Introduction to Computer Network

Lab 3: Linux Socket Programming II

Department: CS

Student's Id: 110062271

Name: 林奕為

Contents

Abstract

This project implements the Linux Socket Programming in both server and client side, and the client should be able to download file from the server using the mechanism of a UDP socket.

The main objective is to have hands-on experience in implementing automatic repeat request (ARQ).

This code meets all the requirements on the specification pdf that it can be successfully compiled by make command and executed on the Linux VM.

For the validation, I use Linux command on the spec pdf:

```
$ cmp -s video.mp4 download_video.mp4 && echo "Same!" || echo
"Different!" Same!
$ sudo apt update && sudo apt install colordiff
$ colordiff -y <(xxd video.mp4) <(xxd download_video.mp4)
```

and I get the “Same!” displayed on the kernel of the virtual machine meaning that the downloaded file from the code of this project is the same as the original mp4 file.

Additionally, I didn’t implement the bonus part of selective repeat mechanism.

Requirements:

Can be compiled by make	Yes
Successfully executed	Yes
Implementation of stop and wait	Yes
Downloaded file correct	Yes
Selective Repeat	No

What I have learned from the Lab

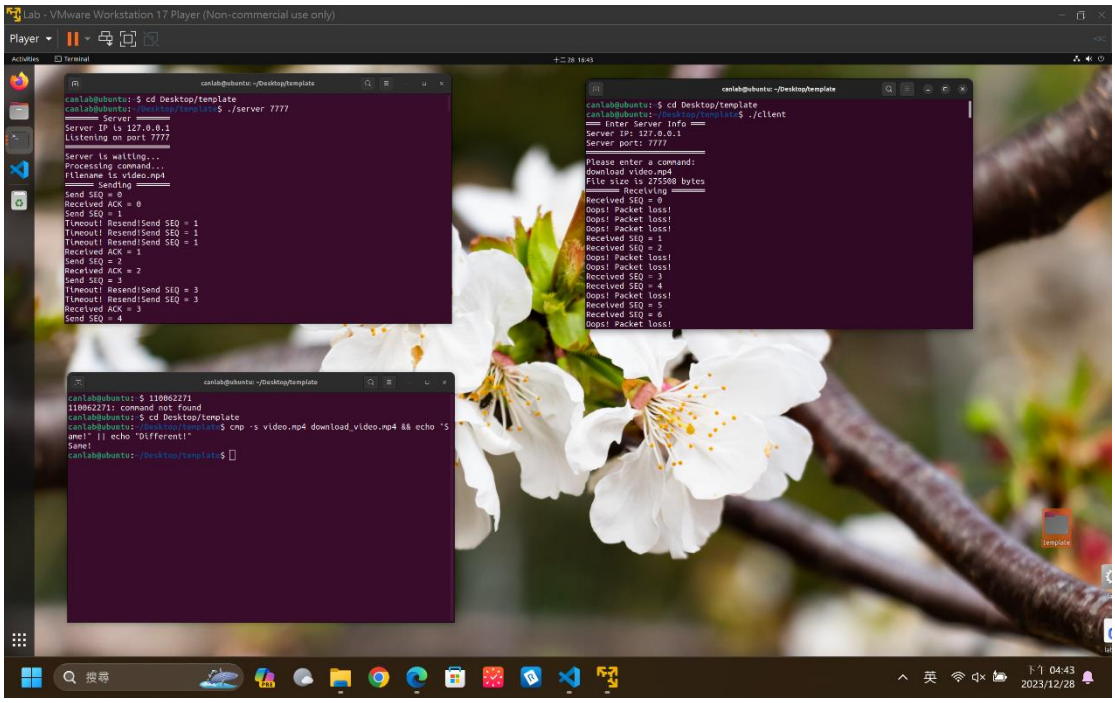
In this lab, I didn't spend as much time as the previous lab since I have become more familiar to Linux socket programming, and it's also because the course has provided me with template code and appropriate command to follow on.

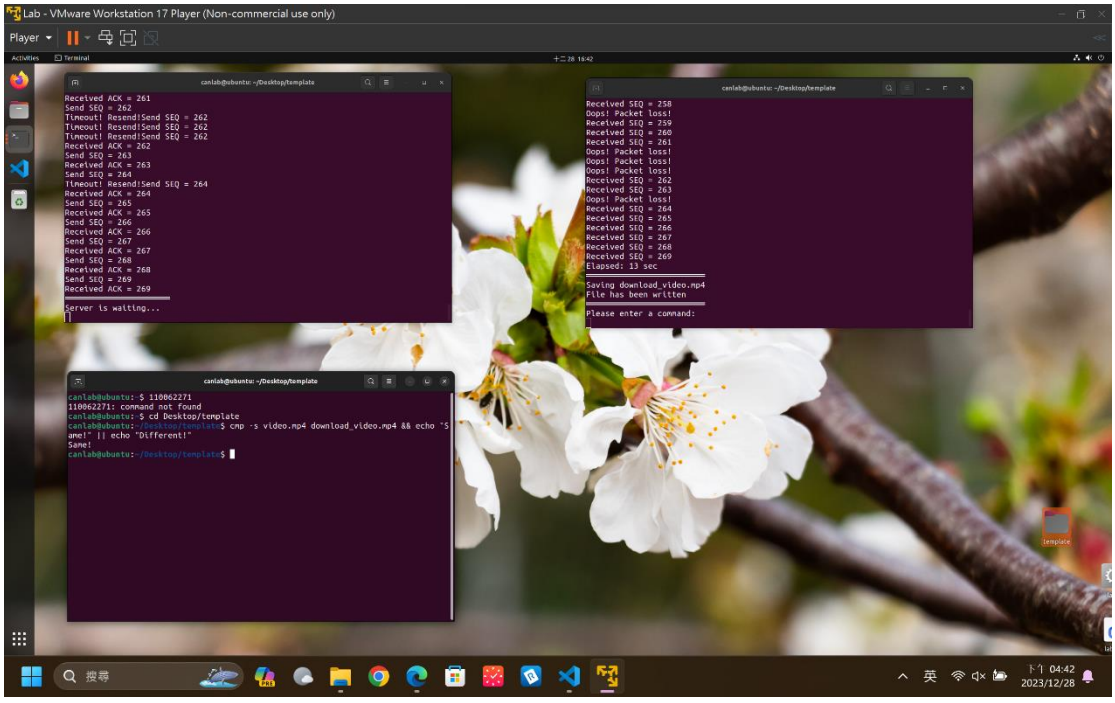
I have encountered two difficulties in implementing, one is the poll() function in the server.c, which I solved it by looking it up in stackoverflow.

The second one being the download file is completely different from the original one after executing cmp command on the VM though I successfully receive and send correct numbers of packets. After closer inspection, I found that I wrongly increment bytesRead variable in server.c by the size of the whole packet structure and not the packet.header.size, this results to copy the wrong location of the buffer array when calling memcpy function, after correcting error the execution result is corrected.

In this lab, I have a hands-on experience as to implementing ARQ as described on the pdf, and I also get more familiar with the Linux socket programming, also the familiarity with FILE read/write operations under C library, which I less frequently used in CS daily coursework.

Execution Result





Reference:

C 库函数 – fread() | 菜鸟教程 (runoob.com)
For file read/write and open/close operations under C library.

sockets - Using poll() for a TCP server in C - Stack Overflow
For the understanding of poll() function

memcpy() in C/C++ - GeeksforGeeks
Refresh the usage of memcpy() in C library