## 由网络节点处截获数据包重组sip-rtp文件

王宗冰(U201317457, [wangbing9269@gmail.com](mailto:wangbing9269@gmail.com), tel 13125145616, 201302班) 签字：



**摘要：**随着信息安全上升至国家战略高度，以及互联网信息技术的蓬勃发展，信息安全行业受到前所未有的重视，迎来一个历史性的发展契机。本次项目实验的主要目的是通过使用wireshark抓包软件抓取数据传输的会话，然后通过分析pcap包的文件内容，设计算法进行tcp和udp会话重组，输出tcp和udp结构体信息，然后提取tcp和udp负载的内容，恢复传输会话的文件。然后，本次项目实验还分析讨论了sip-rtp协议，搭建sip网络电话服务器和终端设备，然后使用wireshark工具抓取传输过程中的会话，分析其文件内容和格式，输出sip-rtp结构体信息，并成功恢复其传输的音频和视频内容。

**关键：**TCP协议，UDP协议，sip-rtp协议，数据包，文件，重组

1. 引言

SIP(Session Initiation Protocol)是一种工作在应用层的协议，采用客户端/服务器结构的消息机制，呼叫建立的过程简单。SIP类似于 HTTP，是基于文本的协议[1]。随着下一代网络的发展，使用SIP作为VoIP网络的核心信令控制协议，已经成为了未来的发展趋势。SIP拥有极高的易用性和灵活性，却在全性方面存在缺陷。本文分析了SIP协议，成功提取其传输的文件。

* 1. 项目研究背景

由 IETF 最年轻的工作组之一的 SIP(session initiation protocol)工作组发布的 SIP 相关标准已经成为通信和网络界的研究热点。目前SIP工作组发表的协议和草案超过30个,除了核心协议以外,其他协议涵盖QoS、安全、消息头和方法扩展、与 PSTN 等其他协议的互操作性、穿透防火墙和 NAT、应用、多消息体、即时消息等诸多话题,其最终目标是为所有Internet成功的应用提供无所不在的接入,成为联系 Internet 和电信、多媒体的媒介。

3GPP(3rd generation partnership project,第三代移动通信伙伴项目)工作组已经采用 SIP 作为其IP多媒体域的工作协议,其目标是对 Internet 所拥有的所有成功服务提供无处不在的接入,将Interne世界和蜂窝世界融合在一起。

* 1. 项目研究目的和意义

本项目的研究目的：

⑴．随着互联网技术的不断发展和广泛应用，计算机网络在现代生活的作用越来越重要，如今，个人、企业以及政府部门，国家军事部门，不管是天文的还是地理的都依靠网络传递信息，这已成为主流，人们也越来越依赖网络。然而，网络的开放性与共享性容易使它受到外界的攻击与破坏。信息安全技术致力于解决如何有效进行介入控制，以及如何保证数据传输安全性，如何保护个人财产和信息安全，在网络技术高速发展的今天，信息安全的研究就显得尤为重要；

⑵．通过本项目的研究，进而提高个人信息安全基本素养，对信息安全目前的研究领域和当前遇到的问题有一个基本了解，培养了个人对信息安全领域的浓厚兴趣，提高了自己的知识面；

⑶．通过对TCP, UDP,SIP,RTP,HTTP,FTP协议的研究，分析这些协议的安全问题，并试图提出一些信息安全的防御措施。

研究意义：

信息安全问题已成为社会关注的焦点。一方面，信息技术已经称为整个社会经济和企业生存发展的重要基础，在国计民生和企业经营中的重要性日益凸现；另一方面，政府主管机构、企业和用户对信息技术的安全性、稳定性、可维护性和可发展性提出了越来越迫切的要求，因此，从社会发展和国家角度来看，加大发展信息安全技术的力度已经刻不容缓。

* 1. 项目主要研究内容

⑴. 练习使用wireshark抓包软件,掌握wireshark抓包工具的使用方法，了解其基本原理；

⑵．学习计算机网络体系结构的有关架构，例如OSI和TCP/IP模型；

⑶．分析TCP和UDP传输的协议内容，利用wireshark抓包软件分析TCP和UDP协议的通信过程，对TCP和UDP按照目的IP、源IP、目的端口、源端口进行分包，并输出其结构体等信息，提取传输过程中的文件和网络信息；

