

SHANGHAI JIAO TONG UNIVERSITY

**计算机通信网网络嗅探器报告**



报告题目： 计算机通信网网络嗅探器报告

学生姓名学号: 陈卓

学生姓名学号: 张侨伦 5140809064

专 业: 信息安全

指导教师: 蒋兴浩

学院(系): 电子信息与电气工程学院

目录

[**第一章 概述** 2](#_Toc502487669)

[1.1 运行环境 2](#_Toc502487670)

[1.2 编译工具 2](#_Toc502487671)

[2.1 程序主界面 2](#_Toc502487672)

[2.1.1 三级标题 2](#_Toc502487673)

[2.2 TCP、UDP数据包的全部存储显示 3](#_Toc502487674)

[**2.2.1 IP重组** 3](#_Toc502487675)

[遍历所有包 3](#_Toc502487676)

[**2.2.2 界面显示** 3](#_Toc502487677)

[2.3 包过滤 3](#_Toc502487678)

[2.4 数据包查询 3](#_Toc502487679)

[**2.4.1 数据包查询部分** 3](#_Toc502487680)

[**2.4.2 查询包显示部分** 3](#_Toc502487681)

[2.5 数据包保存 3](#_Toc502487682)

[2.1 程序主界面 4](#_Toc502487683)

[2.1.1 三级标题 4](#_Toc502487684)

[2.1 图形界面 4](#_Toc502487685)

[2.1.1 三级标题 4](#_Toc502487686)

[2.2 ip重组 4](#_Toc502487687)

[2.3 数据包查询 4](#_Toc502487694)

[第四章 **程序测试截图及说明** 4](#_Toc502487695)

[4.1 软件开始界面 4](#_Toc502487696)

[4.2 基本功能 5](#_Toc502487697)

[4.3 TCP/UDP数据包的全部数据展示（IP分片重组） 7](#_Toc502487698)

[8](#_Toc502487699)

[4.4 包过滤 8](#_Toc502487700)

[10](#_Toc502487701)

[10](#_Toc502487702)

[4.5 数据包查询 10](#_Toc502487703)

[4.6 数据包保存 11](#_Toc502487704)

[第五章 **遇到的问题及解决方法** 12](#_Toc502487705)

[5.1 张侨伦 13](#_Toc502487706)

[2.1 陈卓 14](#_Toc502487707)

[2.1.1 三级标题 14](#_Toc502487708)

[第六章 **体会与建议** 14](#_Toc502487709)

[6.1 张侨伦 14](#_Toc502487710)

[6.2 陈卓 14](#_Toc502487711)

[2.1.1 三级标 14](#_Toc502487712)

**计算机通信网网络嗅探器报告**

**第一章 概述**

1.1 运行环境

系统：ubuntu 17.04

Linux version 4.13.0-21-generic (buildd@lgw01-amd64-037) (gcc version 7.2.0 (Ubuntu 7.2.0-8ubuntu3))

1.2 编译工具

（1）g++

（2）Qt5.9.3

（3）Qt Creator 4.4.1

Based on Qt 5.9.3 (GCC 5.3.1 20160406 (Red Hat 5.3.1-6), 64 bit)

Built on Nov 21 2017 04:12:40

第二章 **主要算法**

2.1 程序主界面

正文内容

2.1.1 三级标题

正文内容

2.2 TCP、UDP数据包的全部存储显示

分为两部分，第一部分是IP重组，第二部分是显示重组之后的内容

**2.2.1 IP重组**

遍历所有包

（1）首先检查标识DF和MF字段，检查是否有分片。

（2）如果有分片，将包的源IP地址，目的IP地址，源端口，目的端口组合在一起，计算一个hash值，存储包和hash值。

（3）检查hash值是否已经存过了。

（4）如果hash值没有存过，那么新建一个vector数据类型，储存hash值以及这个包

（5）如果hash以及存过了，找到hash值对应的vector，将这个包存在对应的vector里面。

（6）处理完所有包，得到最终ip重组的结果。

**2.2.2 界面显示**

2.3 包过滤

读取用户输入的过滤表达式，将其编译，然后在抓包的时候讲这个过滤表达式结合如抓包的函数，在抓包的时候只抓取对应的包。

过滤格式为。

1. 指定源ip： ip src [ip address]，见图4-8
2. 指定目的ip： ip dst [ip address]，见图4-9
3. 指定源端口： src port [port address]，见图4-10
4. 指定目的端口： dst port [port address]，见图4-11

