

**长沙理工大学**

计算机与通信工程学院

实验报告

**实验名称**:**\_\_ 嗅探器设计和实现 \_**

**年级**： **大三 班级**： **网络2102班**

**学号**： **202108060918 姓名**： **王俊\_\_\_\_**

**完成日期**： **2024 年 5 月**  **30 日**

**授课老师： 夏卓群\_\_\_\_\_\_\_\_\_\_\_\_\_**

一、实验目的与任务

1.不限语言，设计并实现一个嗅探器

2.实现一个功能比较简单的、具有图形界面的嗅探器，主线程响应用户界面操作，工作线程完成抓包等工作；

3.能够解析出IP层和传输层的协议头，能够过滤TCP、UDP等数据包；

4.能够输出文本方式传送的数据包的内容；

5.能够进行简单的流量统计。

二、开发环境与工具

**操作系统：windows 11 专业工作站版**

**开发工具：PyCharm**

**开发语言：Python**

**辅助工具：QTDesigner**

**附加库：PyQt5、scapy、matplotlib、multiprocessing**

三、设计原理

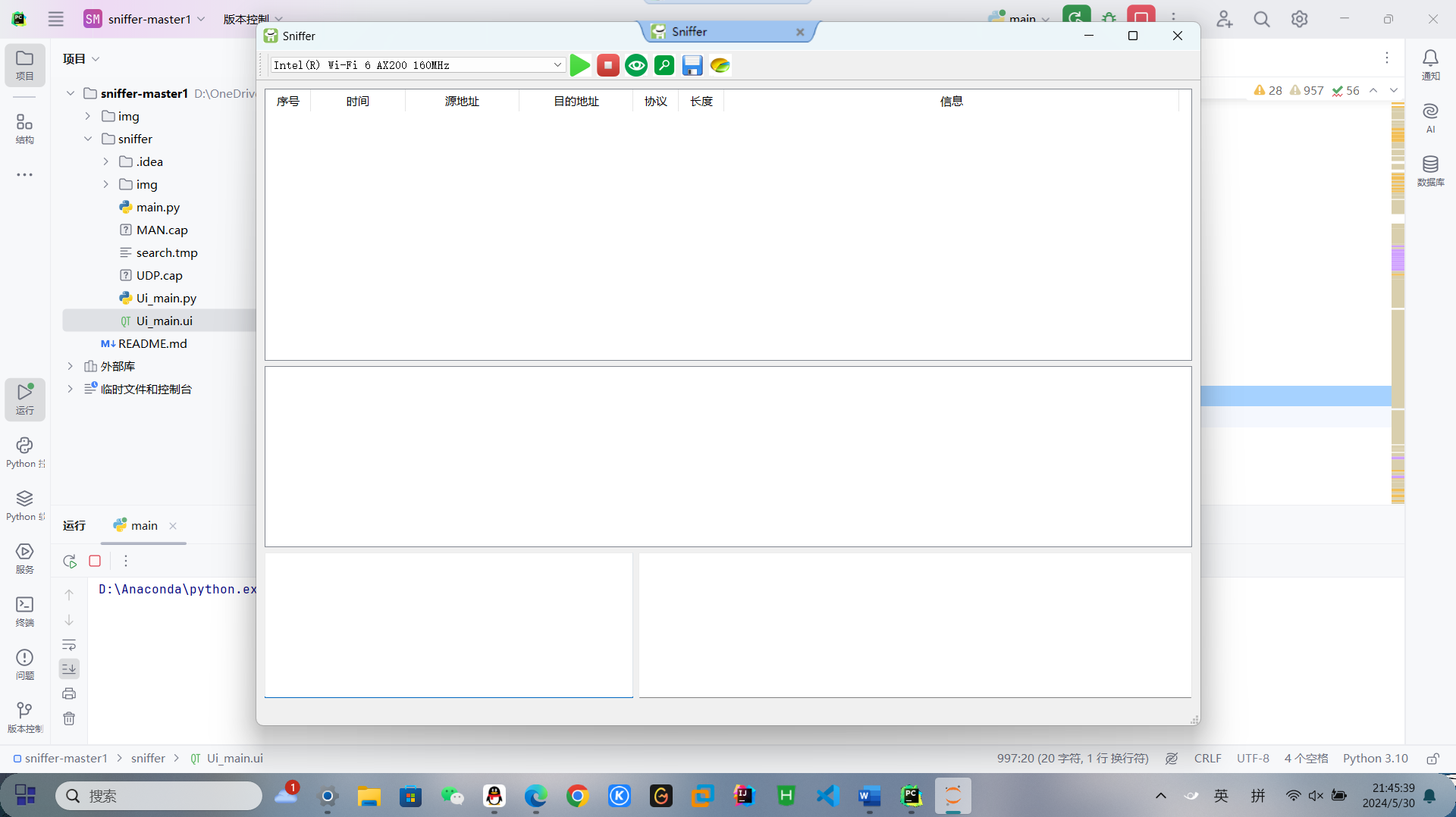
该嗅探器基于pyqt5，scapy，首先在QTDesigner设计美观且适合的界面，然后将该ui界面转换为py文件，在主函数中引用该文件，实现界面的初始化。