⑷．分析SIP和RTP协议，利用wireshark抓包软件分析SIP协议的通信过程学习SIP协议的工作流程，成功恢复其传输的视频和语言信息。

1. 项目问题分析与解决问题的算法

根据对问题的分析和讨论，本项目的需求主要有从从网络节点中获取pcap文件，从pcap文件中析取所有TCP会话与UDP会话，从pcap文件中分析提取所有SIP-RTP会话：



图2.1 项目流程图

* 1. wireshake抓包

**⑴. Pcap文件结构说明**

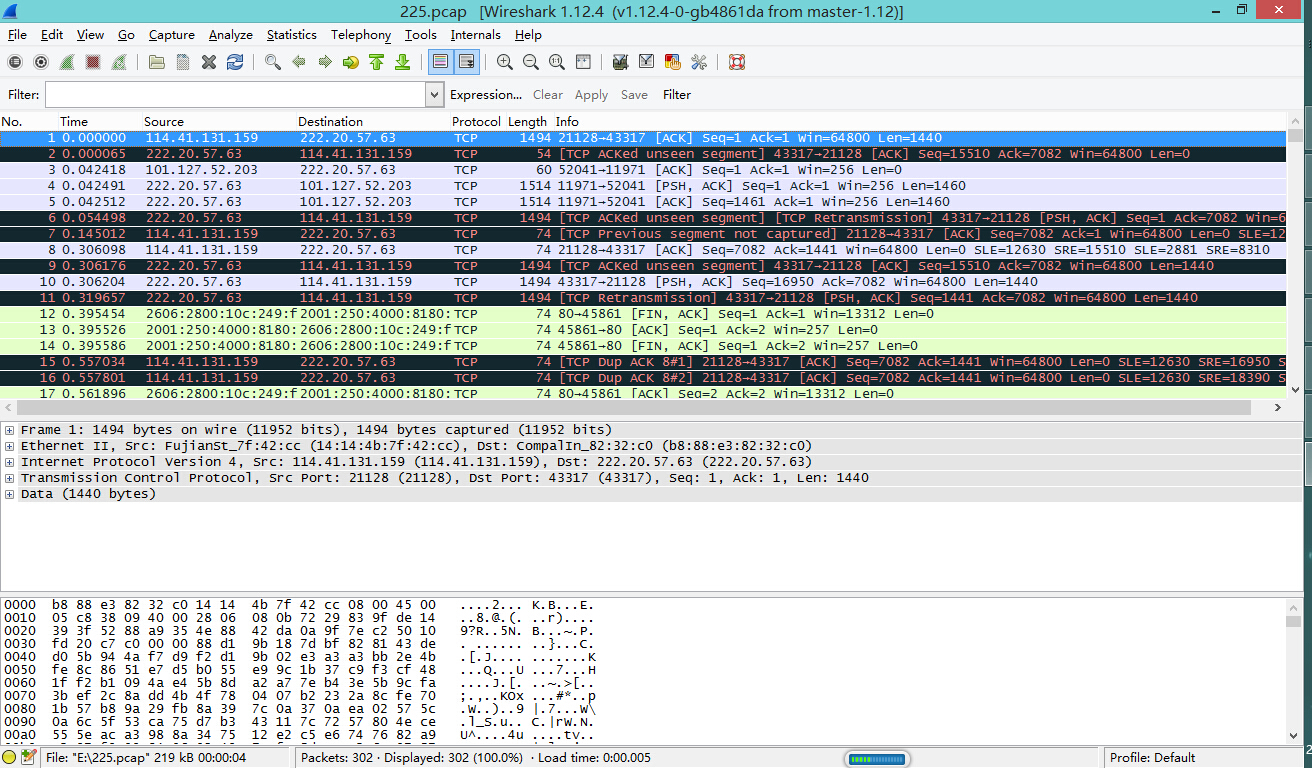
wireshake主界面如下：  
[](https://raw.githubusercontent.com/hujiaxuan1995/MyImage/master/wireshake.jpg)