2.4 数据包查询

包查询分为查找包和显示部分。

**2.4.1 数据包查询部分**

首先新建一个vector,用于存储查找到的用户数据包。接收用户输入，然后遍历每一个包，查询包的数据，数据里包含用户输入的字符串，那么就将这个包存在vector里面。

**2.4.2 查询包显示部分**

2.5 数据包保存

（1）根据用户在页面上的选择，得到用户要保存的包的序号。

（2）遍历用户要保存的序号，打印对应的包到文件。

这中间有一个地方需要注意。由于我是用fprint来保存，文件指针有一定的发现限制。在遍历包的时候。遍历每一个包之后都要释放文件指针，让文件写入。遍历每一个包之前都要重新申请文件指针，而写入采取在最后面添加的方式。

2.1 程序主界面

正文内容

2.1.1 三级标题

正文内容

第三章 **主要数据结构**

2.1 图形界面

正文内容

2.1.1 三级标题

正文内容

2.2 ip重组

下面这个是用于存储属于同一个ip包的所有分组的，size\_t hash\_fragment用于存储ip属于同一个ip包的表示hash值，std::vector<int> ip\_vector\_fragment用于存储属于同一个ip报的分组的序号。

struct ip\_vector {

std::vector<int> ip\_vector\_fragment;

size\_t hash\_fragment;

bool flag;

};

