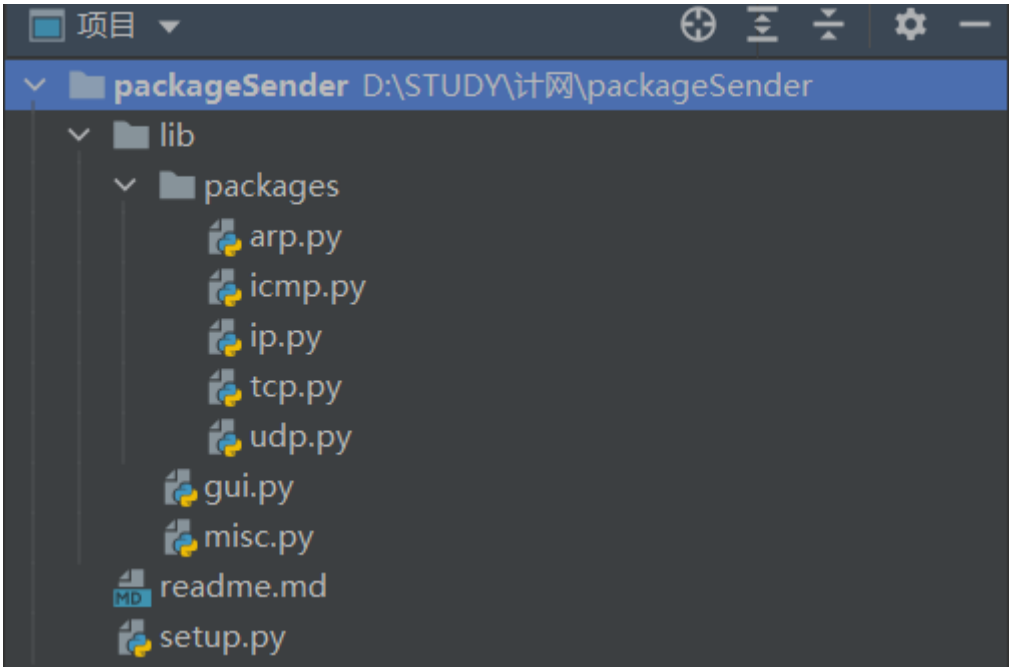
My Sender 文档

概述

本次大作业耗时三天半左右，由于定项目、开工得晚，就没有寻求组队。之所以选择这个项目，一是因为先前已经在EduCoder上的实训进行过一些网络协议包的封装，而且我对python比较熟悉，也知道有socket和struct的库可以调用，有过相关经验；二是觉得这个项目需要涉及计算机通信网的网络层和传输层，可以对整个网络结构有更深入的了解。

本项目是SJTU-2020-IS301的课程大作业，内容为网络发包器，可以用于网络测试。项目源码结构如下，setup.py是整个程序的主入口,gui.py是图形界面设计，misc是一些杂项函数，如检测用户输入是否为合法地址等。package目录下五个文件分别是对应协议的实现，他们的结构相似，由**init**、pack、send等函数组成，调用了python的socket和struct等库。本项目使用的python库均为原生库，无需再安装别的库。

发包的核心思路就是通过struct的pack函数进行封装，按照协议格式组装首部，并于数据拼接，再通过套接字socket进行连接、发送。



该项目实现的功能如下：

- 发送IP报文，手动设定双方IP地址，flag位，ttl，上层协议，option和data，并记录发送历史。
- 发送TCP报文，手动设定双方IP地址和端口，flag位，窗口大小，sequence number,acknowledge number，option和data，并记录发送历史。
- 发送UDP报文，手动设定双方IP地址，端口和data，并记录发送历史。
- 发送ARP报文，手动设定双方IP地址，端口和data，并记录发送历史。
- 发送ICMP报文，手动设定双方IP地址，端口和data，并记录发送历史。
- 以上的功能皆可通过简易的图形用户界面完成。