图2.2 wireshark主界面

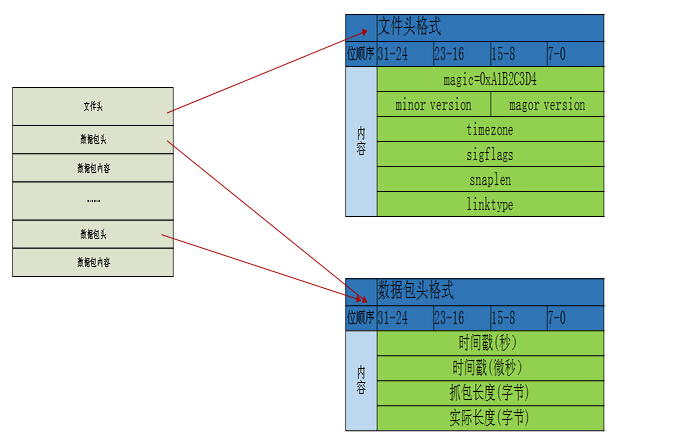
其中pcap 文件头占24个字节且唯一，即一个pcap文件中只有一个pcap文件头。pcap数据包头占16字节，其后面紧跟这数据包。数据包中又分为三层，即以太网数据帧(占据14个字节)、IP包头(一般为20字节)、TCP/UDP。  


图2.3 pcap文件结构

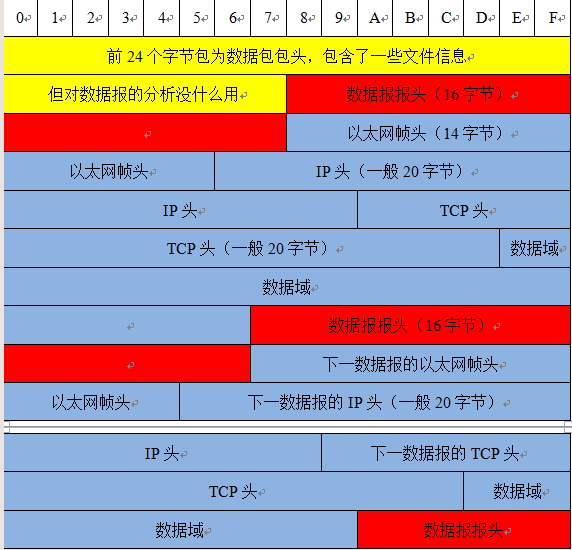


图2.3 pcap文件详细结构图

**⑵. TCP 包头**

根据上方的结构图，可以知道 TCP 的包头结构，如下：

Source Port：源端口，16位。 Destination Port：目的端口，16位。 Sequence

Number：发送数据包中的第一个字节的序列号，32

Acknowledgment Number：确认序列号，32位。 Data

Offset：数据偏移，4位，该字段的值是TCP首部（包括选项）长度除以4。

Flag：标志位，标识TCP不同的控制消息，8 位

Window：接收缓冲区的空闲空间，16位，用来告诉TCP连接对端自己能够接收的最大数据长度。

Checksum：校验和，16位。

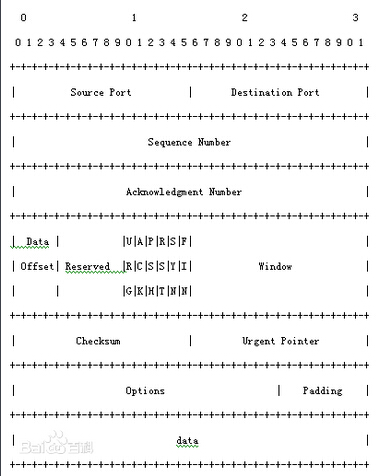
Urgent Pointers：紧急指针，16位，只有URG标志位被设置时该字段才有意义，表示紧急数据相对序列号（Sequence Number字段的值）的偏移   


图2.4 TCP详细结构图

**⑶. UDP 包头**

UDP报头由4个域组成，其中每个域各占用2个字节，具体如下：

源端口号

目标端口号

数据报长度：指包括报头和数据部分在内的总字节数

校验值

* 

图2.5 UDP详细结构图

以上是pcap数据包中头部的几层协议，wireshake抓取的pcap文件中有很多个基本的pcap包

* 1. tcp/udp会话重组

pcap包根据各自不同的源ip、目的ip、源端口、目的端口、协议类型加以区分，我们要做的第一项工作就是把属于同一五元组的pcap包筛选出来。筛选的时候并不区分目的ip和源ip。