而std::vector<unsigned char \*> Data\_after\_reasm;用于存储ip重组之后的所有包。

2.3 数据包查询

存储查找到的数据包的结构是：std::vector<unsigned char \*> Data\_Finded。

第四章 **程序测试截图及说明**

4.1 软件开始界面

运行之后，下图4-1即为软件初始界面。如图4-2选定要用的网卡之后，点击run即可开始抓包。

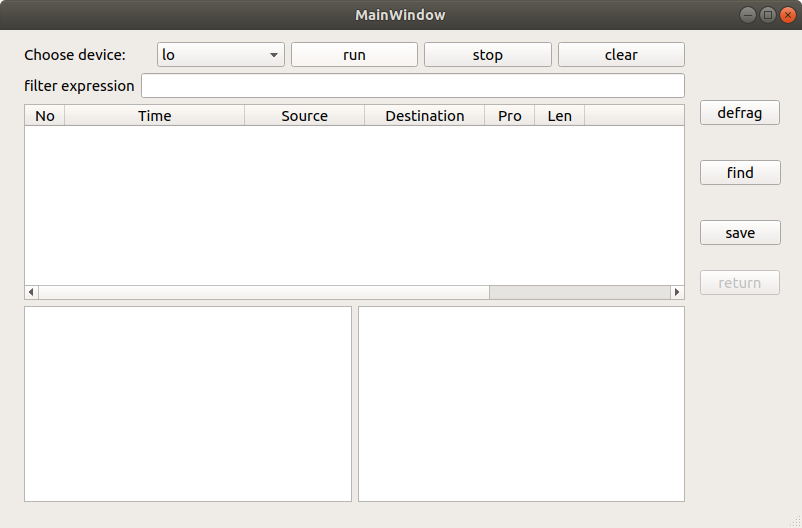


图 4-1 软件初始界面

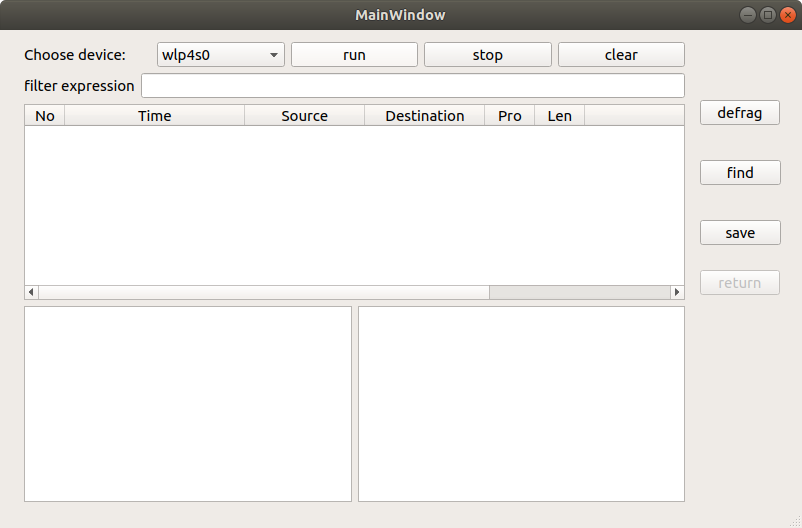


图4-2 选定网卡

4.2 基本功能

下图4-3，图4-4，图4-5，图4-6分别为ARP、ICMP、TCP、UDP等报文中各字段的内容

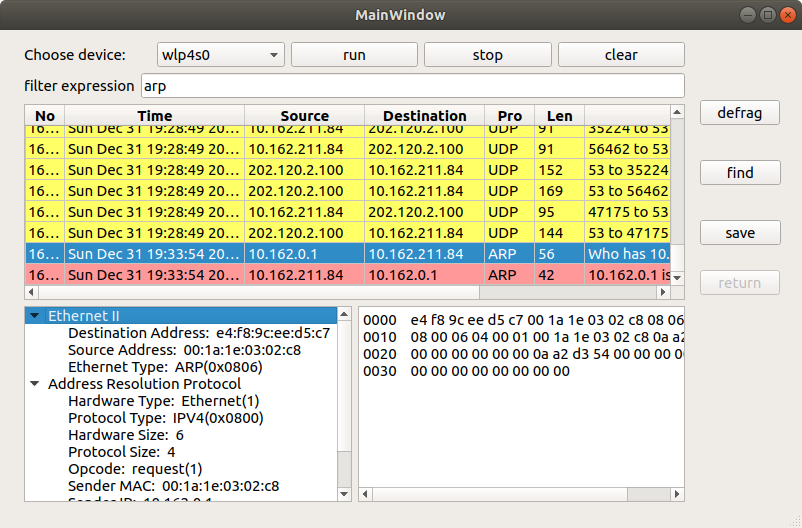


图4-3 ARP报文

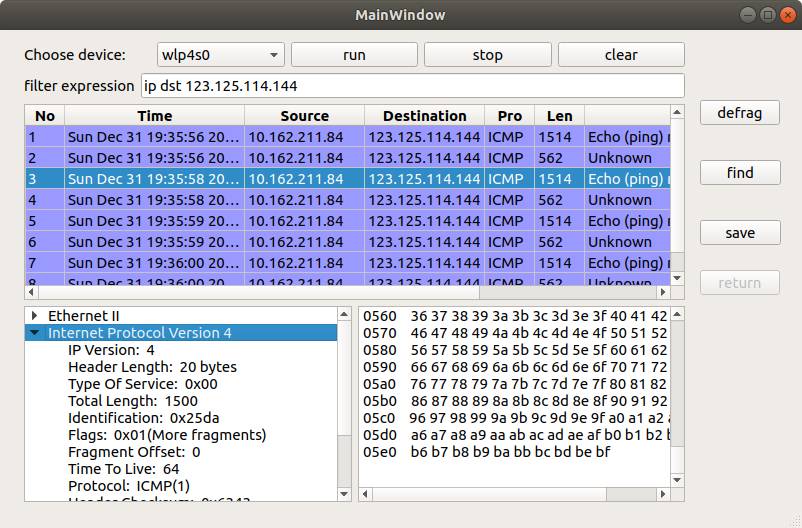