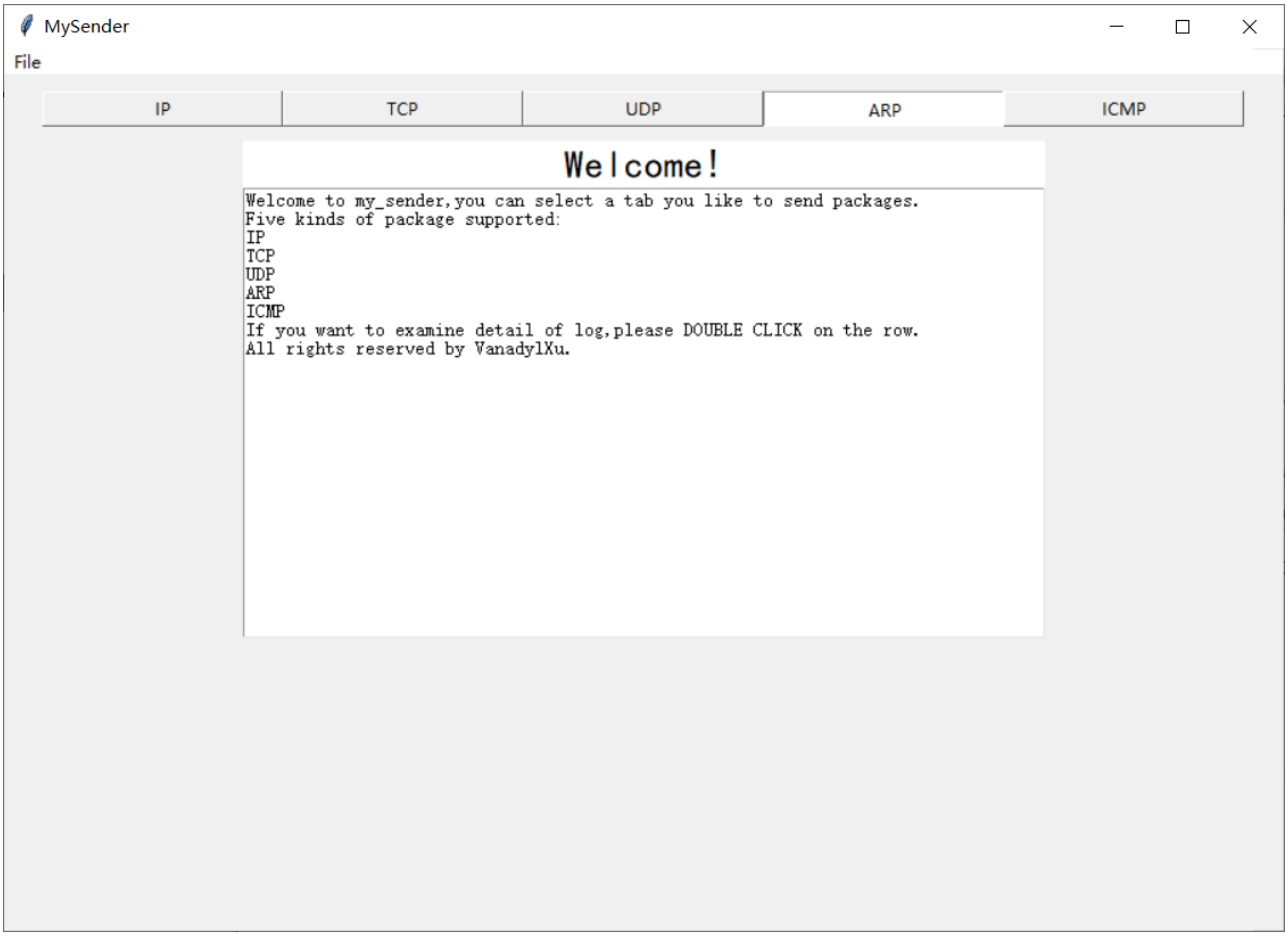
项目的编译通过pyInstaller实现，打包的exe文件可以直接运行。

由于项目完成在期末，各方面压力繁重，我也一个人在google参考了许多代码，其中也遭遇了一些困难，但还是完成了这个项目，虽然这个工具仍然非常简陋，而且可能或多或少存在一些问题,也欢迎助教、老师和我交流，并指出我的不足和一些改进的建议。

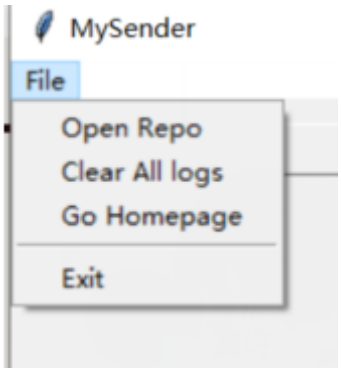
P.S:由于没有正确地设置虚拟环境，所以编译出的可执行文件会有一点大，如有带来麻烦，十分抱歉。

GUI实现

本项目的GUI通过python的原生库Tkinter实现。主界面由一个菜单栏和五个选项卡和相关界面、欢迎界面组成。



菜单栏有4个按钮，分别是：跳转到仓库、清除历史记录、返回首页和退出。



各具体协议界面由参数输入、历史记录和详细信息组成，大部分协议参数可以由用户输入，但是少部分参数如IP版本和ARP的硬件地址长度和类型等由于几乎只使用一种版本，故已经设置好，没有开放输入。

详细信息和历史记录界面和选择选项已经做了禁止用户输入的处理，避免篡改或导致混淆，也设置了滚轮，可以方便得进行查看多条消息。部分参数也设置好了默认值，可以避免一上来一大堆的参数需要用户一个一个输入。程序在初始化时自动获取本机IP地址和MAC地址并填入输入框，并提供在测试时可以通过的目标地址。在用户点击send按钮时，将输入传入后端并发送，在双击历史记录时，展示历史记录的详细信息。

