**目录**

[**1.** **课程目标** 1](#_Toc501801327)

[**2.** **问题描述** 2](#_Toc501801328)

[**3.** **系统设计** 2](#_Toc501801329)

[3.1开发环境的选择和搭建 2](#_Toc501801330)

[3.2项目具体的介绍 2](#_Toc501801331)

[3.3流程图 6](#_Toc501801332)

[**4.** **测试样例** 7](#_Toc501801333)

[（1）初始界面 7](#_Toc501801334)

[（2）显示接口信息 7](#_Toc501801335)

[（3）配置接口： 8](#_Toc501801336)

[（4）抓包 9](#_Toc501801337)

[（5）过滤 12](#_Toc501801338)

[（6）分类排列 13](#_Toc501801339)

[（7）图表分析 13](#_Toc501801340)

[**5.** **体会和建议** 14](#_Toc501801341)

1. **课程目标**

除了继续深造或作网络技术研究之外，相关就业的方向有：网络应用研发、网络运维和网络升级改造等。本课程设计可选择围绕网络应用研发或网络规划展开。

利用已经掌握的计算机网络及套接字编程基本知识和技能，根据课程设计要求，编写网络通信程序；利用已经掌握的网络原理和技术，根据需求，规划网络。

在整个课程设计的过程中，充分发挥个人主观能动性，主动积极参与，有意识培养自己的分析问题解决问题的能力；碰到不太熟悉的东西，查资料、学习和讨论，采用各种手段直到问题解决。以组为单位开展设计，有组织能力和意愿的同学担任组长，协调工作，培养大家的团队合作精神。

所以，课程设计的主要内容归纳如下：

（1）自由成组，选择一个题目；

（2）讨论、分析，确定最终题目，分工，形成设计任务书（附件1）；

（3）需求分析，进行概要设计和详细设计，形成设计文档（过程必须，文档可选）；

（3）编码、调试、修改；

（4）制作安装文档和使用手册（可选）；

（5）撰写设计报告。

1. **问题描述**

实现一个类似wireshark功能的电脑抓包软件，抓取进出该机的报文，并做统计分析。

1. **系统设计**

## 3.1开发环境的选择和搭建

1.开发所用到的类库：jdk8：Java语言开发工具包

jpcap.jar： 抓包类库

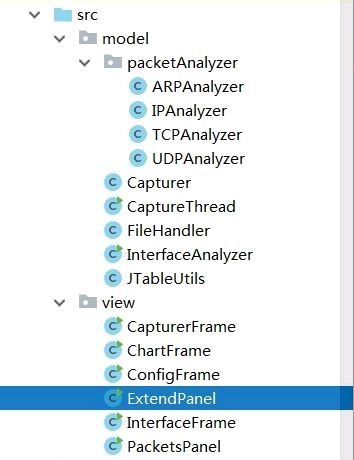
jfreechart.jar ： 图表类库

2.开发环境：windows

开发软件：intelliJ IDEA

## 3.2项目具体的介绍

1.目录：



2.类的说明

项目代码采用模型（model）和视图（view）分层模式，使整个代码结构更加清晰，易于维护。”