图4-4 ICMP报文

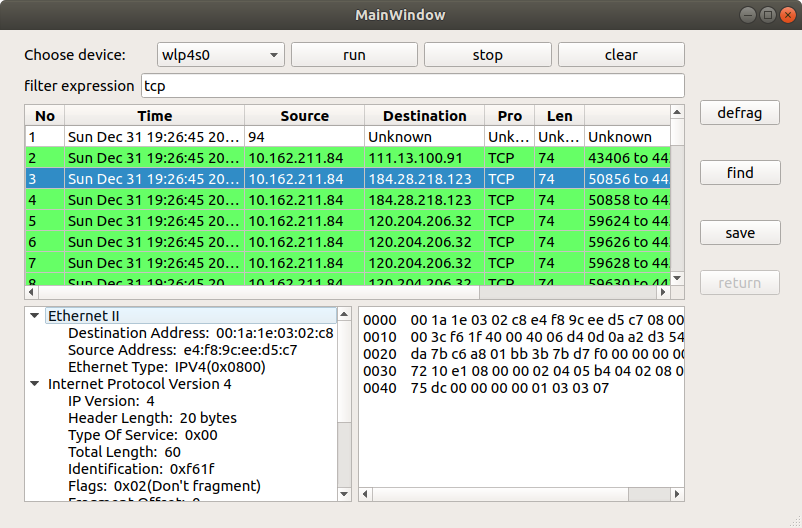


图4-5 TCP报文

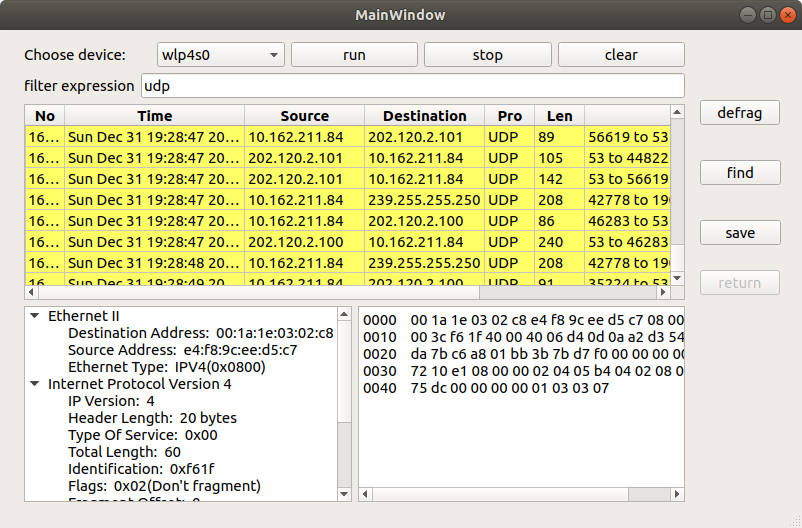


图4-6 UDP报文

4.3 TCP/UDP数据包的全部数据展示（IP分片重组）

点击defrag即可到分片重组界面，软件会显示出分片重组后的包。下图4-9为看到分片重组后的界面。点击return按钮即可回到原来的界面。

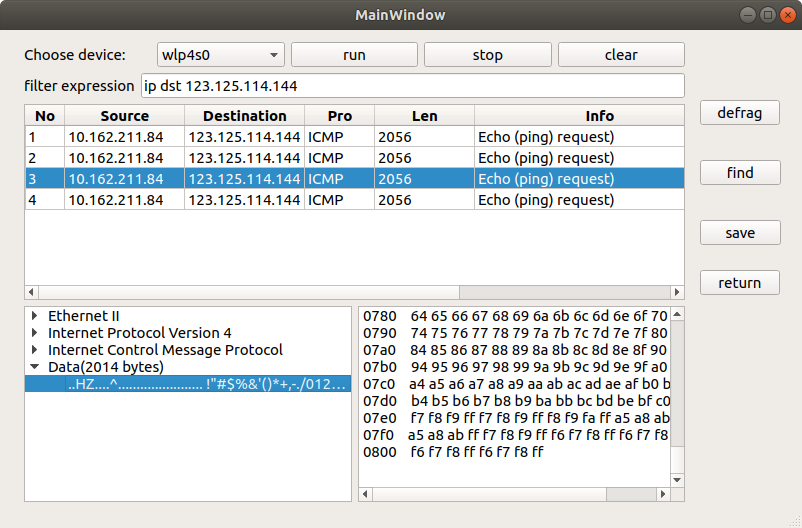


图4-7 产生分片包

4.4 包过滤

包过滤采用抓包时实现，抓包的时候可以指定过滤内容。过滤格式为。

1. 指定源ip： ip src [ip address]，见图4-8
2. 指定目的ip： ip dst [ip address]，见图4-9
3. 指定源端口： src port [port address]，见图4-10
4. 指定目的端口： dst port [port address]，见图4-11