GUI较为臃肿，也存在一定的代码冗余，这是由于我能力不足没有经验，底层没有封装好所致。

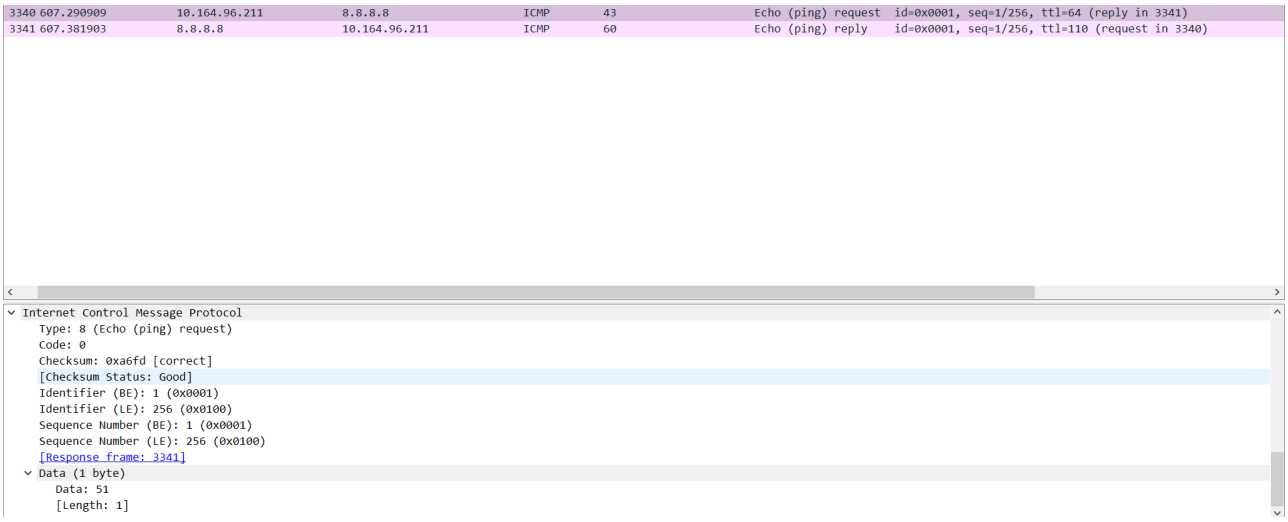
ICMP实现

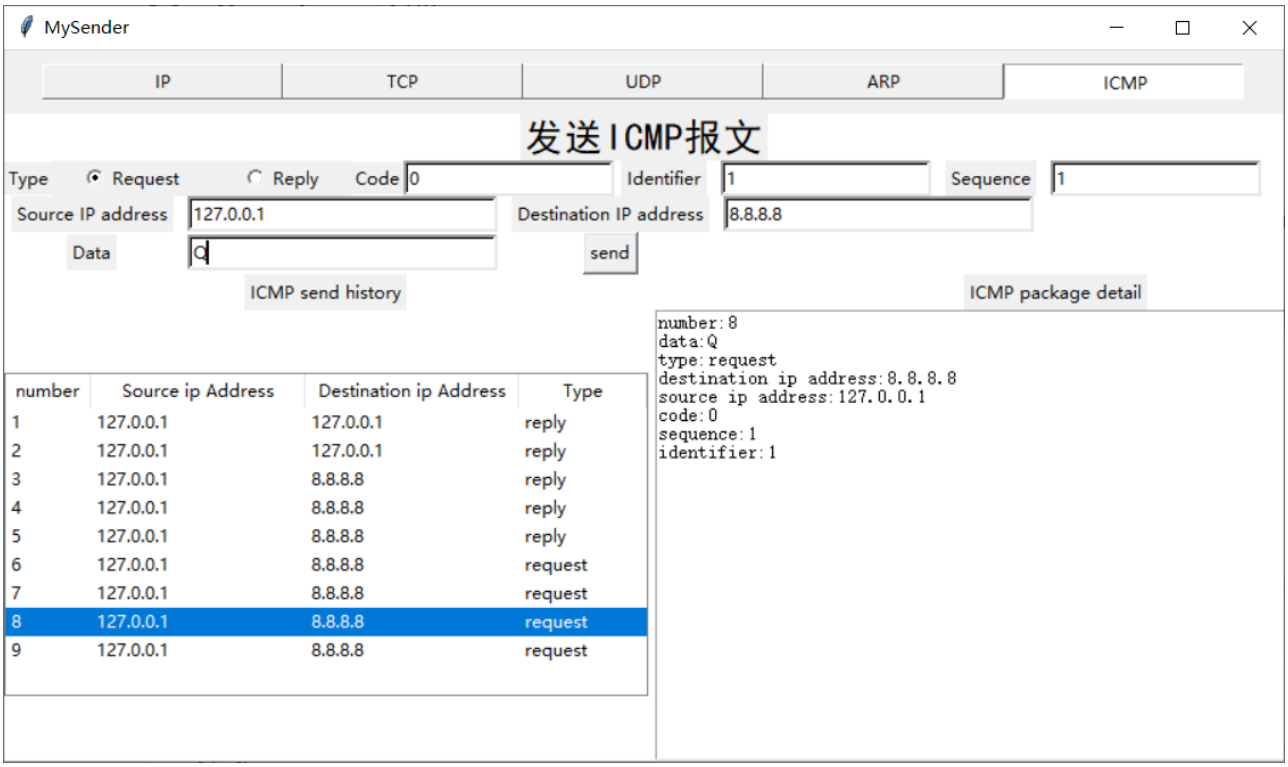
ICMP的type有很多，但常用的正确类型是Reply和Request，故没有设置其他类型。

可以成功ping外网并得到回复，此前难以得到回复，wireshark会提示no response found，估计是因为id默认值为0或数据段为空所致，加入后问题解决，成功实现（Q的编码为0x51）。