ARPAnalyzer.java：封装了对于ARP报文的分析。

IPAnalyzer.java：封装了对于IP报文的分析。

TCPAnalyzer.java：封装了对于TCP报文的分析。

UDPAnalyzer.java：封装了对于UDP报文的分析。

Capturer.java：模型和视图直接的沟通者，搭建起接口配置视图，文件视图和抓包的线程类的桥梁。

CaptureThread.java：抓包的主要类，继承Thread类，使其抓包的过程和视图的渲染互不影响，而且可以控制抓包的开始和停止。

FileHandler.java：将抓取到的包保存到文件中以及从文件中读取抓取到的包。

InterfaceAnalyzer.java：分析出所有接口的名称，描述及mac地址。

JTableUtils.java：有关JTable的一些操作方法。

CapturerFrame.java：主界面。

ChartFrame.java：柱状图和饼状图的展示界面。

ConfigFrame.java：抓包配置界面。

ExtendPanel.java：每个包的具体展开界面。

InterfaceFrame.java：接口展示界面。

PacketsPanel.java：抓取的包展示界面。

3.一些主要方法的说明：

Capturer.init()：初始包的总数目，每种类型包的数目以及包的队列list的清零。

CaptureThread.run()：该线程的运行方法，根据是否是从文件打开调用不同的方法。