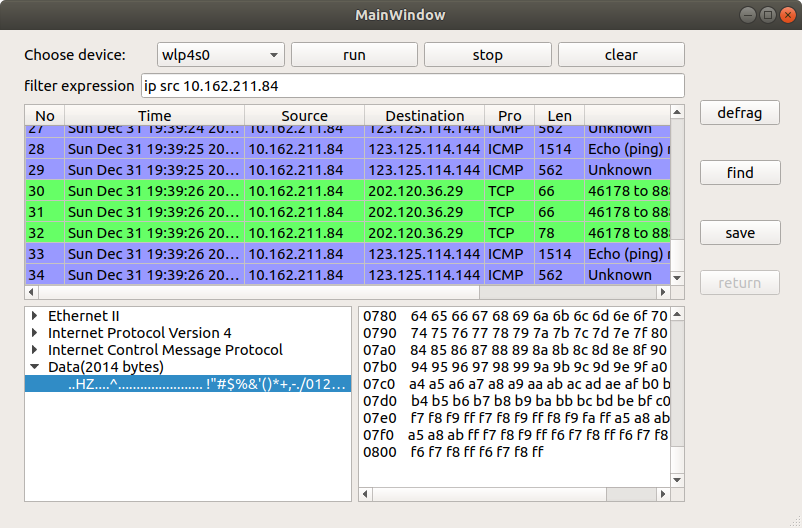


图4-8 指定源ip

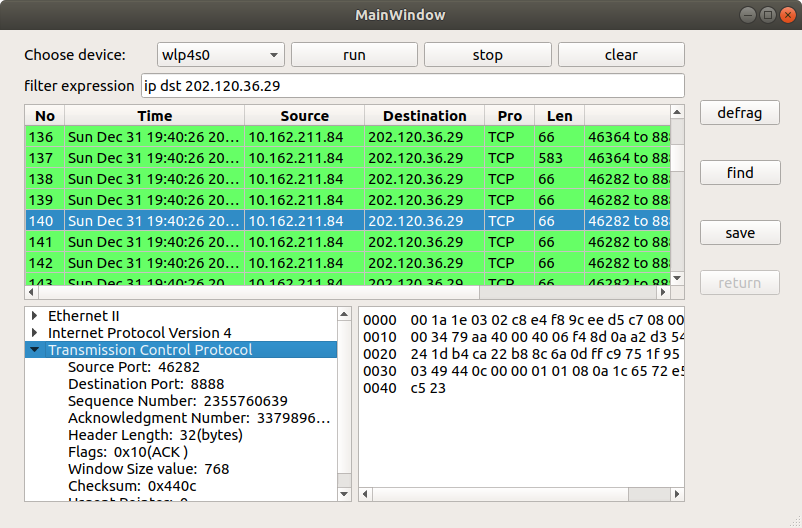


图4-9 指定目的ip

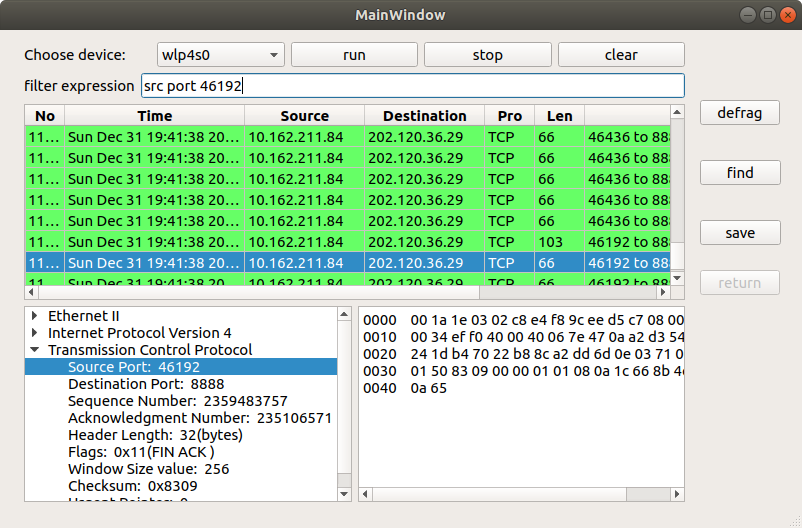


图4-10 指定源端口

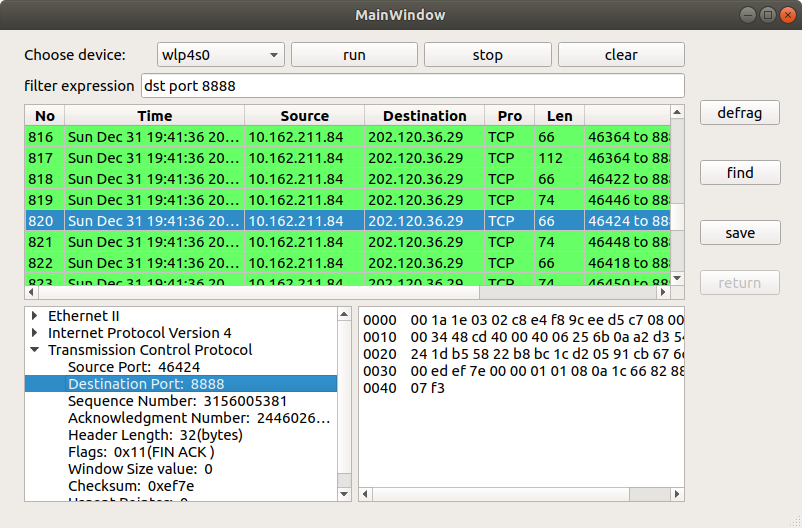


图4-11 指定目的端口

4.5 数据包查询