若不能收到回复，请检测发送报文的类型是否为Request！！或者检查id是否为0！！

ICMP的校验和已经写过，直接进行了复用。按照ICMP协议头部结构进行封装，通过套接字进行发送。



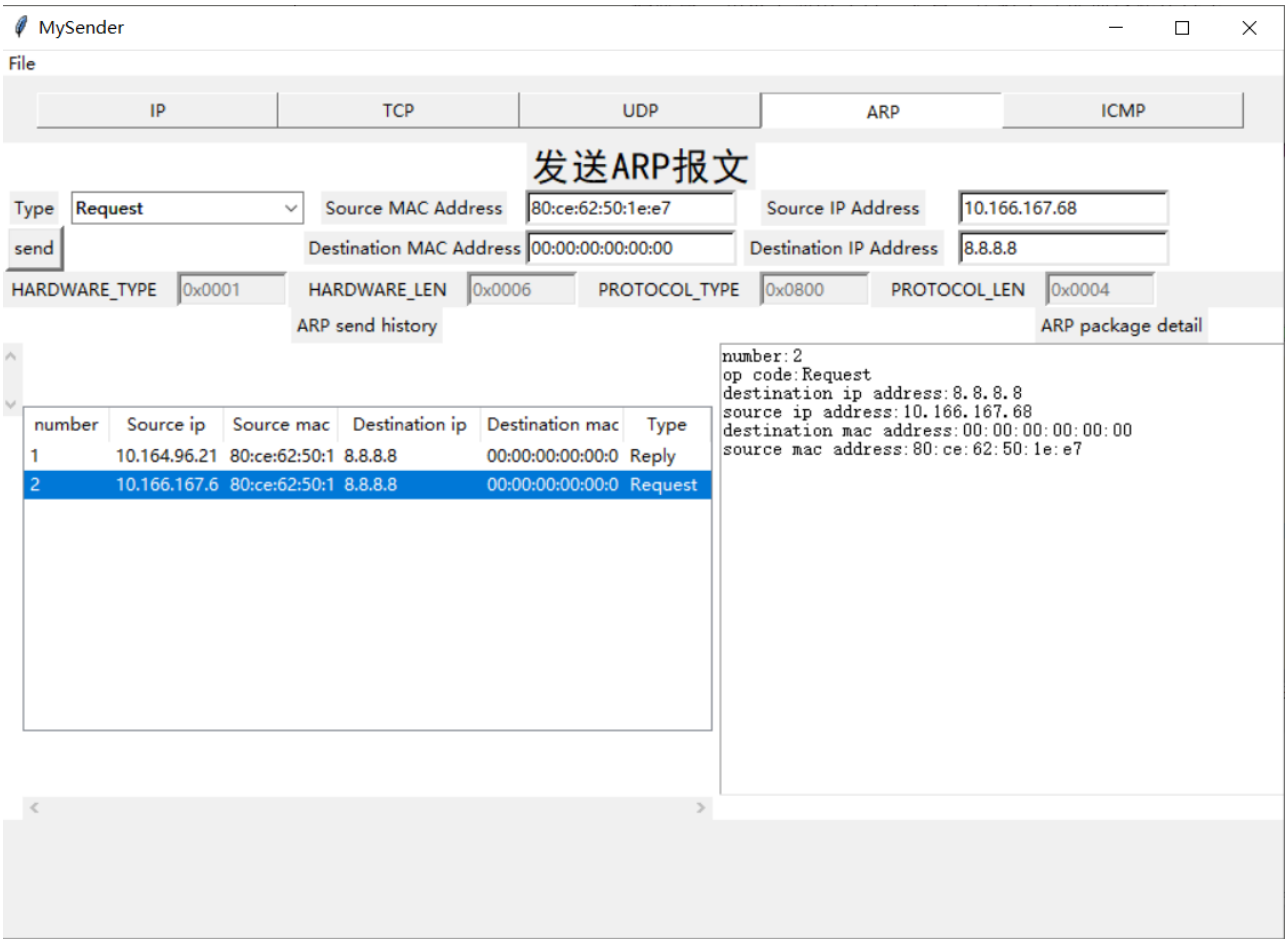


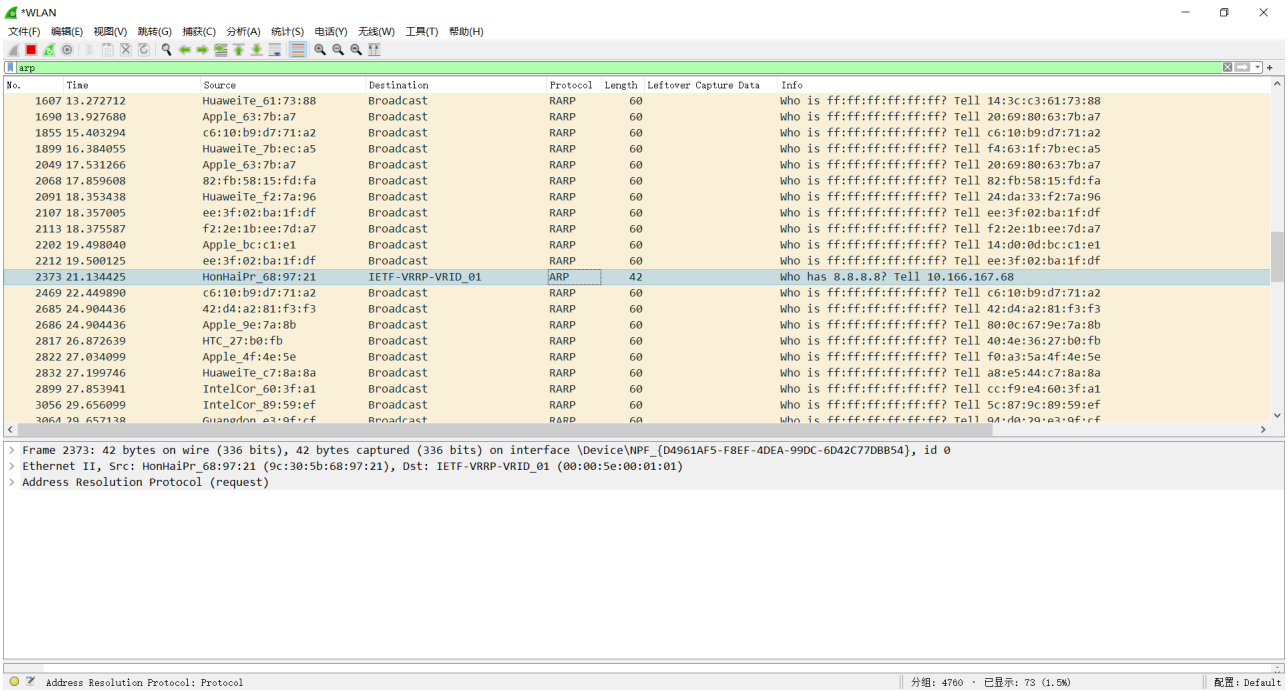
ARP实现

成功图如下。由于网上有关ARP的资料太少，绝大多数都是关于ARP欺骗的内容，只好用了另一个包scapy实现。其封装比较简单，相当于调用了另一个包，原封不动地把参数传过去。