CaptureThread.dealPacket()：对于抓取的包的具体处理。

FileHandler.saveToFile()：保存文件。

FileHandle.openFile()：从文件中打开。

InterfaceAnalyzer. getMessage()：获取接口的相关信息。

CapturerFrame. actionPerformed()：对于每个不同的点击事件渲染不同的界面。

ChartFrame. getTablePanel()：获取要展示相关包的列表。

ChartFrame. getPiePanel ()：获取饼状图。

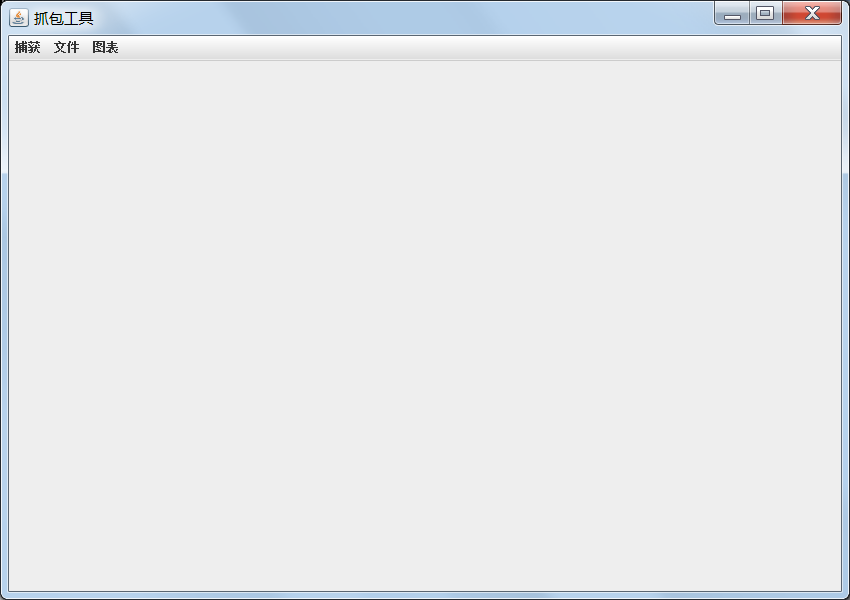
ChartFrame. getBarPanel ()：获取柱状图。