下图4-12为输入字符串窗口，图4-13为数据包查询结果。

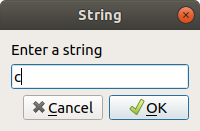


图4-12 输入字符串窗口

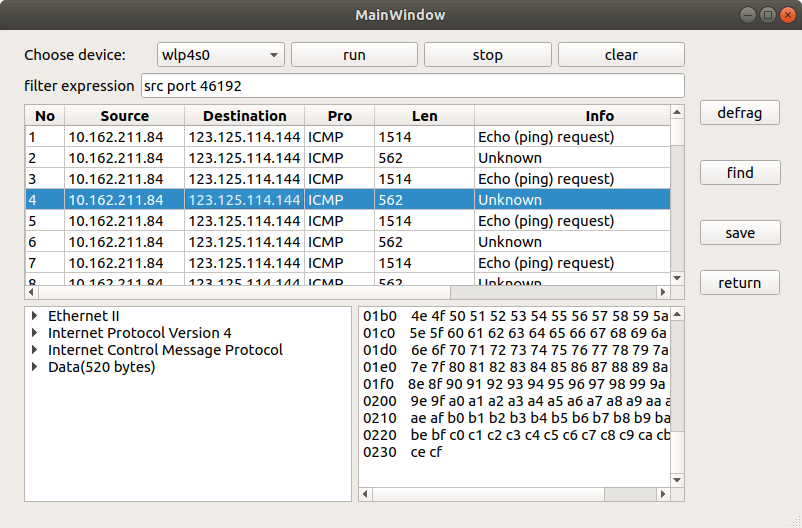


图4-13 找到的数据包

4.6 数据包保存

选中数据包之后，可点击保存，在图4-14中输入文件名，即可保存为可读文件。最后想要，文件的存储路径为可执行文件所在目录。下图4-15，图4-16为保存的数据包样例。