下面四个属性分别针对以太网和IP协议，不可更改。

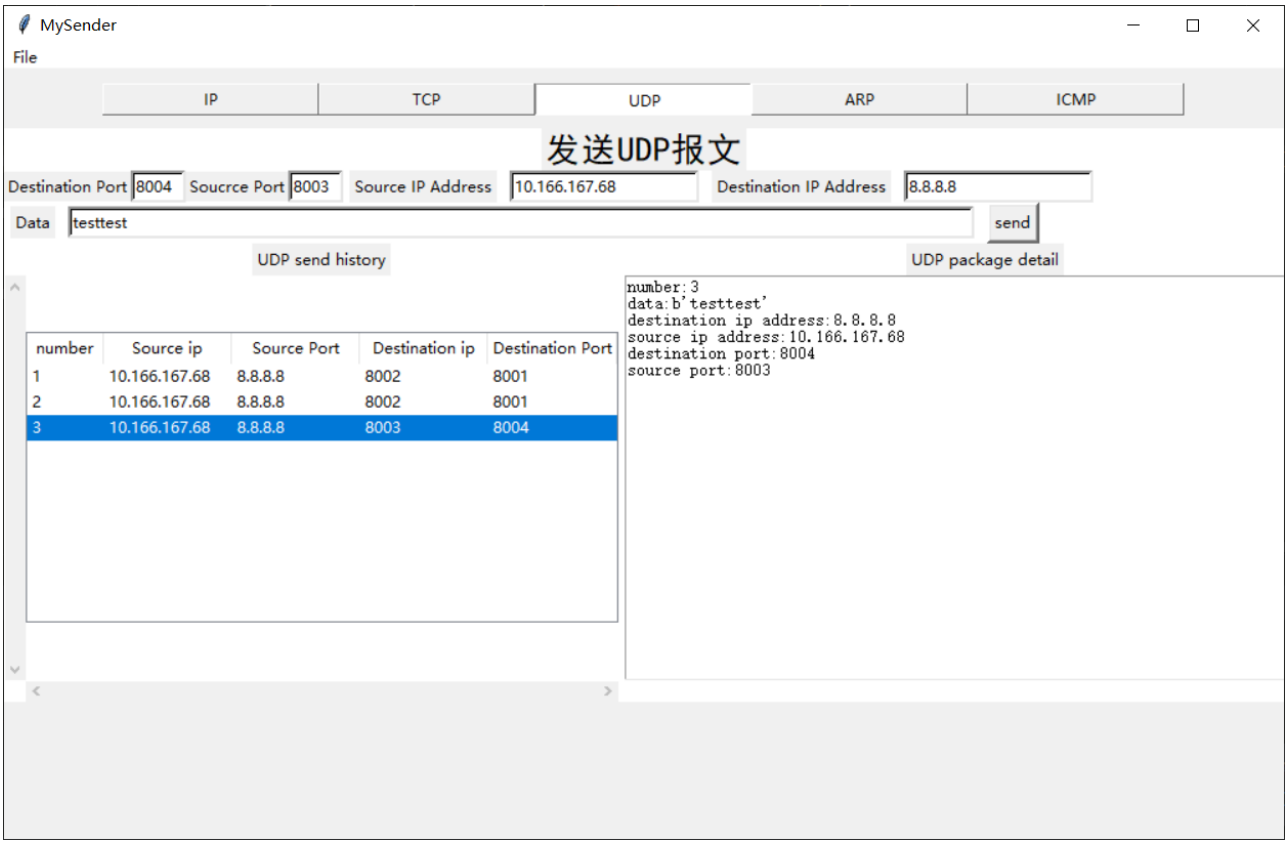
以下为发送成功的抓包内容。

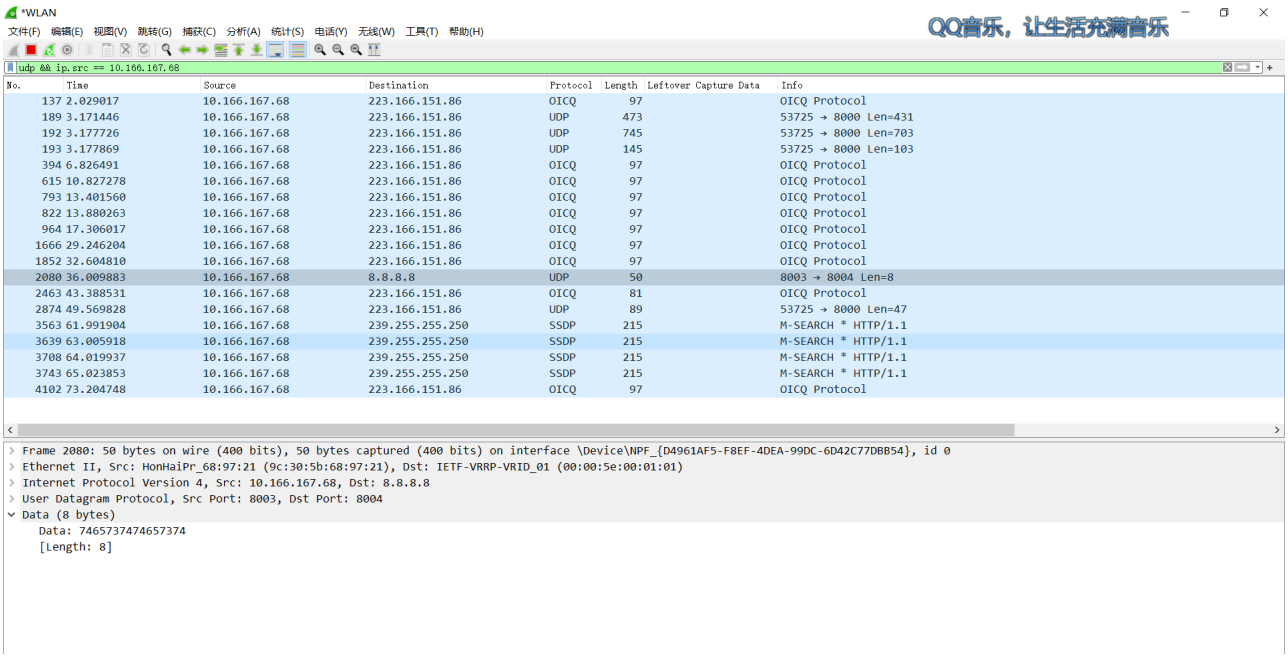




UDP实现

UDP是不保证可靠传输的无连接传输的协议，内容比较简单。



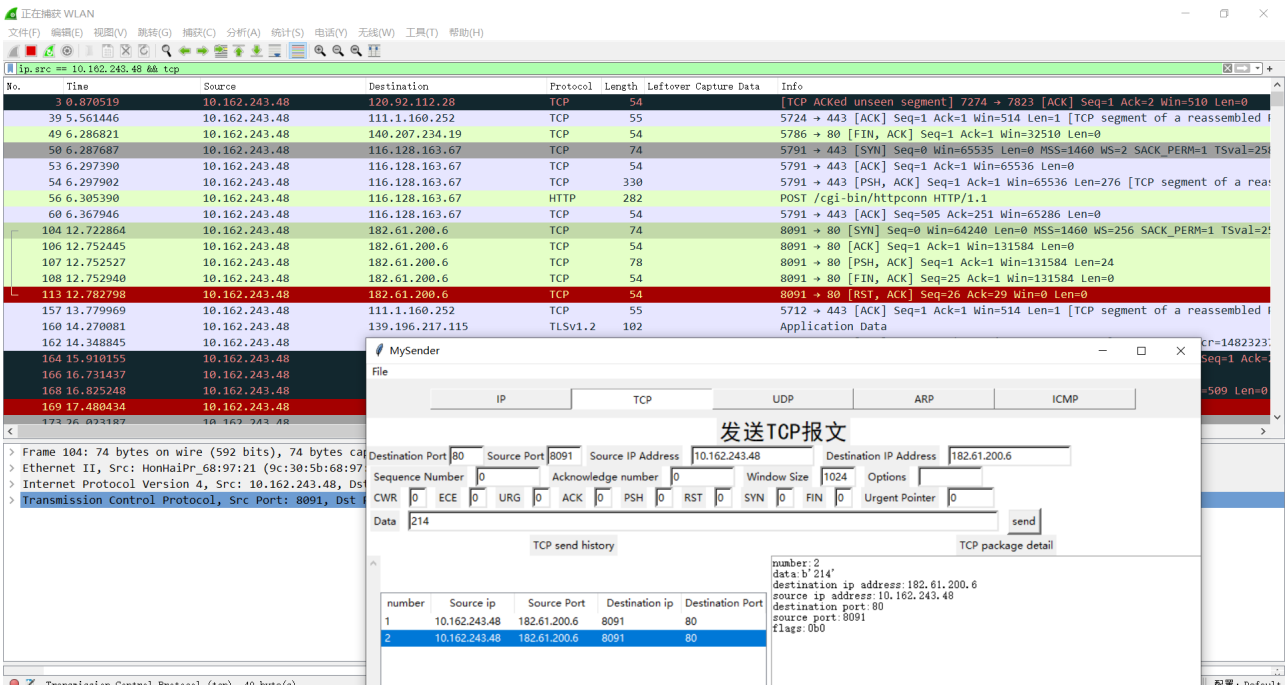


TCP实现

TCP的可输入参数较多，抓包成功记录如下。

TCP的套接字连接方式有少许不同，主要是需要建立连接，使用socket.SOCK_STREAM套接字。在发送时也具有一定区别，TCP直接使用send(msg)函数，而UDP则需要使用sendto(msg,(addr,port)).

```
mysocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```



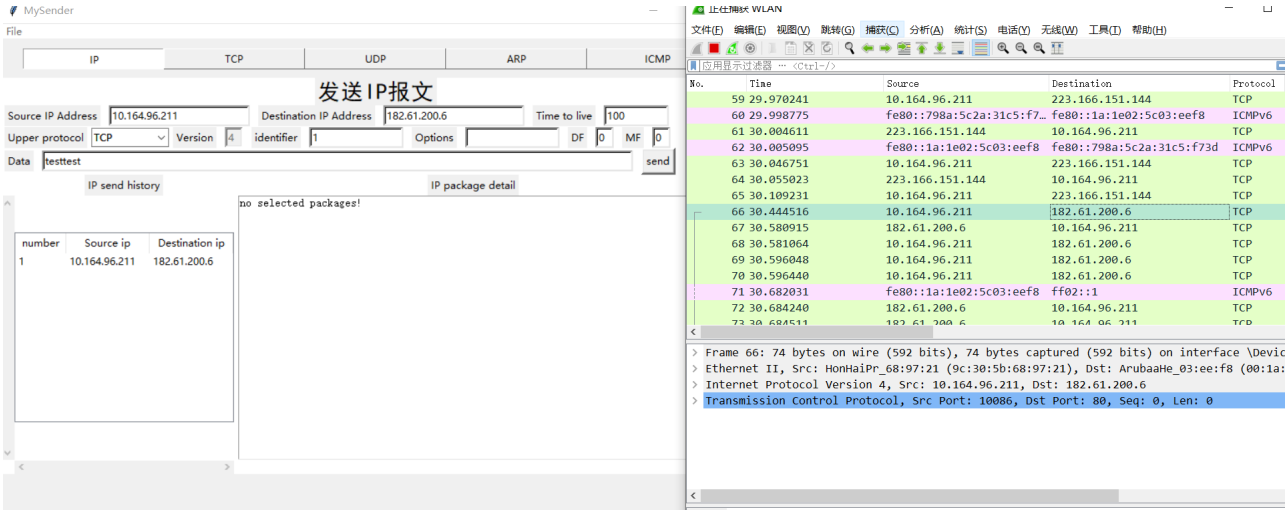
在TCP运行时，若报错