**⑴．各种类型报文对象的构造：**

由于pcap每次抓到一个数据包，会提交其信息，方式为以unsigned char\* 指向的一段缓冲区，现将缓冲区前14个字节读入，按以太帧格式构造以太帧头部的对象。然后根据以太帧头部中的Type字段，决定接下来应该构造ip还是arp还是rarp。假设是ip，那么把缓冲区中第15个字节开始直到这块缓冲区最后的所有字节读入，按ip报文格式构造ip的对象，根据ip的protocol字段，决定接下来构造tcp，udp。假设是tcp，则将ip的数据内容读入，按tcp格式构造tcp的对象。



图2.6 报文对象构造图

**⑵．TCP/UDP重组：**

选中一个TCP/UDP报文，然后开始重组。先对选中的报文生成IP对象和TCP/UDP对象，然后对所有收到的报文依次进行检查，因为一般TCP传输时，每个文件的端口是不同的，所以发现源目的地址和源目的端口和选中那个均相同的，而且其数据长度>0，则认为是同一个文件，利用map，把五元组作为映射为其生成CFounder对象。



图2.7 TCP会话构造图

* 1. sip会话重组

**⑴判断是否为sip协议：**

当捕获到IP数据包后，先去除IP头、TCP/UDP头，然后去除若干字节进行比较，如果符合SIP协议的特征，则可判断该数据包为SIP数据包，具体比较分如下两个部分:

对于SIP请求消息，以method sip开始，其中method可以是Invite、Ack、Bye、Cancel、Opt1n等，本系统的实现只涉及到SIP请求消息中的Invite、Ack、Bye、Cancel,因此只需比较开头若干字节中是否为Invite sip、Ack sip、Bye sip、Cancel sip四者中的一个即可判断该数据包是不是SIP请求数据包。

对于SIP应答，处理方法相似，但是SIP应答消息的开头若干字节是SIP/2.0xxx,其中XXX可以是1XX-6XX中任意一个应答代码。因此对于SIP应答，只需在去除IP头、TCP/UDP头后判断接下来的几个字节是否是SIP/2.0即可。

**⑵．rtp Header解析**

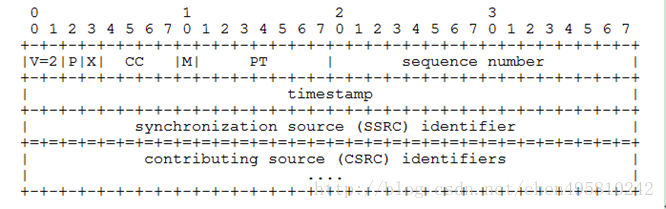


图2.8 rtp Header

V：RTP协议的版本号，占2位，当前协议版本号为2

P：填充标志，占1位，如果P=1，则在该报文的尾部填充一个或多个额外的八位组，它们不是有效载荷的一部分。

X：扩展标志，占1位，如果X=1，则在RTP报头后跟有一个扩展报头

CC：CSRC计数器，占4位，指示CSRC 标识符的个数

 M: 标记，占1位，不同的有效载荷有不同的含义，对于视频，标记一帧的结束；对于音频，标记会话的开始。

PT: 有效荷载类型，占7位，用于说明RTP报文中有效载荷的类型，如GSM音频、JPEM图像等,在流媒体中大部分是用来区分音频流和视频流的，这样便于客户端进行解析。

序列号：占16位，用于标识发送者所发送的RTP报文的序列号，每发送一个报文，序列号增1。这个字段当下层的承载协议用UDP的时候，网络状况不好的时候可以用来检查丢包。同时出现网络抖动的情况可以用来对数据进行重新排序，序列号的初始值是随机的，同时音频包和视频包的sequence是分别记数的。

时戳(Timestamp)：占32位，必须使用90 kHz 时钟频率。时戳反映了该RTP报文的第一个八位组的采样时刻。接收者使用时戳来计算延迟和延迟抖动，并进行同步控制。