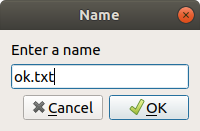


图4-14 存储文件名

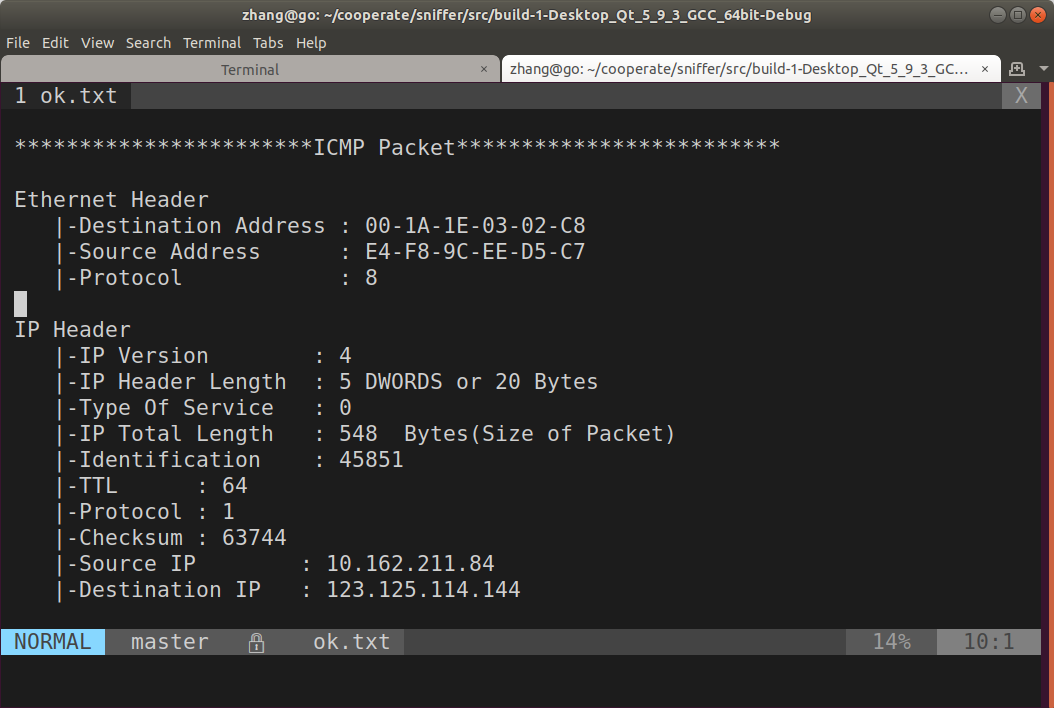


图4-15 存储文件样例

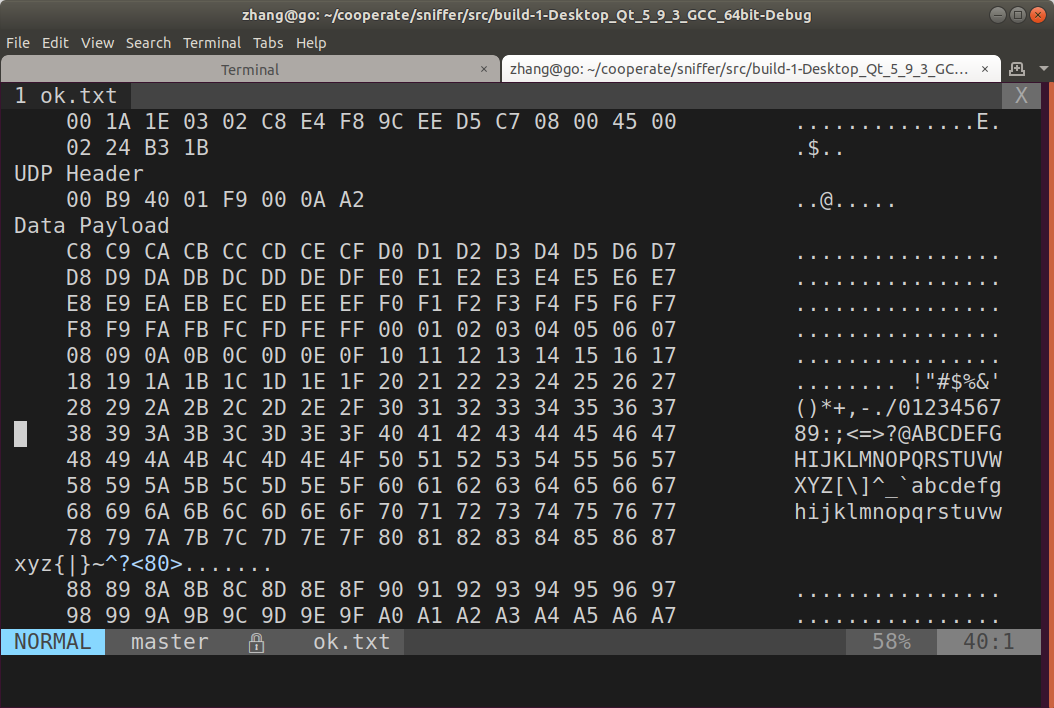


图4-16 存储文件样例

第五章 **遇到的问题及解决方法**

5.1 张侨伦

（1） 直接用QT的IDE，需要管理员权限的时候不能运行，刚开始的时候不知道为什么程序会崩溃，后来经过谷歌搜索发现解决方法。可以制定用命令行运行，这时输入密码就可以用管理员权限来运行。

（2） IP重组问题。如何找到属于同一个包的分组。

经过网上查找，看到了linux内核ip重组的实现方法，发现是根据原地址，目的地址，标识以及协议做一个hash来判断属于同一个包。

（3） 刚开始没有注意到网络字节序和主机的字节序，导致一些数据不对。甚至有时候程序直接崩溃。

后来经过调试以及上网的搜索，发现是大数端和小数段的问题。而导致崩溃是因为我用了C++，如果有数字错了，指针指错了位置，就可能引用无法引用的位置，导致出错。

（4）由于大一之后都没有用过C++，刚开始上手就出现了一些问题，比如C++变量的作用域，C++指针的用法，C++多个头文件，多个源文件如何组织。后来通过复习，上网查资料，以及看书知道了这些问题的解决方法。

（5）刚开始想存网络数据包的时候，使用std：：vector一直崩溃。后来经过搜索，C++的std库里面的数据结构，在比较现代的版本里面，最好是不要有结构体内有指针，很容易导致崩溃。后来改变了策略，不再结构体里面放指针。在结构体里面放的都是一份拷贝。

（6）C++和指针相关的字符串的问题。由于这个项目使用QT，C和C++，C的字符串和std::string以及QString三种互相转换就很重要，刚开始经常需要上网查阅，后来总结之后就容易解决多了。