`OSError: [WinError 10013] 以一种访问权限不允许的方式做了一个访问套接字的尝试。`

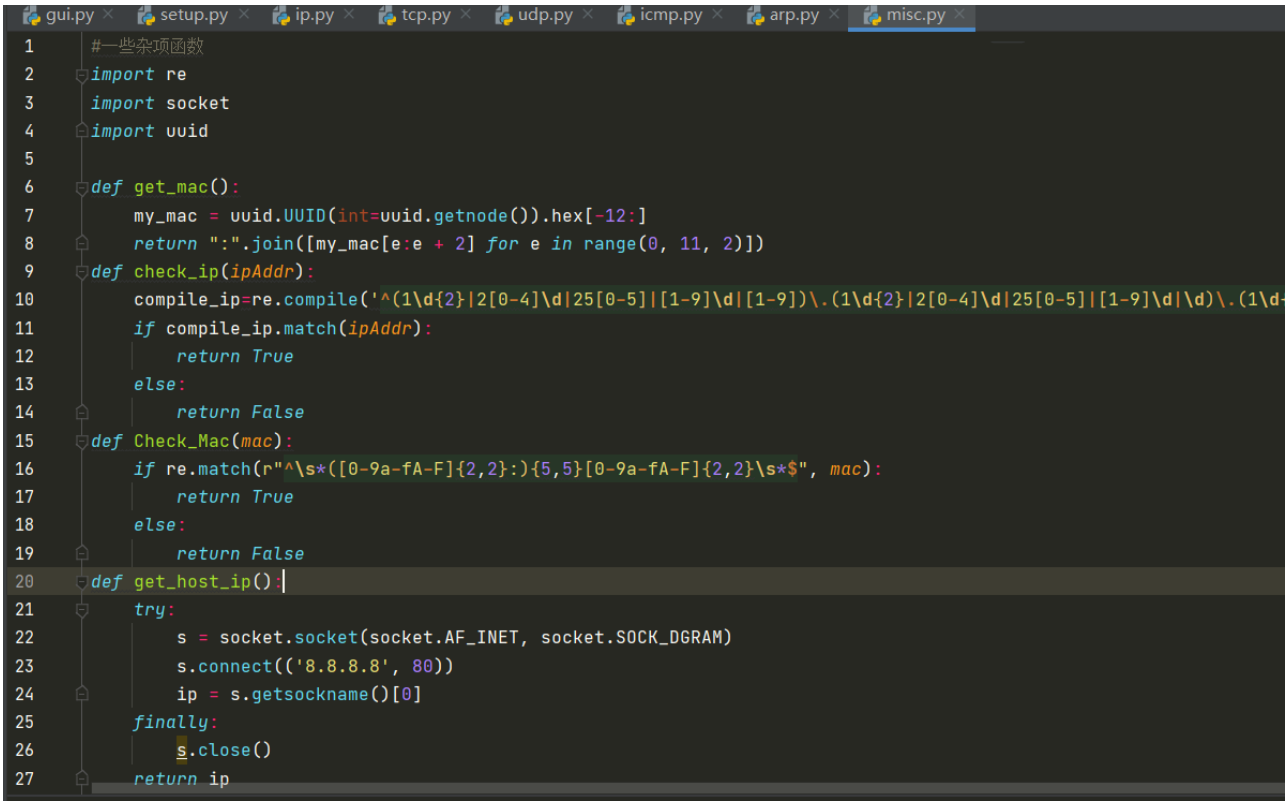
则是由于本地端口被占用，需要更换一个端口。

IP实现

上层协议提供了ICMP、TCP和UDP三个选项。TCP的套接字连接同样有不同。由于IP协议位于网络层，没有端口，故连接时使用了内部定好的端口号。



一些杂项函数



本项目用的杂项函数只有四个，分别是IP地址和MAC地址的检测和获取。检测通过正则表达式实现，获取则调用了一些库进行实现。

遇到的问题及解决

在整个开发过程中，遭遇了许多困难，如图形界面组件乱跑、包无法成功发送等问题。这些都没有现成的答案，只能通过自己寻找问题，并且不断进行修改才能解决。

对于图形界面，我去寻找了各种组件的使用的样例和文档，并且进行调整使其符合美观。

在进行打包的过程时，一些格式化的参数问题困扰了我很久，如一些数据类型应该是整数而不是字符串，变长字符串如何进行打包等等，IP地址应该以什么形式打包等等。