**实验3-2 基于UDP服务设计可靠传输协议并编程实现**

1813069 郭怡霏

1. 实验目的：

利用数据报套接字在用户空间实现面向连接的可靠数据传输，功能包括：

1. 建立连接；
2. 差错检测；
3. 确认重传；
4. 固定窗口大小；
5. 累积确认。

采用滑动窗口机制，完成给定txt格式和jpg格式测试文件的单向传输（与TCP功能类似，在UDP服务上实现）。

1. 实验原理：

滑动窗口协议（Sliding Window Protocol），属于TCP协议的一种应用，用于网络数据传输时的流量控制，以避免拥塞的发生。该协议允许发送方在停止并等待确认前发送多个数据分组。由于发送方不必每发一个分组就停下来等待确认。因此该协议可以加速数据的传输，提高网络吞吐量。

由于停止等待协议效率太低，因此有了GBN协议，这也是滑动窗口协议真正的用处，这里发送的窗口大小为n，接收方的窗口仍然为1。发送方可以发送窗口内的所有n个数据包，发送后监听收到的ack，如果窗口前k个数据包已成功接收，则窗口向后滑动k位；接收方每次接收到的数据包如果不符合需求编号就丢弃，发送的ack为编号i，表示i号及之前的数据包都已经成功接收。后退n协议的好处无疑是提高了效率，但是一旦网络情况糟糕，则会导致大量数据重发，反而不如停等协议。

本次实验实现了停等的机制和滑动窗口的数据包流动。

1. 实验重点：
2. 固定滑动窗口大小；
3. 累积确认；
4. 应用层的自定义协议。
5. 实验环境：
6. 操作系统：Oracle VM VirtualBox 6.1.14 r140239 (Qt 5.6.2)

Ubuntu 18.04.5 LTS (64 bit)

1. IDE：Visual Studio Code 1.51.1
2. 使用语言：C++
3. 使用库：stdio.h string.h stdlib.h unistd.h sys/types.h sys/socket.h sys/time.h netinet/in.h time.h pthread.h arpa/inet.h iostream fstream cstdlib ctime
4. 实验步骤：

自定义协议：

1. 建立连接：同实验3-1；
2. 传输数据类型：同实验3-1；
3. 读写方式：同实验3-1；
4. 文件结束：若文件结束，检查窗口内是否还有没有被成功接收的数据包，如果还有则继续发送数据包直至窗口全部清空，否则与实验3-1设置相同的最后id（-1），发送完退出；
5. 差错检测：同实验3-1；
6. 确认重传：server端设置一个链表，装载了四个数据包，使用停等机制，将收到的ack编号对应数据包的used位置1。设置一个while(1)的循环，每一次循环中发送链表中的第一个数据包（更新校验和后发送），收到一个ack后检查是否需要滑动窗口；
7. 滑动窗口大小：为4；
8. 结束：同实验3-1。