ExtendPanel. analyze(Packet packet) ; 分析报文信息

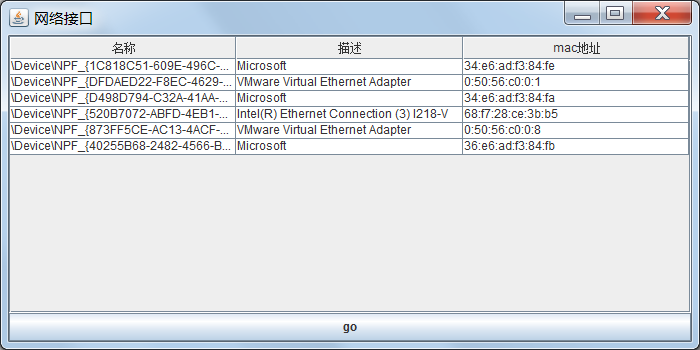
## 分析流程.png3.3流程图

1. **测试样例**

**（1）初始界面**

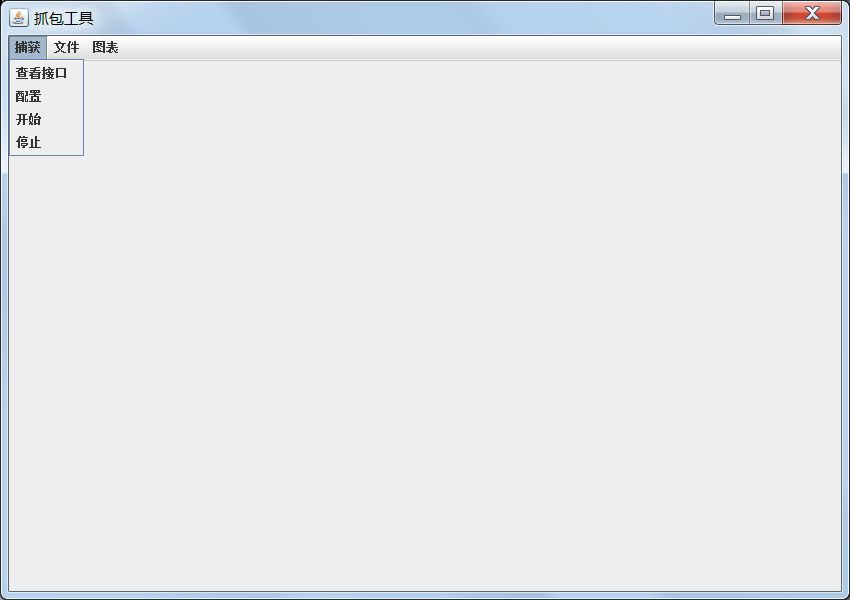
****

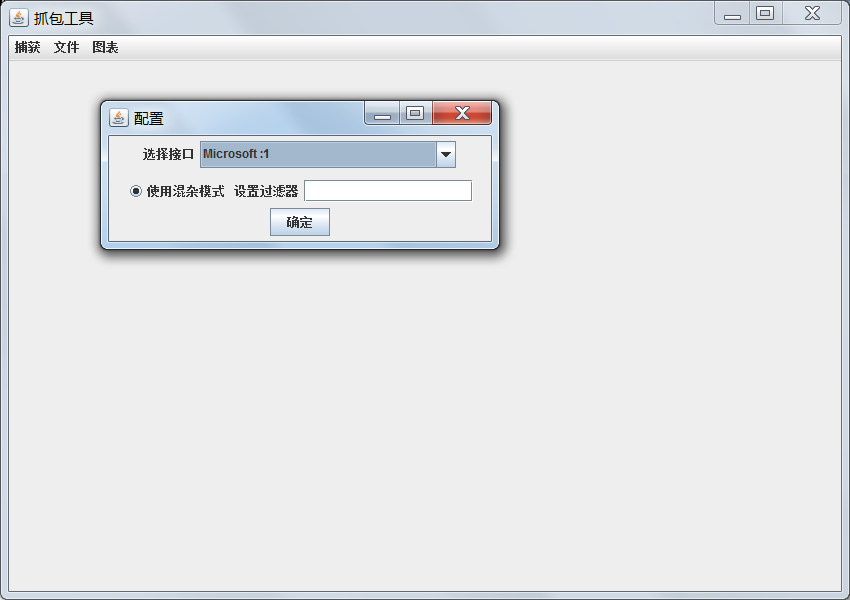
## （2）显示接口信息

****

## （3）配置接口：

可以选择在上一步查看接口中点击接口并按下方go按钮选择；或者点击配置菜单，在如下图弹出的选项栏中进行高级配置，如设置过滤器等

