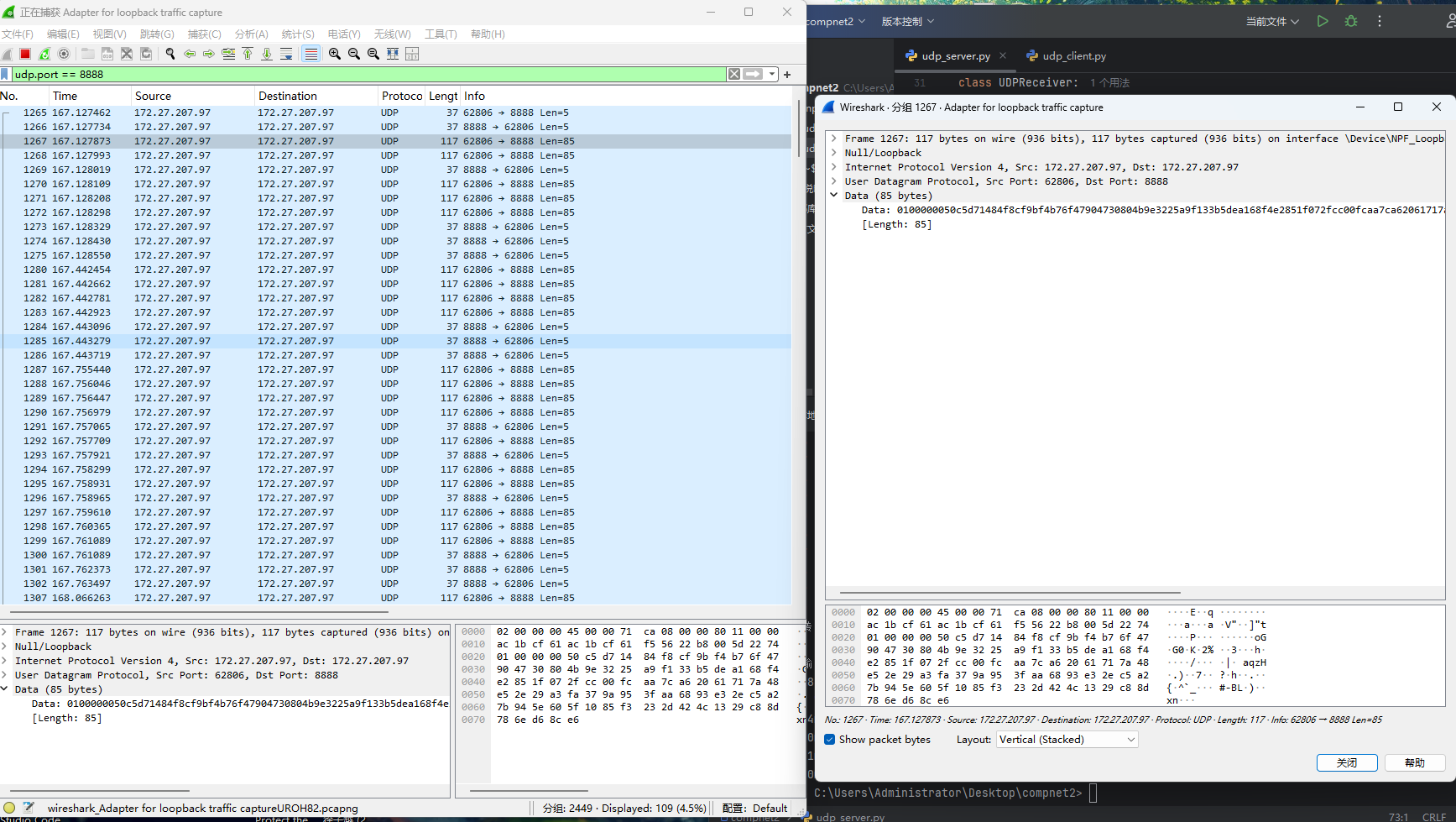
a.抓包截图如下



b. 实现上的关键点和对应的代码解决方案

关键点1: 滑动窗口机制实现

解决方案:

while len(self.ack\_received) < TOTAL\_TO\_SEND:

# 填充发送窗口

while (self.next\_to\_send < self.window\_start + WINDOW\_CAPACITY and

self.next\_to\_send < TOTAL\_TO\_SEND):

if self.next\_to\_send not in self.ack\_received:

# 发送数据包...

self.next\_to\_send += 1

self.process\_responses()

使用window\_start和next\_to\_send管理窗口边界

仅发送窗口内未确认的数据包

关键点2: 超时与重传处理

解决方案:

def process\_responses(self):

timeout\_point = time.time() + RETRY\_TIMEOUT

while time.time() < timeout\_point:

# 使用select等待

ready, \_, \_ = select.select([self.udp\_socket], [], [], max(0, time\_left))

if ready:

# 处理ACK...

# 内部超时检查

if time.time() >= timeout\_point:

self.handle\_retransmission()

Return

关键点3: 累积确认机制

解决方案:

if pkt\_type == ACK\_PACKET and ack\_num not in self.ack\_received:

self.ack\_received.add(ack\_num)

# 移动窗口到下一个未确认位置

while self.window\_start in self.ack\_received:

self.window\_start += 1

使用集合存储已确认包序号

移动窗口起始位置实现累积确认

关键点4: 连接建立握手

解决方案:

def establish\_connection(self):

syn\_header = struct.pack(PACKET\_HEADER, SYN\_PACKET, 0, 0)

self.udp\_socket.sendto(syn\_header, self.target)

# 等待ACK响应...

关键点5: 数据包构造

解决方案:

def create\_data\_packet(self, seq\_num):

header = struct.pack(PACKET\_HEADER, DATA\_PACKET, seq\_num, PAYLOAD\_SIZE)

payload = bytes([seq\_num % 256] \* PAYLOAD\_SIZE)

return header + payload

使用struct模块打包二进制头部

生成可预测的模拟负载数据

关键点6: 性能统计与分析

解决方案:

def generate\_report(self):

acked\_count = len(self.ack\_received)

loss\_percent = (1 - acked\_count / TOTAL\_TO\_SEND) \* 100

rtt\_series = pd.Series(self.rtt\_history)

# 计算统计指标...

记录RTT历史数据

使用pandas进行统计分析

c. 掌握的知识点

通过本项目熟悉并掌握的知识点:

可靠数据传输协议实现

深入理解GBN协议原理与实现细节

滑动窗口机制的编程实现

累积确认与超时重传策略

UDP协议高级应用

在UDP基础上实现可靠传输

自定义协议头部设计

连接建立与终止机制

网络编程核心技术

非阻塞socket编程

select多路复用技术

超时处理与重传机制

数据结构与算法

滑动窗口算法实现

序列号管理与窗口移动

高效数据结构应用（集合、字典）

性能分析与优化

RTT测量与统计

丢包率计算

使用pandas进行网络性能分析

调试与测试技术

网络协议调试方法

Wireshark抓包分析

边界条件测试（乱序、丢包、超时）

二进制数据处理

struct模块打包/解包二进制数据

协议头部设计

字节级数据处理

新掌握的知识点:

精确超时控制技术

绝对时间点超时检查

select与超时协同工作

高效重传触发机制

协议状态机实现

连接建立、数据传输、连接终止状态

基于事件的状态转换

错误恢复机制

网络性能度量方法

RTT统计分析方法

实际丢包率计算

重传次数统计

协议健壮性设计

处理乱序到达的ACK

处理重复ACK

处理部分ACK丢失情况

d.URL: