南开大学
Nankai University

南　　开　　大　　学

网 络 空 间 安 全 学 院

计算机网络实验报告

基于 UDP 服务设计可靠传输协议并编程实现

李潇逸 2111454

年级：2021 级

专业：信息安全、法学

指导教师：张建忠

2023 年 12 月 1 日

目录

一、 实验要求

在实验 3-1 的基础上，将停等机制改成基于滑动窗口的流量控制机制，发送窗口和接收窗口采用相同大小，支持累积确认，完成给定测试文件的传输。

二、 原理

(一) client 端

- 为了实现消息的同时收发，提高效率，使用多线程编程
- 如果 base<ackseq+1, 则说明 server 端对新消息进行了确认, 滑动窗口, 使 base=ackseq+1, 并重置计时器
- 如果 base>=ackseq,则说明 server 端或 client 端发送的消息发生了失序,由于 msgsend[0:base-1] 已经被确认，所以可以不对该项情况进行处理，直接跳过等待接收下一条消息即可
- 滑动窗口机制需要在发送前预先将窗口内的消息存储至内存中
- 如果窗口未满且当前状态不需要重传，则继续发送信息，发送完毕后重置计时器
- 如果超时未收到对方发来的所需的 ACK，则按超时处理，重新发送窗口中已发送但是未确认的消息
- 如果出现重发一定次数依然无法收到所需 ACK 的情况，进行断网处理
- 当所有消息都被对方确认即 base==buffersize（发送缓冲区），结束当前发送线程

(二) server 端

- 按顺序接收对方发来的消息 msgseq（收到的消息序号）==recvnextseq，且校验和正确，返回对应的 ACK（ackseq=msgseq）
- 如果发生消息失序,即 msgseq!=recvnextseq,或校验和错误,则丢弃消息,返回第 recvnextseq-1 条消息的 ACK

三、 代码分析

发送线程

接收线程

```
int sendthread() {
    int sendcase = 0;
    message msg;
    while (!sendcase) {
        for (int i = sendbase; i <= sendtop; i++) {
            if (state[i] == 0) {
                state[i] = -1;
                msg.seq = i;
                sendOneMsg(msg, i);
```

```
                sendcase = ((i == (messagenum - 1)) ? (sendcase + 1) :
                    sendcase);
            }
        }
    }
    ExitThread(TRUE);
}
```

接收线程

接收线程

```
int recvthread() {
    while (1) {
        message msg = recvmessage();
        while (!msg.isEXT()) {
            continue;
        }
        if (msg.isACK()) {
            time_t now_time = time(NULL);
            tm* t_tm = localtime(&now_time);
            cout << "收到ack为" << msg.ack << "的数据包" << endl;
            cout << asctime(t_tm) << endl;
            state[msg.ack] = 1;
            cout << endl;
            if (msg.ack == messagenum - 1) {
                ExitThread(TRUE);
            }
        }
    }
}
```

窗口滑动

窗口滑动

```
while (!sendcase) {
    while (state[sendbase] == 1) {
        // 收到当前滑动窗口底的数据包ack
        cout << "滑动窗口前移一位" << endl;
        if (sendbase == messagenum - 1) {
            sendcase += 1;
            break;
        }
        sendbase++;
        sendtop = ((messagenum - sendtop - 1) ? (sendtop + 1) : sendtop);
        cout << "现在窗口底部是:" << sendbase << ", 窗口顶部是:" << sendtop
            << endl;
    }
```

四、 展示

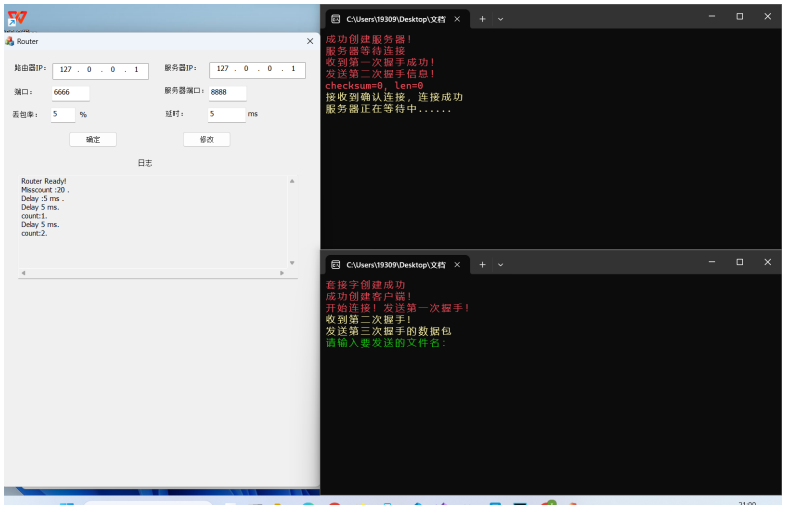

图 1: 初始设置

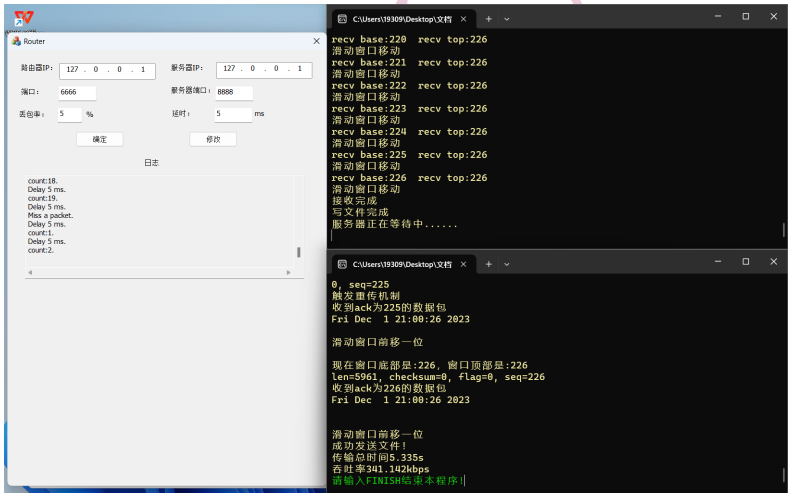

图 2: 传输过程

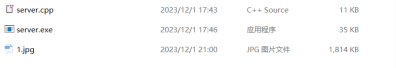

图 3: 结果

五、 总结

实现了 UDP 传输滑动窗口，将研究如何加快传输速度