****

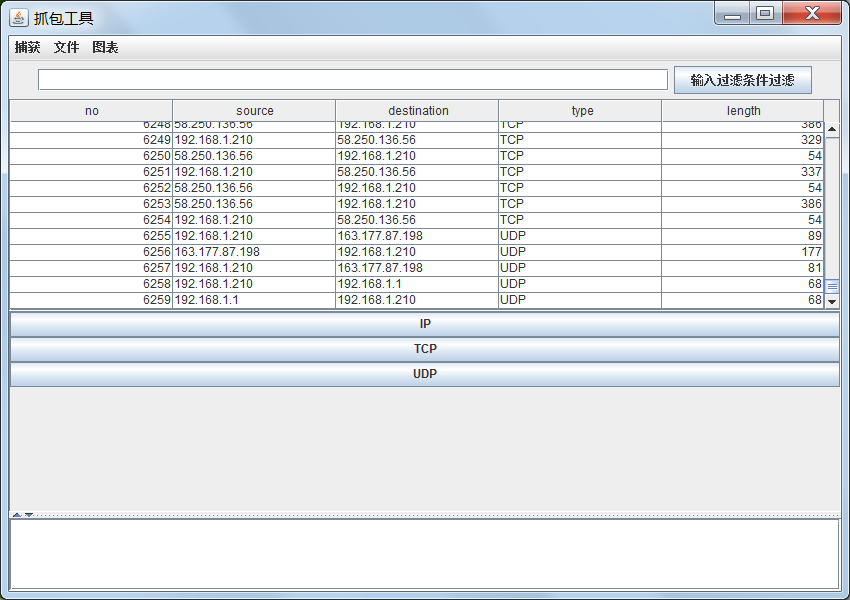
****

## （4）抓包

点击捕获-->开始，程序开始抓包。

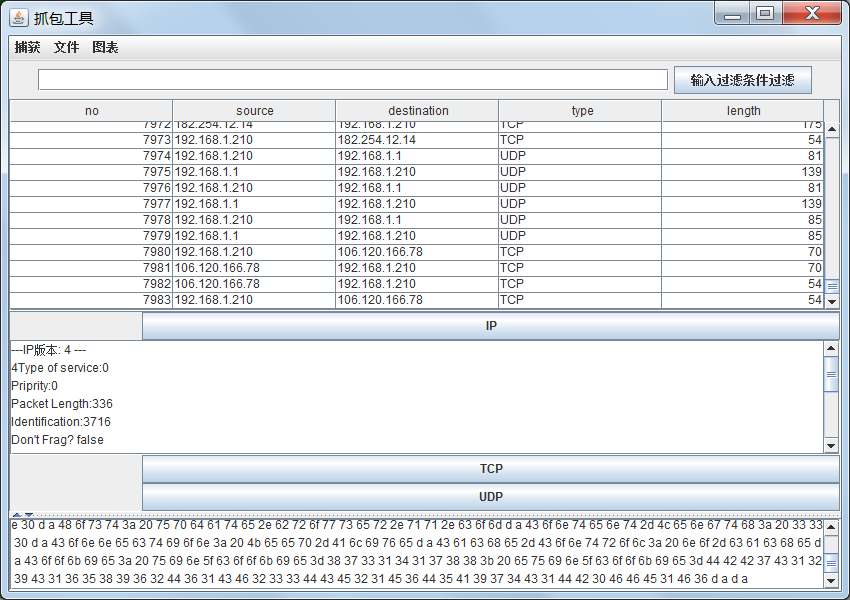
点击捕获—>停止，程序停止抓包。

抓包界面如图所示：

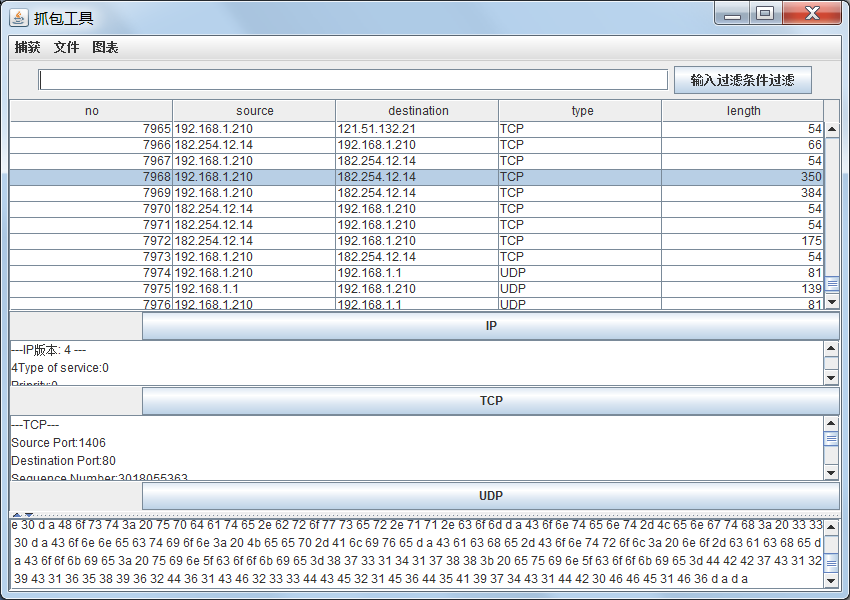
****

分别以网络层和传输层两层展开点击IP，TCP或UDP查看相应报文信息：

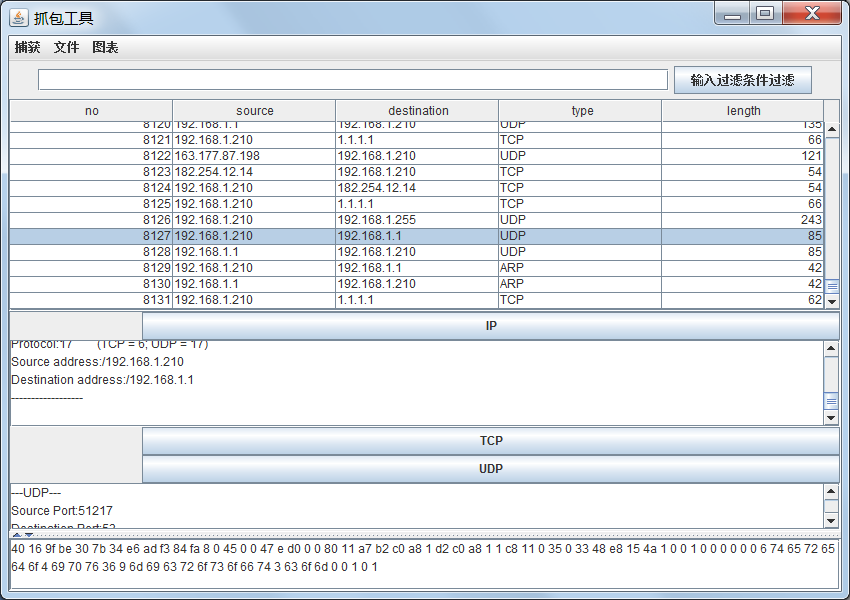
IP：

****

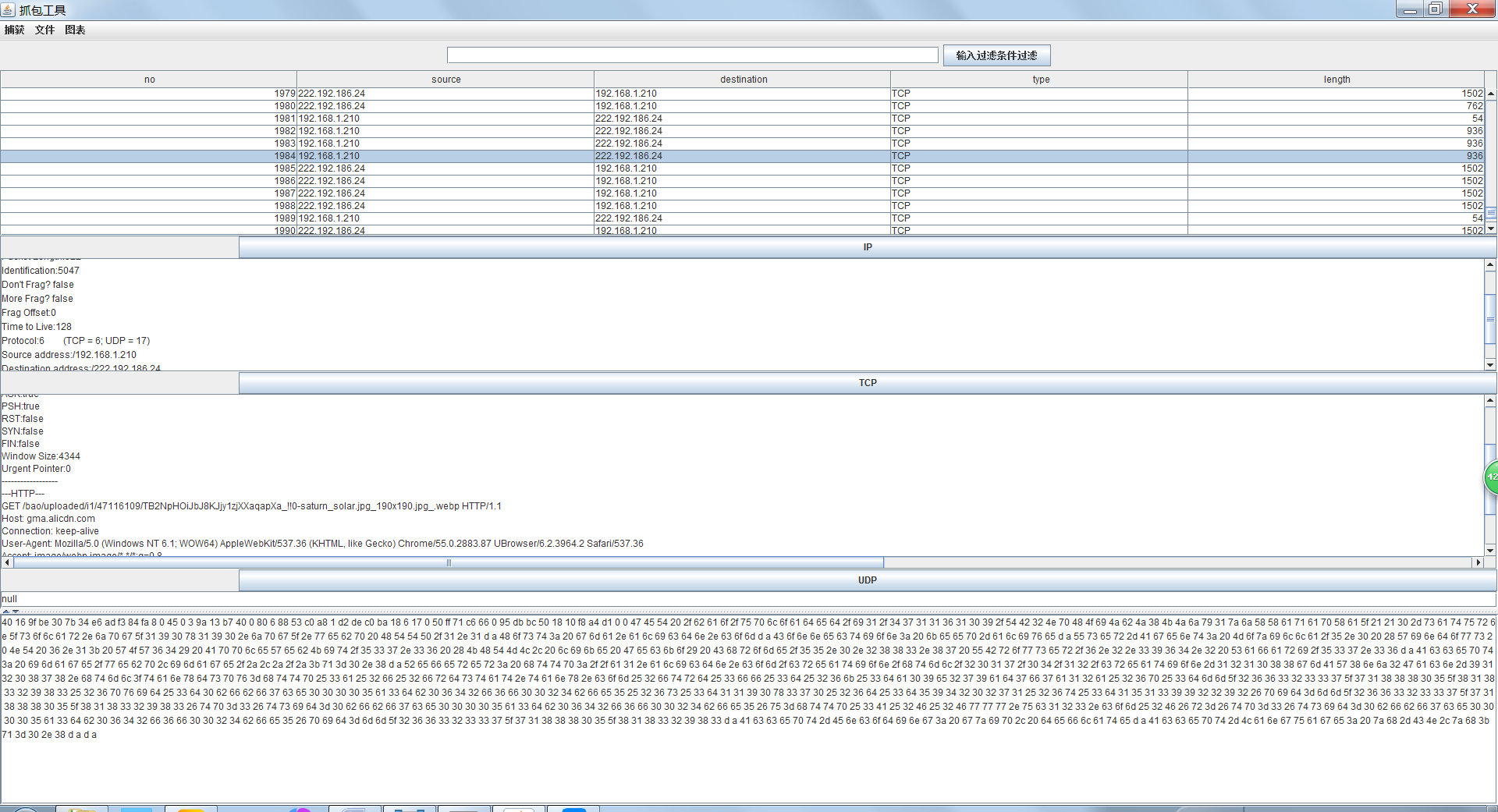
TCP：

****

UDP:

****

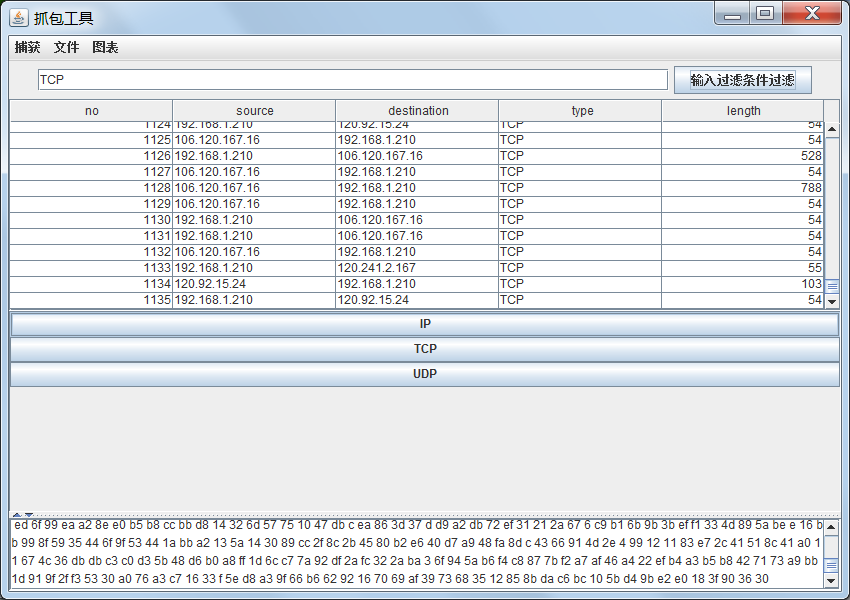
总览：

****

## （5）过滤

可在列表上方的输入框内输入过滤要求，并点击过滤按钮进行筛选

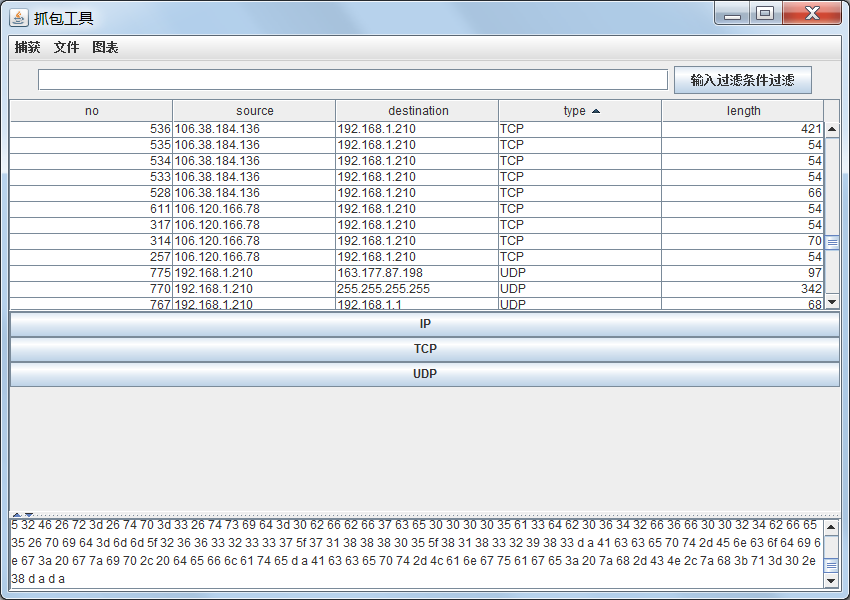
下图以TCP为例：

****

## （6）分类排列

在点击列表最上部的每一个属性名称（no，source等）后，列表中的项会以对应的属性进行分类排列

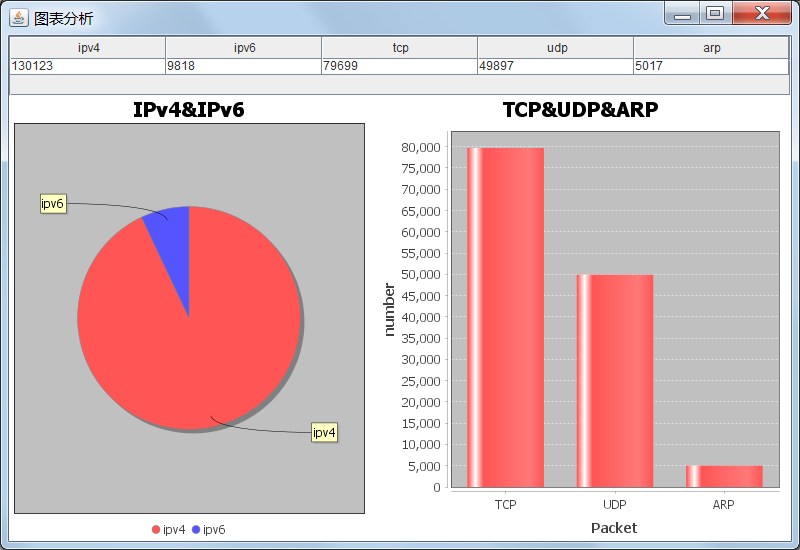
以type为例：

****

## （7）图表分析

可对抓取的报文的种类和数量进行统计分析

如下图所示：

****

12:10开始抓包

晚12：10结束

1. **体会和建议**

本次大作业收获颇多，首先再次重温了计网相关的知识,对于包方面有了更深的理解，其次了解到了Jpcap的使用以及Java GUI编程，其中也遇到了不少问题，比如一开始包展示那里点击具体的包，展开的结果不正确，经排查发现是下标转换的问题。期间我们经过多次讨论，终于把项目完成，从零到项目的完成，我们体会到了很多。

**参考文献**

[1] Keita Fujii.Jpcap开发文档. <https://github.com/mgodave/Jpcap.2017-11-15>