在ARP的头部组装中，曾一直出现不需要整数类型参数的错误，但我反复排查之后并没有整数数据，后面发现可能由于python的语言特性，长度为1的字节变量可能会被表示成一个256以内的整数，在我加上了Bytes函数之后问题得到了解决。

我第一个开发的协议是ICMP实现，结束时发现一直抓不到包，或发出去没有收到回复，经过了很多排查才发现原因是id段置零和类型不是Request，代码实现并没有问题。在开发结束后也觉得自己的代码有所不足，如GUI承担了过多的功能，可以把详细信息等内容交给底层的包，不同选项卡之间有很多相似，可以通过数组等方式进行复用等等。

在TCP开发过程中，由于它的套接字连接方式不同，也进行了许多尝试。如果只是单纯把UDP的搬过来进行协议名称的修改，则会报错“传入不必要的参数”。在查阅资料后发现TCP的建立参数就是不同的，也需要先建立连接，后续发送时就不用再输入地址。在这里出错也反映出我对理论知识掌握不牢，虽然大概知道原理，但是实际应用的时候就有些捉襟见肘。

体会与建议

本次项目由于时间紧张，没有按照完整的开发流程进行，只是在查阅资料，想好图形界面的大概形状就直接开始一边学习一边开工，其中也存在很多不足和遗憾。