（7）有一次碰到数据类型大小导致的问题。

更确切的说，是size\_t和int大小不同的问题。后来在定义的地方改正之后，发现还是会有问题，经过检查，发现是函数返回的地方写的仍然是错的。

由于函数返回值处的类型转换，导致返回值出现了问题。

（8）有一次碰到程序莫名崩溃。一直找不出问题，检查了指针，指针也没有乱指。后来发现是有一个死循环。

（9）写ip重组的时候，发现重组的过程中一直少掉一个包。后来通过debug，发现是新建一个重组的集合时（程序中用vector实现），没有在结构体中插入第一个包。

（10）程序在第一次点击run的时候可以正常运行，stop之后重新run的时候就会出错。

发现原来类里面有一个对象是指针，它指向传进来的设备名指针，但是设备名指针会被回收。这个时候就会出现指向的位置不对。于是我把它改成了string类型，需要用到的时候转换成c的字符串数组类型。

2.1 陈卓

正文内容

2.1.1 三级标题

正文内容

第六章 **体会与建议**

6.1 张侨伦

由于首次接触C++网络编程，同时对编程中使用到的QT并不了解，这个编程过程历时较长，也碰到了很多问题，这也导致了到最后期限也没有完成所有的功能。自己在之后的过程中也应吸取教训更为妥善合理地安排时间，而未能完全解决的问题我也将在寒假中进行进一步地完善。

由于个人对于网络编程比较陌生，对C++也不熟悉，参考很多Stack Overflow以及博客上的讲解，再次一并表示感谢。而通过对该课程作业的完成。我对于抓包嗅探以及C++编程，图形界面编程熟悉了很多，同时对于课程的理论部分有了更为深入的了解。最后，感谢老师与助教在整个学期中的悉心指导！

6.2 陈卓

正文内容

2.1.1 三级标