同步信源(SSRC)标识符：占32位，用于标识同步信源。该标识符是随机选择的，参加同一视频会议的两个同步信源不能有相同的SSRC。

特约信源(CSRC)标识符：每个CSRC标识符占32位，可以有0～15个。每个CSRC标识了包含在该RTP报文有效载荷中的所有特约信源。

**判断是否为RTP协议：**遇到UDP包，则找到srcIP,srcPort,dstIP,dstPort, 根据这四个信息判断是否是RTP包，找到其callid信息

**⑶．Sip会话重组算法**

1. 建立map<callid,order<sipHeader>>, map<order<sipHeader>,callid>, map<callid, vector<rtp>>；
2. 判断是否是sip包；
3. Sip-invite，则找到srcIP,srcPort,callid（均指的负载信息中的传输IP和Port)存在于map<callid,order<sipHeader>>；
4. Sip-invite-ok，则找到dstIP,dstPort,callid, 根据map<callid,order<sipHeader>> 找到sipHeader，完善其信息（作用仅限于此)，并插入到map<order<sipHeader,callid>；
5. 遇到UDP包，则找到srcIP,srcPort,dstIP,dstPort, 根据这四个信息判断是否是RTP包，找到其callid信息，然后插入map<callid,vector<rtp>>；
6. 遇到sip-bye，设置sip中的bye为true;；
7. 遇到sip-bye-ok，设置sip中的bye-ok为true，输出payload并进行解码，解码后删除map<callid,order<sipHeader>>,map<order<sipHeader,callid>和map<callid,vector<rtp>>。

图2.9 sip提取算法过程

1. 项目软件结构与详细程序文字描述

本项目主要设计了三个类，多个结构体：

//五元组设计

struct five

{

u\_int8 protcol;

u\_int32 srcIP;

u\_int32 dstIP;

u\_int16 srcPort;

u\_int16 dstPort;

bool operator<(const five& \_\_x) const

{

return (srcIP + dstIP + srcPort + dstPort) <

(\_\_x.srcIP + \_\_x.dstIP + \_\_x.srcPort+\_\_x.dstPort);

}

};

//sip结构体信息

struct sipHeader

{

u\_int32 srcIP; //sip会话的源IP地址 并不是指中间服务器

u\_int32 dstIP; //sip会话的目的IP地址 并不是指中间服务器

u\_int16 srcMediaPort; //RTP传输的源端口

u\_int16 dstMediaPort; //RTP传输的目的端口

bool INVITE\_OK; //是否收到INVITE的OK反馈

bool BYE; //是否发送结束信息

bool BYE\_OK; //是否收到结束的反馈

bool operator<(const sipHeader& \_\_x) const

{

return (srcIP + dstIP + srcMediaPort + dstMediaPort) <

(\_\_x.srcIP + \_\_x.dstIP + \_\_x.srcMediaPort+\_\_x.dstMediaPort);

}

};

//RTP文件头结构体

struct rtpHeader

{

/\*

版本号（V）：2比特，用来标志使用的RTP版本。

填充位（P）：1比特，如果该位置位为1，则该RTP包的尾部就包含附加的填充字节。

扩展位（X）：1比特，如果该位置位为1，则RTP固定头部后面就跟有一个扩展头部。

CSRC计数器（CC）：4比特，含有固定头部后面跟着的CSRC的数目。

\*/

u\_int8 stream;

u\_int8 pt; //负载类型

u\_int16 sn; //序列号，按照这个排序

u\_int32 timeStamp; //时间戳

u\_int32 ssrc; //同步源标识符

//u\_int8 payLoad[0]; //负载信息

u\_int8 \*payLoad; //负载信息

size\_t payLoadLen; //长度

bool direction; //标识方向, 0, 小的到大的， 1， 大的到小的

//根据sn排序

bool operator<(const rtpHeader& \_\_x) const

{

if (direction < \_\_x.direction) return true;

if (direction > \_\_x.direction) return false;

return (sn < \_\_x.sn);

}

};

//au头 https://en.wikipedia.org/wiki/Au\_file\_format

struct auHeader

{

u\_int32 magicNum; //the value 0x2e736e64 (four ASCII characters ".snd")

u\_int32 dataOffset; //the offset to the data in bytes, must be divisible by 8.

u\_int32 dataSize; //data size in bytes. If unknown, the value 0xffffffff should be used.

u\_int32 encoding; //Data encoding format:

u\_int32 sampleRate; //the number of samples/second, e.g., 8000

u\_int32 channel; //the number of interleaved channels, e.g., 1 for mono, 2 for stereo; more channels possible, but may not be supported by all readers.

};



图3.1 MFC类图 图3.2 PCAP类图

CpcapApplicationDlg 类继承自 CDialogEx 主要的目的是UI设计，sip和pcap属性是pcap分包和sip分包的类

PCAP类逻辑中最重要的一个类图。



图3.2 SIP类图

SIP类图处理SIP会话逻辑

1. 项目软件安装

直接运行即可，无需安装

测试环境:

OS：Windows 10专业版

处理器：Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz 2.60GHz

RAM：4.00GB

系统类型：64位操作系统

1. 项目软件运行截屏与说明

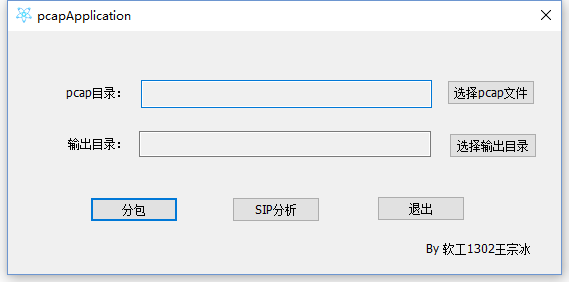


图5.1 开始界面

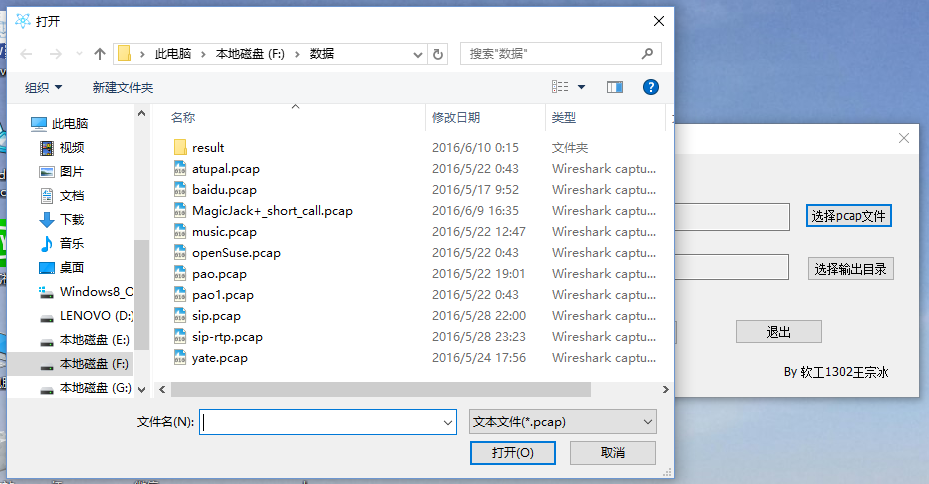


图5.2 点击选择pcap文件

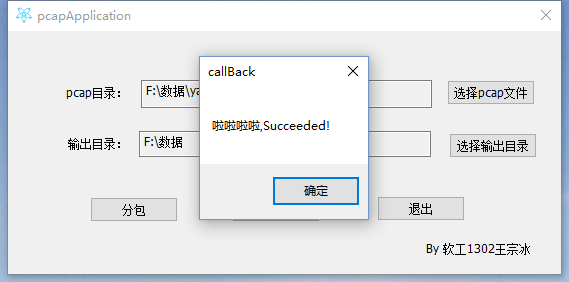


图5.3 选择pcap文件和输出目录后，点击分包按钮界面

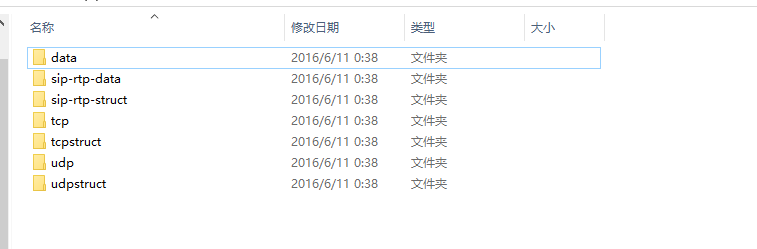


图5.4 输出文件目录结构

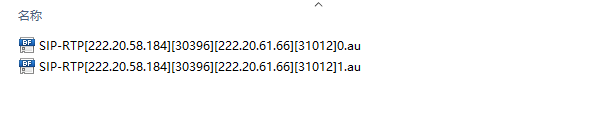


图5.5 输出SIP-RTP文件



图5.6 输出SIP-RTP结构体目录

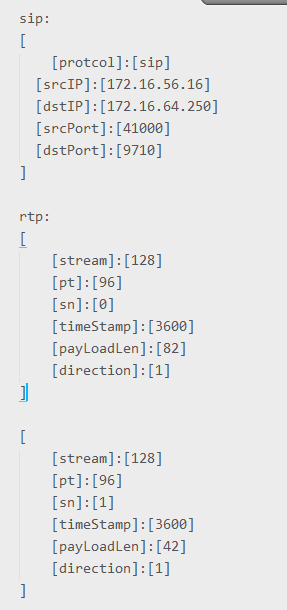


图5.6 输出SIP-RTP结构体内容

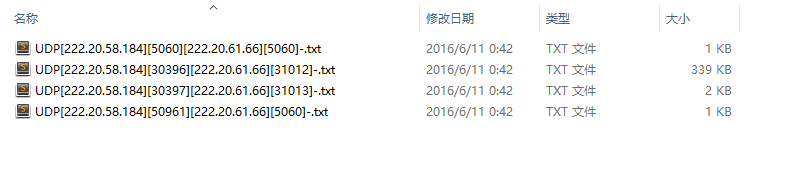