编译：

在 vscode 命令行打开当前文件夹，输入如下代码：



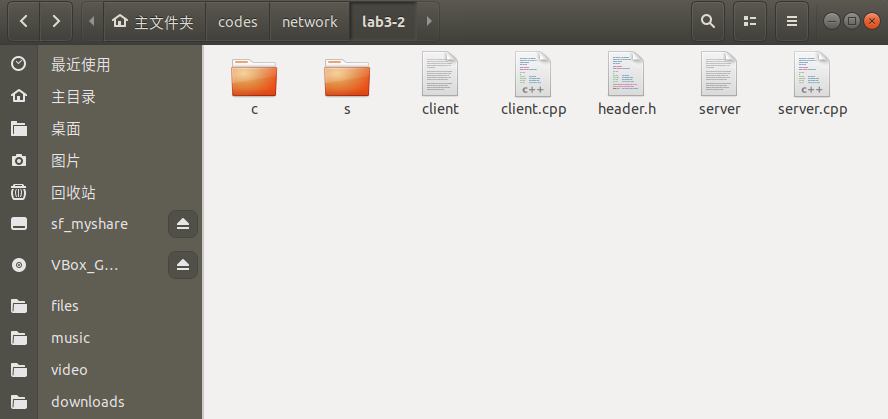
运行：

在命令行打开当前文件夹，输入如下代码，即可运行：

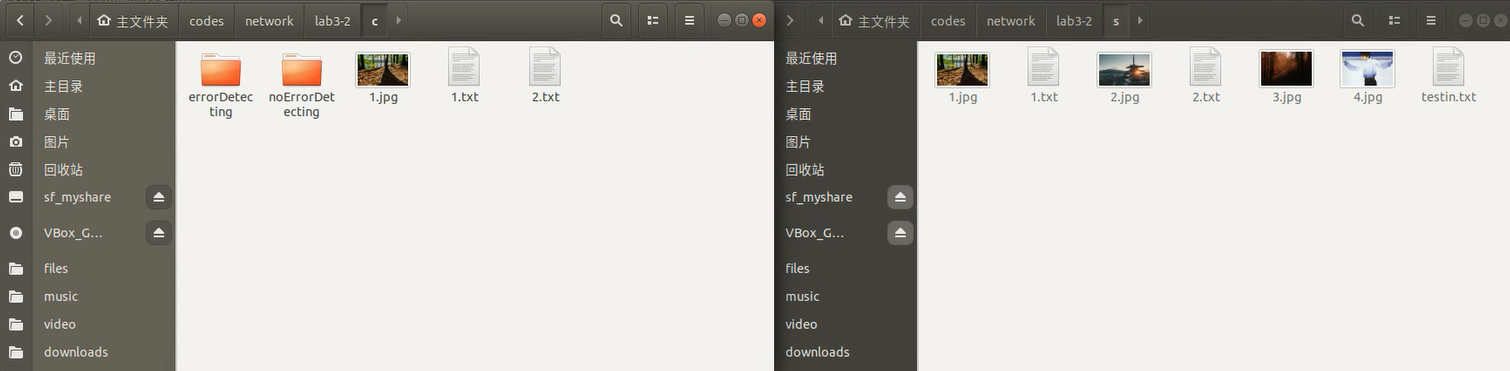