在该嗅探器中，可以实现选择网卡、开始、暂停、捕获过滤、显示过滤、保存以及数据统计功能。

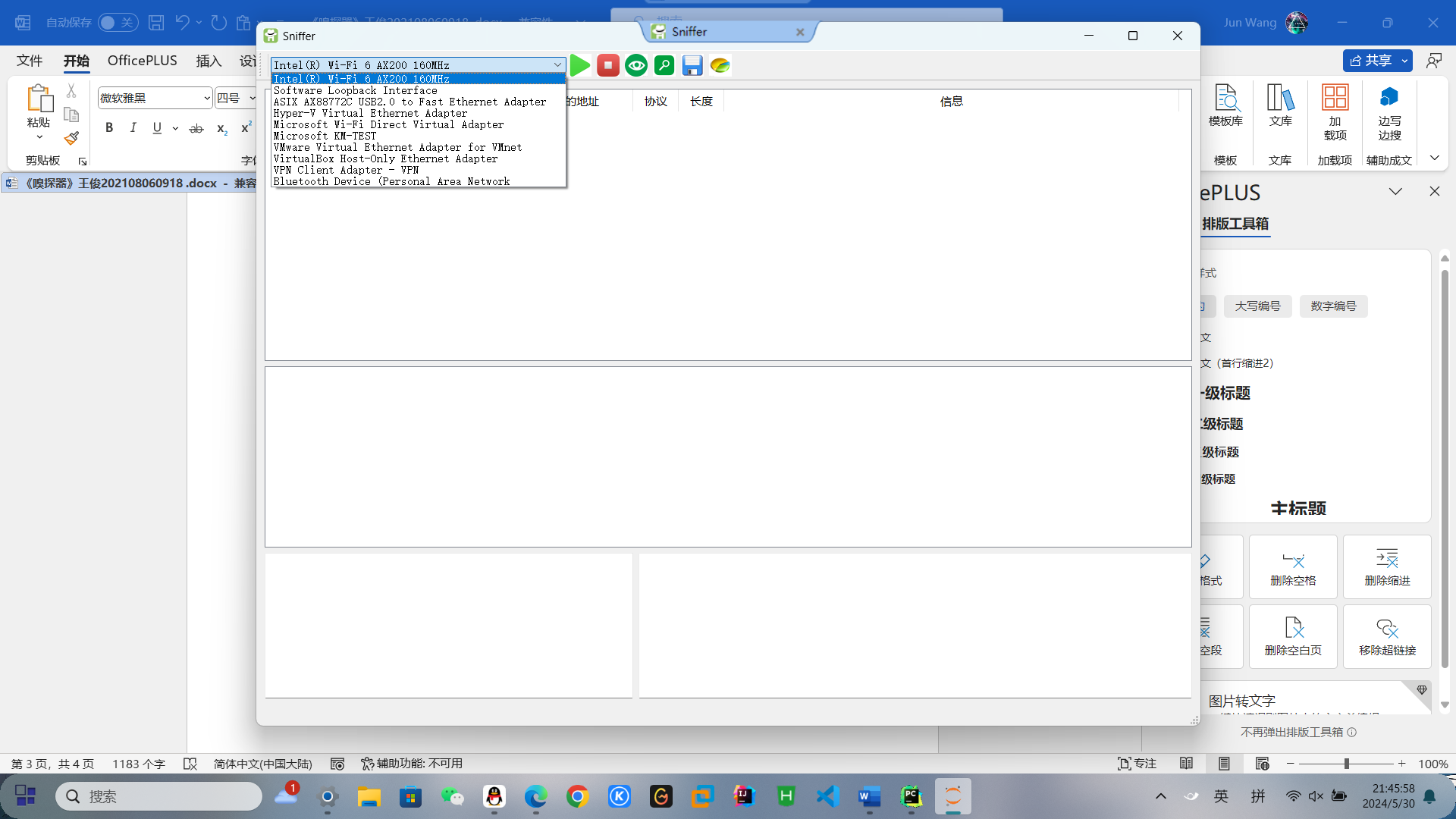
选择网卡功能为一个combox，可以遍历本地所有网卡并显示。开始和暂停功能有两个按钮实现，捕获数据包点击可以显示数据包内容。捕获过滤可以实现指定原目的地址、指定目的IP地址、指定原端口、指定目的端口、指定协议类型以及自定义规则。显示过滤功能可以实现指定协议过滤。保存功能可以实现将捕获的保存在本地，并可以使用wireshark打开。统计数据功能可以统计网络层和传输层的数据包。