图5.7 输出UDP结构体目录

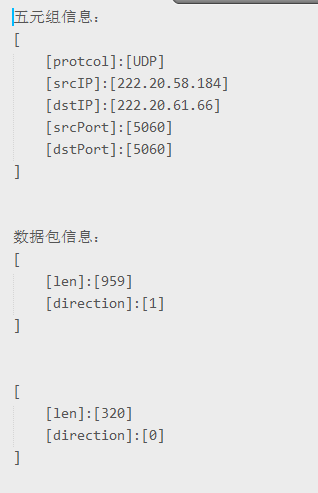


图5.8 输出UDP结构体内容

1. 测试与分析对比
   1. **测试SIP语音**

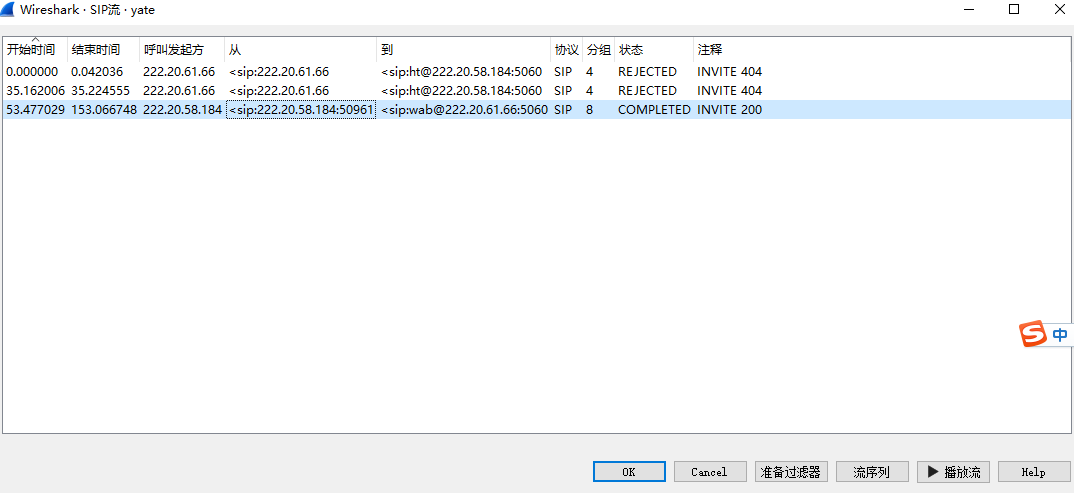


图6.1 Wireshark截取的SIP通话

通过WireShark软件分析得到有一个完整的双向通话

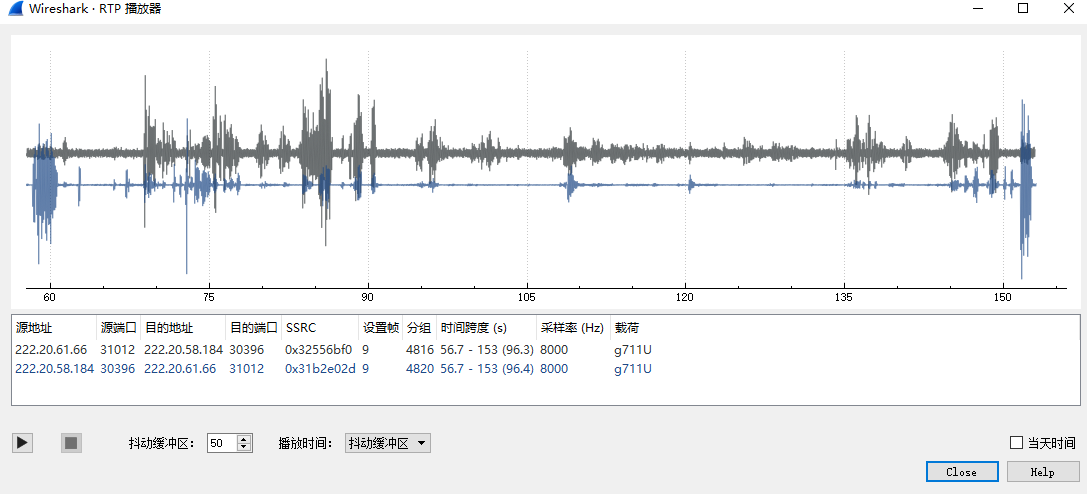


图6.2 WireShark RTP播放器

使用WireShark语音播放器播放音频

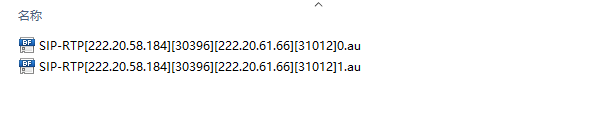


图6.3. 输出SIP-RTP文件

运行项目程序得到两条完整音频，试听与WireShark语言播放器的音频完全一致，且没有任何杂音。

* 1. **测试SIP视频**



图6.4 程序输出视频文件1



图6.5 程序输出视频文件2

经过测试，老师给的标准测试包，有4个视频文件，且均能正常播放，无损坏。

1. 结束语（总结 项目收获）

通过这个程序，加深了对网络报文的认识和理解。尤其是在重组时，以SIP-RTP电话为实验对象，对其工作方式有了一些粗略的了解。SIP网络电话客户端主动和服务器端的控制端口（一般默认为21）建立控制连接，进行会话。当有文件进行传输时，控制连接利用临时端口建立一条数据连接，因为临时端口数量比较大，一般多个文件传输，一个文件分配一条数据连接，每个连接的端口不同。