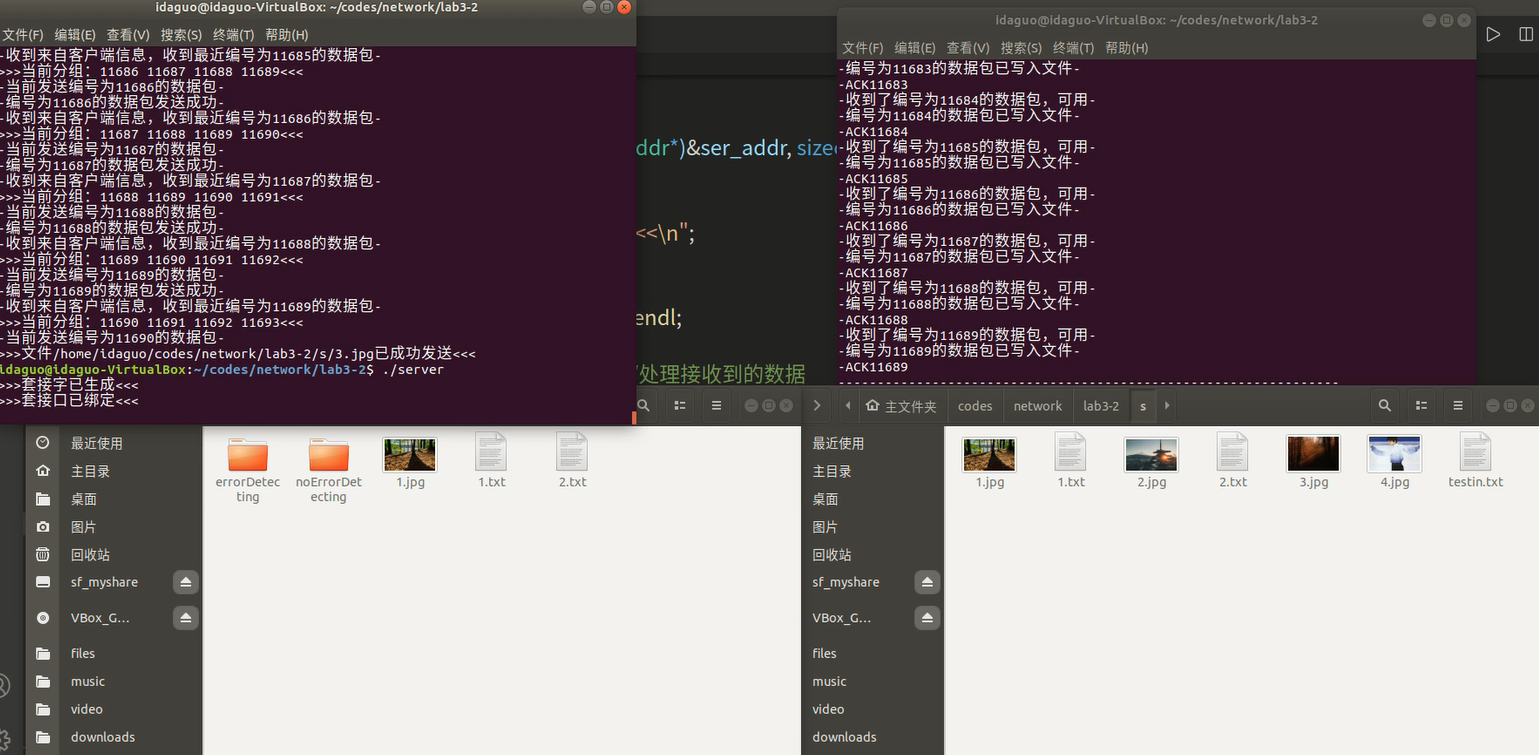
1. 实验结果：



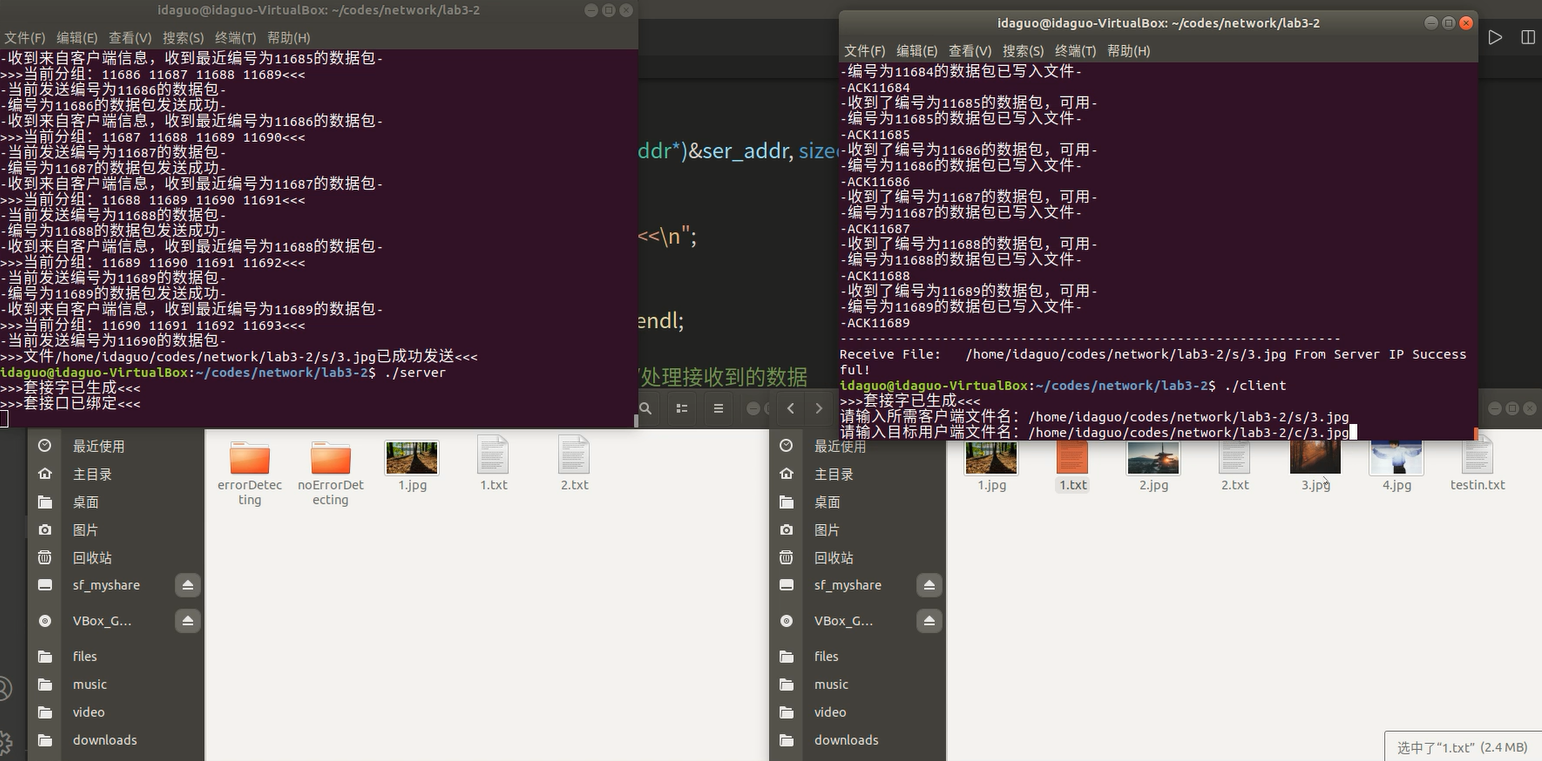
以上是文件列表，分别是客户端文件夹、服务器文件夹、客户端可执行文件、客户端源代码、服务器可执行文件、服务器源代码。通过在命令行输入./xxx（xxx 为可执行文件名）来运行。