四、实验步骤

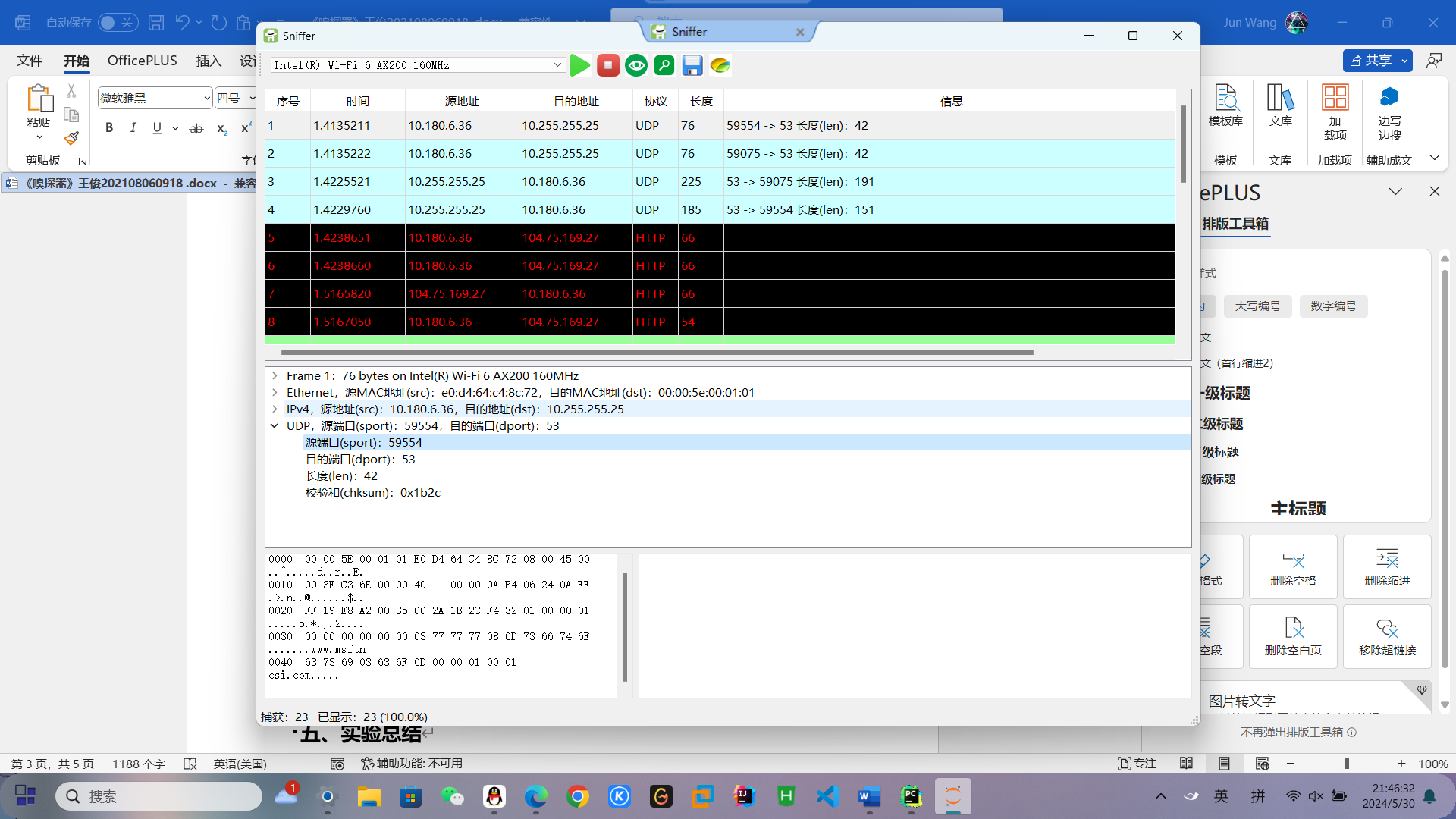
1. 界面设计



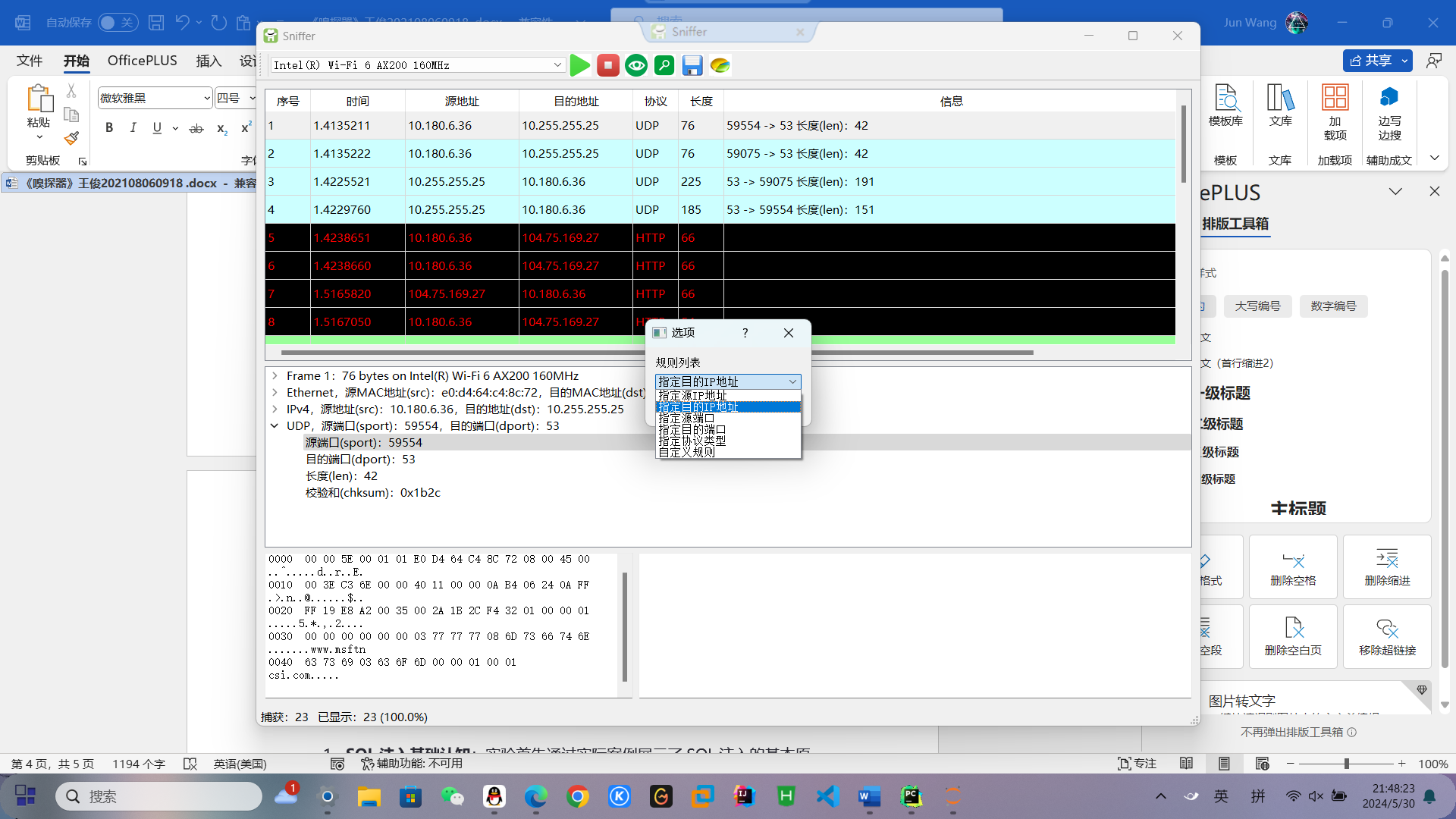
1. 选择网卡



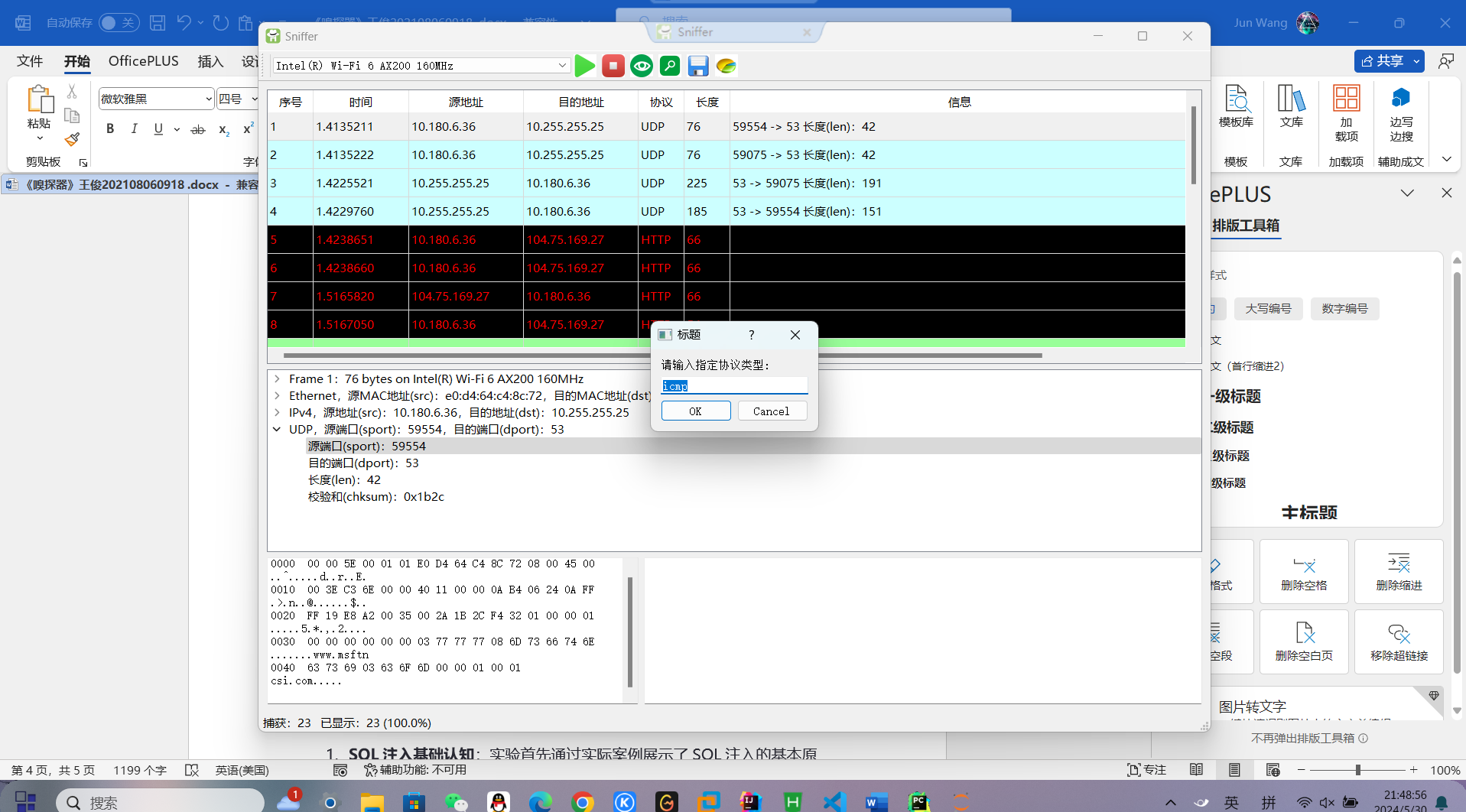
1. 捕获数据包



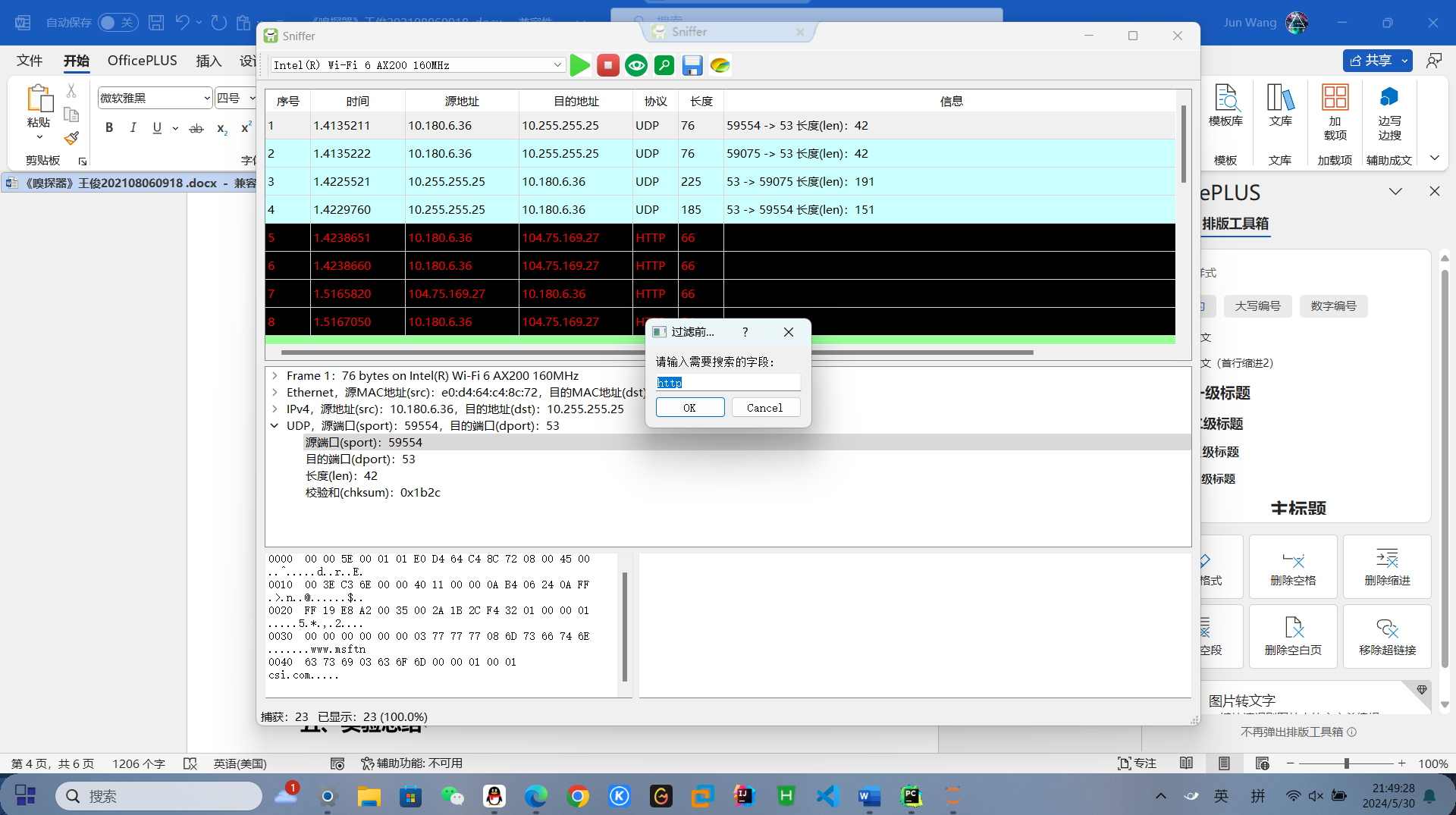
1. 捕获过滤



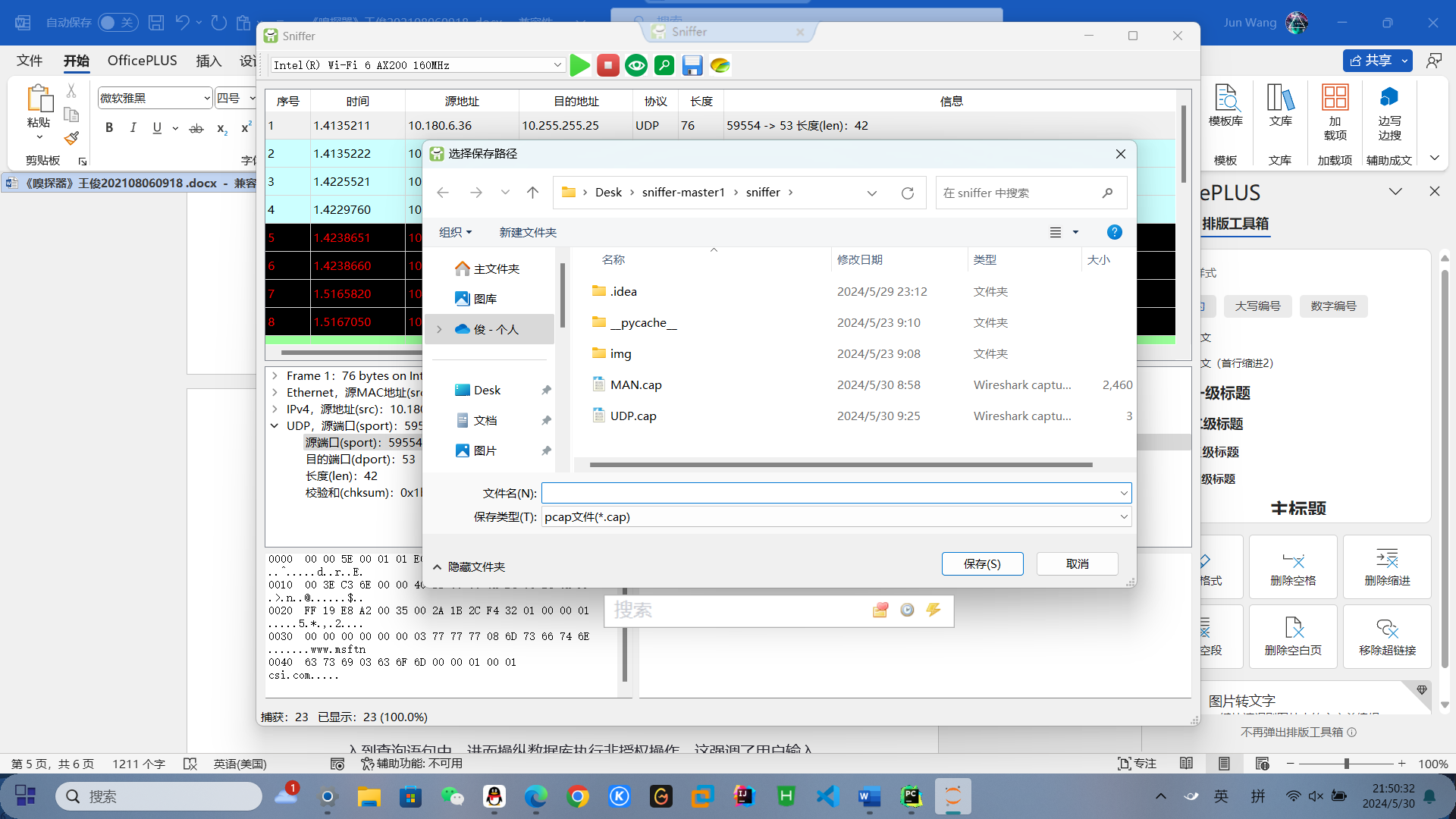
1. 指定协议类型



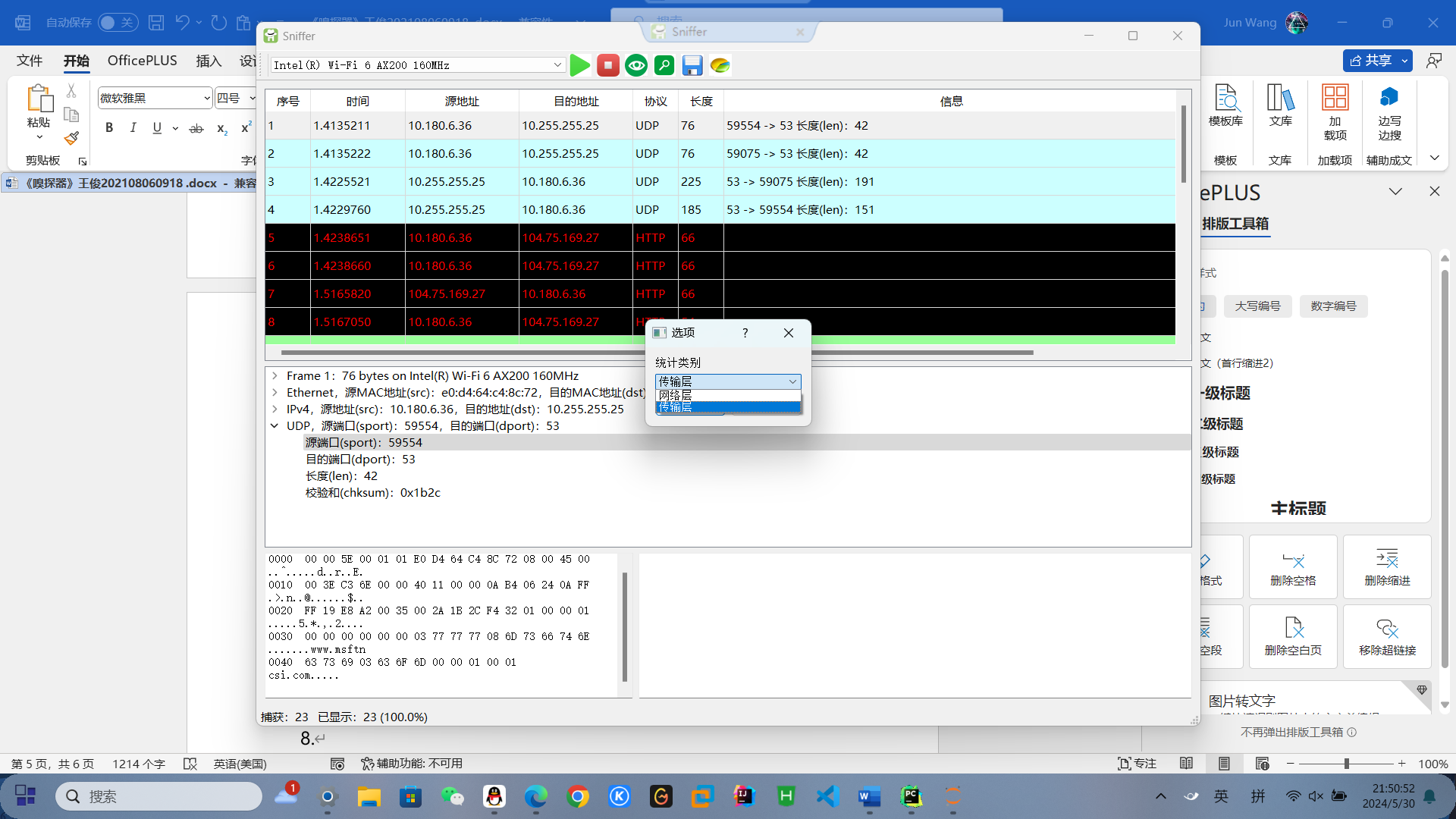
1. 显示过滤

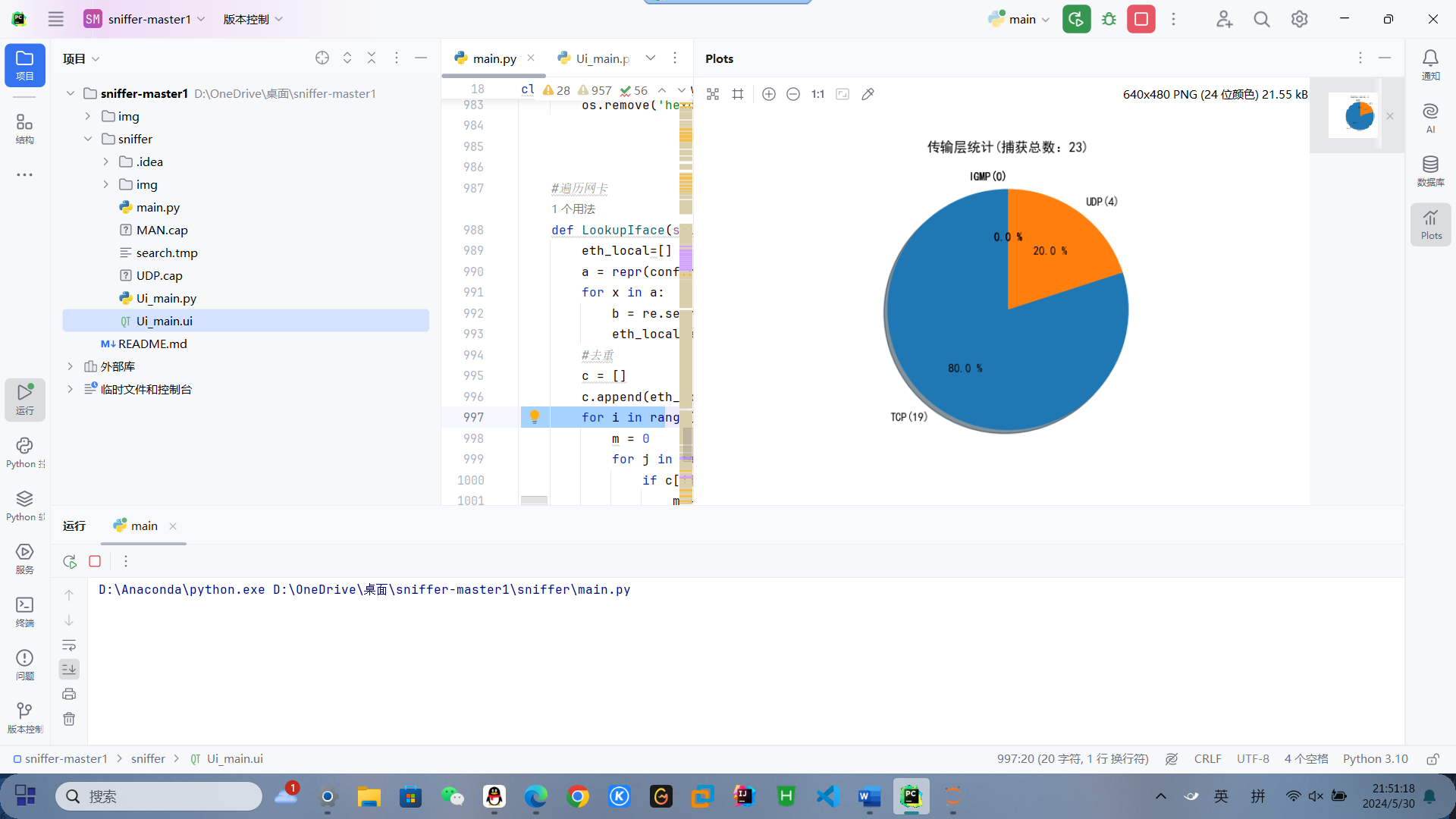


1. 保存



