一： Pjsip方案简要：

网址： pjsip

此文档的目的是对LocalService层或UI层的同事，对使用PJSIP方案中，RTP相关接口进行指导。 并对自己前面工作的一次总结。使工作思路更加清晰,指导自己今后的工作。

主要是介绍: 1:RTP收发包方案的接口。 2：RTP加解密的函数接口 。 3; RTP转发接口。 4;记录调试过程中的疑难问题。

后面将继续支持和维护的平台有 1： iOS64； 2： Windows 64； 3： Android； 4： Linux Qualcomm 32；

PJSIP方案中有比较多的模块，RTP收发包使用的是PJMEDIA （Open Source Media Stack[开源的媒体栈]）模块； 功能是实现对语音PCM 数据的采集，编码，压缩，打包；并且对RTP包的解压，使用播放等可逆的工作。

pjsip库没有裁剪的大小为：

iOS版本：

.a 大小为3M。

头文件为3M.

Windows版本：

.lib 大小为16M，

头文件为3M.

二： 收发包的函数接口说明

结构体：

```
struct info_rtp
{
    char *codec_id;      /* default codec */
    unsigned int local_port;     /* default local port to receive the rtp */
    char *remote_addr;           /* default remote addr to send */
    unsigned int remote_port;    /* default remote port to send the rtp here */
};
```

收发RTP的接口说明：

```
int main_rtp_send_rece( struct info_rtp *info_rtp_temp);
```

输入： 结构体具体赋值。
输出： 无
说明： 此函数是具体的线程，有for循环。 当一直收发包时，一直运行； 当不收发包时，退出for循环即可。

调用接口参考实例：

```
void call_like_this()
{
    struct info_rtp info_rtp_temp;
    memset(&info_rtp_temp,0,sizeof(info_rtp_temp));

    info_rtp_temp.codec_id="pcma";
    info_rtp_temp.local_port = 4001;
    info_rtp_temp.remote_addr = "172.26.49.2";
    info_rtp_temp.remote_port = 4003;

    /*Use the function to Send and receive the RTPS*/
    main_rtp_send_rece(&info_rtp_temp);
    }
```

三： 加解密的接口说明：

结构体：

结构体：

```
typedef struct META_DATA_INFO_S
{
        int len;
        void *addr;
}META_DATA_INFO_TT;
len    :  数据的长度；
addr :  数据在内存中的首地址；


typedef void (*PTR_ENCRYPT)(META_DATA_INFO_TT *p1,META_DATA_INFO_TT *p2);
回调函数定义
```

加解密的接口：

```
void CRYPTO_AES_encrypt_init(void)

功能：  对AES 进行初始化
输入:无
输出：无


void CRYPTO_AES_encrypt_rtp(META_DATA_INFO_T *text_t, META_DATA_INFO_T *cipher_t)

功能：对于传入的数据进行加密
输入:要加密的数据信息（待加密的长度，数据内存地址）。
输出：密文的数据信息（密文长度，密文数据内存地址）。


void CRYPTO_AES_decrypt_rtp(META_DATA_INFO_T *cipher_t, META_DATA_INFO_T *text_t)
功能：对于传入的数据进行解密
输入:要解密的数据信息（待解密的长度，待解密数据内存地址）。
输出：原文的数据信息（原文长度，原文数据内存地址）。


void CRYPTO_AES_encrypt_decrypt_register(PTR_ENCRYPT encrypt_temp,PTR_ENCRYPT decrypt_temp);
功能：注册加密，解密函数到RTP收发包函数中。
输入:加解密函数的地址。
输出：无。
```

调用接口参考实例：

```
void CRYPTO_AES_encrypt_rtp(META_DATA_INFO_T *text_t, META_DATA_INFO_T *cipher_t)
{
    printfkey(text_t->addr,text_t->len);
    if (CRYPTO_AES_encrypt(text_t->addr,cipher_t->addr, text_t->len, &g_key_handle_t))
    {
        printf("CRYPTO_AES_encrypt Error\n");
    }
    cipher_t->len = text_t->len;
}

void CRYPTO_AES_decrypt_rtp(META_DATA_INFO_T *cipher_t, META_DATA_INFO_T *text_t)
{
    if (CRYPTO_AES_decrypt(cipher_t->addr, text_t->addr, cipher_t->len, &g_key_handle_t))
    {
        printf("CRYPTO_AES_encrypt Error\n");
    }
    text_t->len = cipher_t->len;
}


void CRYPTO_AES_encrypt_init(void)
{
    CRYPTO_AES_encrypt_decrypt_register((PTR_ENCRYPT)CRYPTO_AES_encrypt_rtp,(PTR_ENCRYPT)CRYPTO_AES_decrypt_rtp);
    g_key_handle_t.feedback_mode = AES_128_OFB;
    strcpy((char*)&g_key_handle_t.IV[0], AES_IV);

    if (CRYPTO_AES_init(&g_key_handle_t))
    {
        printf("CRYPTO_AES_init ERROR: %s\n", CRYPTO_err_string());
    }
}
```

四： RTP转发端口申请接口说明：

ICE文件VoipMgr.ice中结构体和函数接口：

```
struct RtpIn
{
    string callerNum;
    string calleeNum;
    string callerUuid;
    string calleeUuid;
};

说明:
callerNum  主叫号码。
calleeNum  被叫号码
callerUuid    主叫的UUID
calleeUuid   被叫的UUID


struct RtpOut
{
    string RtpServerIP;
    string RtpServerPort;
};
说明:
RtpServerIP   服务器地址
RtpServerPort  服务器转发PORT


/*
Get The info out the Rtp info

*/
void GetRtpInfo(Environment env, RtpIn RtpInListTemp,
                out RtpOut RtpOutListTemp)
                throws  ServerException;
输入:  终端结构体赋值信息。
输出：服务器转发信息（IP和端口）。
```

五： 遗留问题

1： iOS 通话声音相对较小，

进展：但是通过Windows平台 通话声音验证，相同的代码和配置情况下，不会有这样的问题。 后面继续跟进。

六： git地址：

pjsip的git地址：

https://github.com/championlovecode/pjsip.git

all_platform的git地址：

https://github.com/championlu/rtp_for_all_platform.git