app = ondemand/foobar, only pass app=ondemand */
                applen = 8;
                appnamelen = 8;
        }
    else { /* app!=ondemand, so app is app[/appinstance] */
        if(slash3)
            appnamelen = slash3-p;
        else if(slash2)
            appnamelen = slash2-p;

        applen = appnamelen;
    }

    app->av_val = p;
    app->av_len = applen;
    RTMP_Log(RTMP_LOGDEBUG, "Parsed app     : %.*s", applen, p);
```

```
161.
162.     p += appnamelen;
163.     }
164.
165.     if (*p == '/')
166.         p++;
167.
168.     if (end-p) {
169.         AVal av = {p, end-p};
170.         RTMP_ParsePlaypath(&av, playpath);
171.     }
172.
173.     return TRUE;
174. }
175.
176. /*
177.  * 从URL中获取播放路径（playpath）。播放路径是URL中“rtmp://host:port/app/”后面的部分
178.  *
179.  * 获取FMS能够识别的播放路径
180.  * mp4 流: 前面添加 "mp4:", 删除扩展名
181.  * mp3 流: 前面添加 "mp3:", 删除扩展名
182.  * flv 流: 删除扩展名
183.  */
184. void RTMP_ParsePlaypath(AVal *in, AVal *out) {
185.     int addMP4 = 0;
186.     int addMP3 = 0;
187.     int subExt = 0;
188.     const char *playpath = in->av_val;
189.     const char *temp, *q, *ext = NULL;
190.     const char *ppstart = playpath;
191.     char *streamname, *destptr, *p;
192.
193.     int pplen = in->av_len;
194.
195.     out->av_val = NULL;
196.     out->av_len = 0;
197.
198.     if ((*ppstart == '?') &&
199.         (temp=strstr(ppstart, "slist=")) != 0) {
200.         ppstart = temp+6;
201.         pplen = strlen(ppstart);
202.
203.         temp = strchr(ppstart, '&');
204.         if (temp) {
205.             pplen = temp-ppstart;
206.         }
207.     }
208.
209.     q = strchr(ppstart, '?');
210.     if (pplen >= 4) {
211.         if (q)
212.             ext = q-4;
213.         else
214.             ext = &ppstart[pplen-4];
215.         if ((strncmp(ext, ".f4v", 4) == 0) ||
216.             (strncmp(ext, ".mp4", 4) == 0)) {
217.             addMP4 = 1;
218.             subExt = 1;
219.         /* Only remove .flv from rtmp URL, not slist params */
220.         } else if ((ppstart == playpath) &&
221.             (strncmp(ext, ".flv", 4) == 0)) {
222.             subExt = 1;
223.         } else if (strncmp(ext, ".mp3", 4) == 0) {
224.             addMP3 = 1;
225.             subExt = 1;
226.         }
227.     }
228.
229.     streamname = (char *)malloc((pplen+4+1)*sizeof(char));
230.     if (!streamname)
231.         return;
232.
233.     destptr = streamname;
234.     if (addMP4) {
235.         if (strncmp(ppstart, "mp4:", 4)) {
236.             strcpy(destptr, "mp4:");
237.             destptr += 4;
238.         } else {
239.             subExt = 0;
240.         }
241.     } else if (addMP3) {
242.         if (strncmp(ppstart, "mp3:", 4)) {
243.             strcpy(destptr, "mp3:");
244.             destptr += 4;
245.         } else {
246.             subExt = 0;
247.         }
248.     }
249.
250.     for (p=(char *)ppstart; pplen >0;) {
251.         /* skip extension */
```

```
252.         if (subExt && p == ext) {
253.             p += 4;
254.             pplen -= 4;
255.             continue;
256.         }
257.         if (*p == '%') {
258.             unsigned int c;
259.             sscanf(p+1, "%02x", &c);
260.             *destptr++ = c;
261.             pplen -= 3;
262.             p += 3;
263.         } else {
264.             *destptr++ = *p++;
265.             pplen--;
266.         }
267.     }
268.     *destptr = '\0';
269.
270.     out->av_val = streamname;
271.     out->av_len = destptr - streamname;
272. }
```

rtmpdump源代码（Linux）：http://download.csdn.net/detail/leixiaohua1020/6376561

rtmpdump源代码（VC 2005 工程）：http://download.csdn.net/detail/leixiaohua1020/6563163

文章标签：RTMPDump 源代码 RTMP_ParseURL url 解析

个人分类：libRTMP

所属专栏：开源多媒体项目源代码分析