1. 统计数据





五、功能实现关键代码展示

1. 网卡选择

def LookupIface(self):  
 eth\_local=[]  
 a = repr(conf.route).split('\n')[1:]  
 for x in a:  
 b = re.search(r'[a-zA-Z](.\*)[a-zA-Z]',x)  
 eth\_local.append(b.group())  
 *#去重* c = []  
 c.append(eth\_local[0])  
 for i in range(0,len(eth\_local),1):  
 m = 0  
 for j in range(0,len(c),1):  
 if c[j] == eth\_local[i]:  
 m += 1  
 if m==0:  
 c.append(eth\_local[i])self.comboBoxIface.addItems(c)

② 捕获过滤

def PreFilter(self):  
 list = ["指定源IP地址","指定目的IP地址", "指定源端口","指定目的端口","指定协议类型","自定义规则"]  
 item, ok = QInputDialog.getItem(self, "选项","规则列表", list, 1, False)  
 type=0  
 if ok:  
 if item=="指定源IP地址":  
 filter,ok\_1 = QInputDialog.getText(self, "标题","请输入指定源IP地址:",QLineEdit.Normal, "\*.\*.\*.\*")  
 rule = "src host "+filter  
 elif item =="指定目的IP地址" :  
 filter,ok\_2 = QInputDialog.getText(self, "标题","请输入指定目的IP地址:",QLineEdit.Normal, "\*.\*.\*.\*")  
 rule= "dst host "+filter  
 elif item =="指定源端口":  
 filter,ok\_3 = QInputDialog.getInt(self, "标题","请输入指定源端口:",80, 0, 65535)  
 rule="src port "+str(filter)  
 elif item =="指定目的端口":  
 filter,ok\_4 = QInputDialog.getInt(self, "标题","请输入指定目的端口:",80, 0, 65535)  
 rule ="dst port "+str(filter)  
 elif item =="指定协议类型" :  
 filter,ok\_2 = QInputDialog.getText(self, "标题","请输入指定协议类型:",QLineEdit.Normal, "icmp")  
 rule =filter  
 elif item =="自定义规则":  
 filter,ok\_2 = QInputDialog.getText(self, "标题","请输入规则:",QLineEdit.Normal, "host 202.120.2.1")  
 rule=filter  
 rule=rule.lower()  
 self.setPreFilter(rule)