由于没有经验，我在图形界面花费了非常多的精力，研究各个组件的参数和摆放，如何绑定事件，传递参数，尽量让它看起来不那么丑，这虽然锻炼了我的能力，但是在一定程度上与大作业的目的有冲突，也压缩了我对后端协议实现的内容。如在进行TCP包的测试时，一度找不到可以提供建立连接的端口，最后发现我们学校的80端口可以连接，但是没有详细地对包的各个参数进行检查，因为TCP需要建立连接会发送一些其他的包，比较复杂。

但是做完了这个项目我仍然十分自豪，它基本实现了目标的功能，也具有一定的代码量和复杂性，这是我个人宝贵的开发经历，我的能力也得到了全方位提升，不仅是前端后端的开发，也有整个项目的统筹，使用git进行版本管理，使用函数和类进行面向对象的实现和对课程理论知识的学习等等。

建议就是可以让助教学长参与更多课程内容，如答疑和大作业的一些指导等等。并且觉得课程的教材过于老旧了，在上课时也比较少用到，都是PPT为主，希望更换。

最后感谢我的授课教师姚立红老师，老师在课堂上细致讲解的网络协议是本项目的基础，其中也穿插了一些关于大作业的要求和内容，让我对项目有了更清晰的认识；也感谢耐心审阅我不那么美观的代码的助教。谢谢！