一开始重组\*.txt格式文件成功后，增加成就感，更推动自己重组其他格式文件。在重组大文件时发生内存不够问题，经过不断研究，反复尝试，优化算法，终于使程序顺利运行。即使是一般\*.rm文件，也可以成功重组并保存，再进行观看。

同时对MFC的机制有了进一步了解。在编写这个程序时，用到了线程，消息传递，窗口划分，文档串行化，时钟事件，自定义状态栏等等。经过查看相应书籍，并在网上找到一些可参考的文档，通过不断尝试，终于成功。

提高了自己的计算机科学素养，对信息安全目前的研究领域和当前遇到的问题有一个基本了解，培养了个人对信息安全领域的浓厚兴趣，提高了自己的知识面。对TCP, UDP,SIP,RTP,HTTP,FTP协议的研究，加深了对计算机网络知识的了解，使自己在平时开发过程中更加注重软件项目的安全性。

最后，感谢覃教授在本次项目中的指导和帮助，覃老师的授课风格和以前授课老师的风格完全不同，我感觉在信息安全这门课上学到了很多知识，受益匪浅，谢谢！

参考文献

[1] IETF.RFC3261.SIP:Session Initiation Protocol.2002

[2] IETF.RFC 3551 - RTP Profile for Audio and Video Conferences with Minimal Control

[3] 白芸洁，穆维新，刘润杰，申金媛.SIP协议分析及其安全性研究[J].现代电子技术，2009，40（6）：72-74.

[4] 白建军，彭晖.SIP 揭秘[M].北京：人民邮电出版社，2003.

[5] Dierks T,Allen C.TLS Protocol Version.RFC 2246.1999**.**