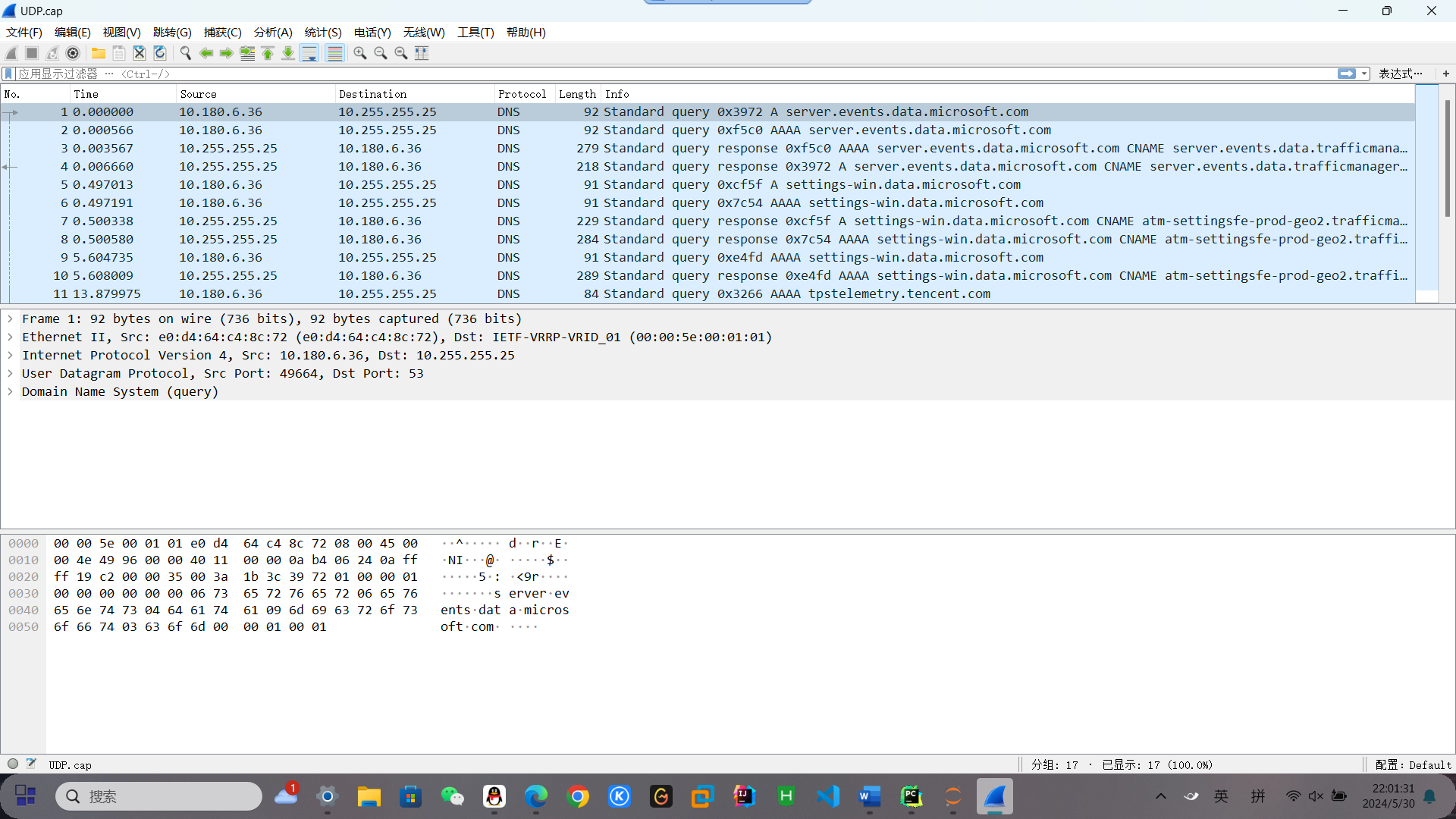
③ 显示过滤

def PostFilter(self):  
 filter,ok = QInputDialog.getText(self, "过滤前需要停止抓包","请输入需要搜索的字段:",QLineEdit.Normal, "http")  
 if ok:  
 global display  
 display = 0  
 if filter == None:  
 for row in range(self.tableWidget.rowCount()):  
 self.tableWidget.setRowHidden(row,True)  
 display += 1  
 else:  
 for row in range(self.tableWidget.rowCount()):  
 if(self.tableWidget.item(row, 7))!=None:  
 p = self.tableWidget.item(row,7).text()  
 packet = scapy.layers.l2.Ether(p.encode('Windows-1252'))  
 f = open('search.tmp','w')  
 old = sys.stdout sys.stdout = f packet.show()  
 sys.stdout = old  
 f.close()  
 f = open('search.tmp','r')  
 data = f.read()  
 f.close()  
 os.remove('search.tmp')  
 obj= re.search(filter.lower(),data.lower())  
 if obj is None:  
 self.tableWidget.setRowHidden(row,True)  
 display += 1  
 else:  
 self.tableWidget.setRowHidden(row,False)

④ 数据包统计

def statistics(self):  
 global count  
 global display  
 if count != 0:  
 percent = '{:.1f}'.format(display/count\*100)  
 self.statusbar.showMessage('捕获：%s 已显示：%s (%s%%)' % (count,display,percent))

⑤ 数据包在wireshark的使用



⑥ 线程嗅探

class SnifferThread(QtCore.QThread):  
 HandleSignal = QtCore.pyqtSignal(scapy.layers.l2.Ether)  
 def \_\_init\_\_(self,filter,iface):  
 super().\_\_init\_\_()  
 self.filter = filter  
 self.iface = iface  
 def run(self):  
 sniff(filter=self.filter,iface=self.iface,prn=lambda x:self.HandleSignal.emit(x))  
class pdfdumpThread(QtCore.QThread):  
 pdfdumpSignal = QtCore.pyqtSignal(bool)  
 def \_\_init\_\_(self,packet,path):  
 super().\_\_init\_\_()  
 self.packet = packet  
 self.path = path  
 def run(self):  
 self.packet.pdfdump(self.path,layer\_shift=1)  
 self.pdfdumpSignal.emit(True)

六、实验总结

本次实验主要是设计和实现嗅探器，总体来说，本次实验十分顺利，不仅设计了美观的界面，且完美实现了嗅探器的功能，但本次实验仍有以下不足：

1. 无法解析DNS协议
2. 可能出现bug直接卡死
3. 传输层数据统计是文字叠在一起，看不清