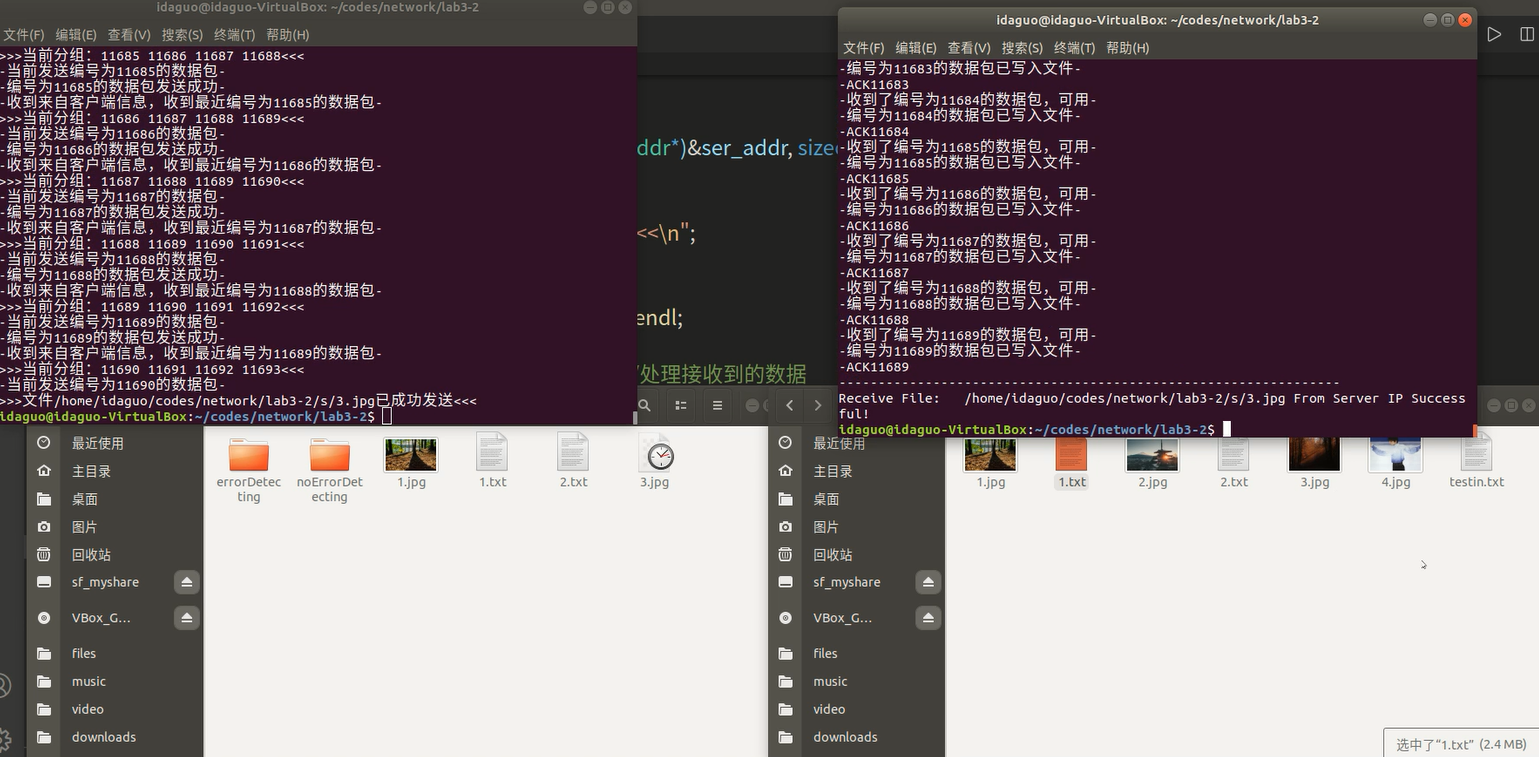
传输文件前的client文件夹和server文件夹



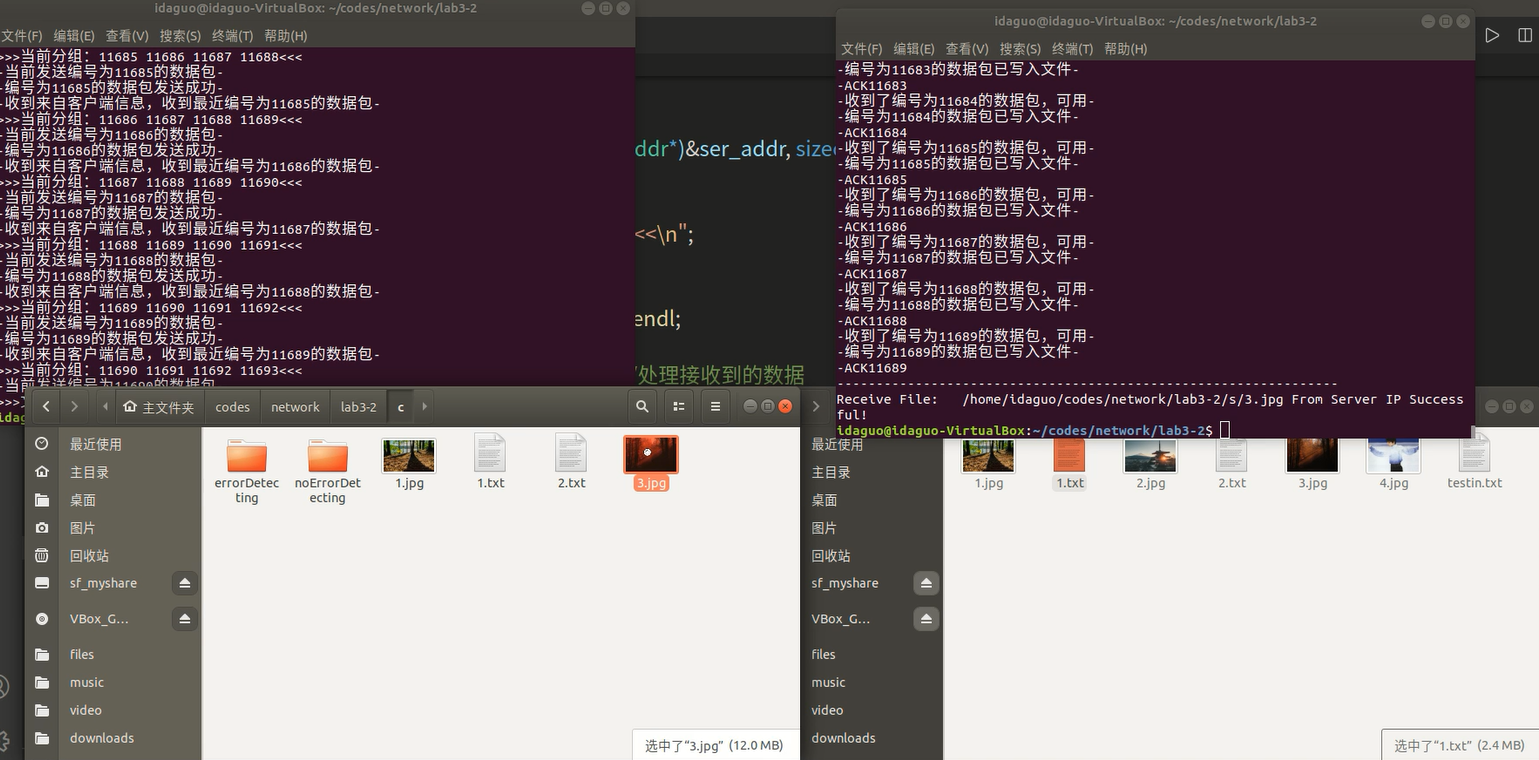
（1）服务器完成初始化设置开始监听



（2）客户端初始化，由用户输入希望得到的文件和要保存的位置



（3）名为3.jpg的文件传输结束，服务器和客户端都退出



（4）传输结束，client文件夹有了被传过来的3.jpg文件，可正常显示

1. 问题与不足：

设置为链表不是很有必要，因为我滑动窗口时也是将后一个节点的值直接复制到前一个节点上，没有充分利用链表的优良性能，此时类的构造析构其实也是时间开支，因此使用数组可能更好，下次注意。