計算機網路概論 – Lab1 104062223 鄭凱文

此FTP能模擬的功能與特色有以下：

1. 上傳檔案
2. 下載檔案
3. 列出server端資料夾的檔案資訊
4. 重新命名檔案(Extra)
5. 附有Welcome資訊，每次執行輸入指令時都會顯示適當、client和server各自不同的提示資訊，包含進度、執行成功與否，並精心調整文字排版

程式實作部分：

**有利用到的特殊header file：**

<stdio.h> -> rename function

<sys/types.h> ->包含可能會用到的相關檔案型別

<winsock2.h> ->socket相關函式庫

<dirent.h> ->directory相關函式庫

<unistd.h> ->一些關於檔案讀寫的函式庫

<sys/stat.h> ->取得檔案資訊的函式庫

<time.h> ->將檔案時間轉換成localtime

**Server端：**

共包含七個function，並限制單一socket最大容量為1024 byte。

**int main (int argc, char \*\*argv)**

開啟server前的前置作業，首先會利用WSAStartup啟用winsock，並確認版本有沒有符合，接著利用serverSocket = socket(PF\_INET, SOCK\_STREAM, 0) 創建serverSocket，並將server的IP address設為INADDR\_ANY(預設為127.0.0.1)，port number讀取argv[1] (在run cmd時必須輸入port number)。接著利用bind將serverSocket和serverAddress建好聯繫，並執行listen。以上若都成功的話，則會跑出server端的歡迎資訊，然後等著client發出連線申請，利用accept接收，成功的話會顯示client端的IP，並進入**Connection\_handler** function

**void Connection\_handler (SOCKET sockfd)**

此為主宰連線的main function，透過recv(sockfd, command, MAX\_SIZE, 0)接收來自client的socket，利用’command’內的字串來判斷client送出何種指令，分別為”exit”, “dir”, “get”, “put” 和 “rename”，並各自進入所負責的function，若接收到非上述五種字串，則輸出 ”Invalid Command”。

**void Welcome\_handler (SOCKET sockfd)**

無特別功能，僅是在進入Connection\_handler時server傳出歡迎資訊，用來模擬真實的FTP連線情況，以及作為第一次利用send function，利用socket來傳送資訊的測試。

**void File\_listing\_handler (SOCKET sockfd)**

列出./upload資料夾內的檔案資訊，當command為”dir”時進入。首先利用DIR\* pDir紀錄 ./upload的資料夾位置( pDir = opendir("./upload") )，接著透過struct dirent\* pDirent = readdir(pDir) 進行while迴圈，不斷的讀取 ./upload下的所有檔案(為求簡單化略過上層和下層的資料夾不顯示)，並利用socket傳回檔名、檔案大小、最後修改時間。

取得檔案大小和修改時間，這裡要透過stat function，先將檔案的完整路徑存在buf，作為參數傳給stat便可得到檔案的眾多資訊 (此處只取用file size & st\_mtime最後修改時間)，又因為st\_mtime預設為從1970年算到現在的”秒數”，不符合現代人需求，再透過localtime & strftime將其轉換成我們熟悉的時間表示法。

**void File\_sending\_handler (SOCKET sockfd, char filename[])**

將檔案以socket的形式傳給client，當command為”get”時進入。首先利用FILE \*fp = fopen(path, "rb")(path為檔案完整路徑)，來開啟目標檔案，接著透過

fseek(fp, 0, SEEK\_END) file\_size = ftell(fp) rewind(fp)

等步驟取得檔案存取位置、file size並傳回給client作為接收用。

傳送這邊因為限制MAX\_SIZE = 1024 byte，我們利用一個while迴圈將檔案切成一次不大於1024的packet，經由send socket批次傳送給client，透過變數write\_sum來記錄檔案傳送完畢沒，讀取檔案則是利用fread。等到傳送完成，藉由sleep的小停頓來模擬真實的FTP，最後再傳回成功資訊。如果找不到檔案，則顯示錯誤資訊，並傳出能代表錯誤資訊”E\_R\_R\_O\_R”的socket給client。

**void File\_receving\_handler (SOCKET sockfd, char filename[])**

接受以socket形式來自client的檔案，當command為”put”時進入。同樣利用FILE \*fp = fopen(path, "wb")開啟”將來”的檔案位置(path為之後寫入的地方)，接著等待client傳來的資訊，若buf為”E\_R\_R\_O\_R”即代表client找不到此檔案，無法上傳。如果接收到的是歡迎資訊，則接著接收file size。跟下載一樣一次只能接收1024 byte，透過變數read\_sum檔案接收完畢沒，寫入檔案則是利用fwrite。最後當檔案接收完成，會顯示file size。

**void File\_Rename (SOCKET sockfd, char oldname[], char newname[]) (EXTRA)**

額外功能，將目標檔案改成新檔名，當command為”rename”時進入。開啟資料夾和檔案的方式和File\_listing\_handler function一樣，只是尋找時利用strcmp(pDirent->d\_name, oldname) ==0 ?判斷是否為目標檔案，若有找到則透過rename function，傳入舊檔案位置和新檔案位置(含檔名)來進行更名動作，如果回傳值為1代表更名成功，回傳成功資訊給client。假使最後沒有找到此檔案，則回傳錯誤訊息給client。

**Client端：**

共包含四個function，並限制單一socket最大容量為1024 byte。

**int main (int argc, char \*\*argv)**

與server雷同，先建立clientSocket，然後將serverAddress設為argv[1]、port number設為argv[2](在run cmd時必須輸入兩個參數)，接著inet\_pton會測試是否為有效IP，通過後利用connect function向server送出連線申請。

**void Connection\_handler(SOCKET sockfd)**

此為主宰連線的function，一開始會透過recv接受來自server端的歡迎資訊，接著會提示使用者"Enter The Command: "，合法的指令有”exit”, “dir”, “get”, “put” 和 “rename”，其餘皆視為Invalid command不予理會。因為dir & rename兩種指令client屬於被動等待server回應，並不另外設function處理，在dir時會依序接收server傳來的檔案資訊，做好排版後輸出在螢幕上，直到接收到特殊的訊息”o\_v\_e\_r”，代表所有的檔案資訊皆取得。Rename方面則是單純的送出兩次socket，分別為oldname & newname，之後等待server回應。

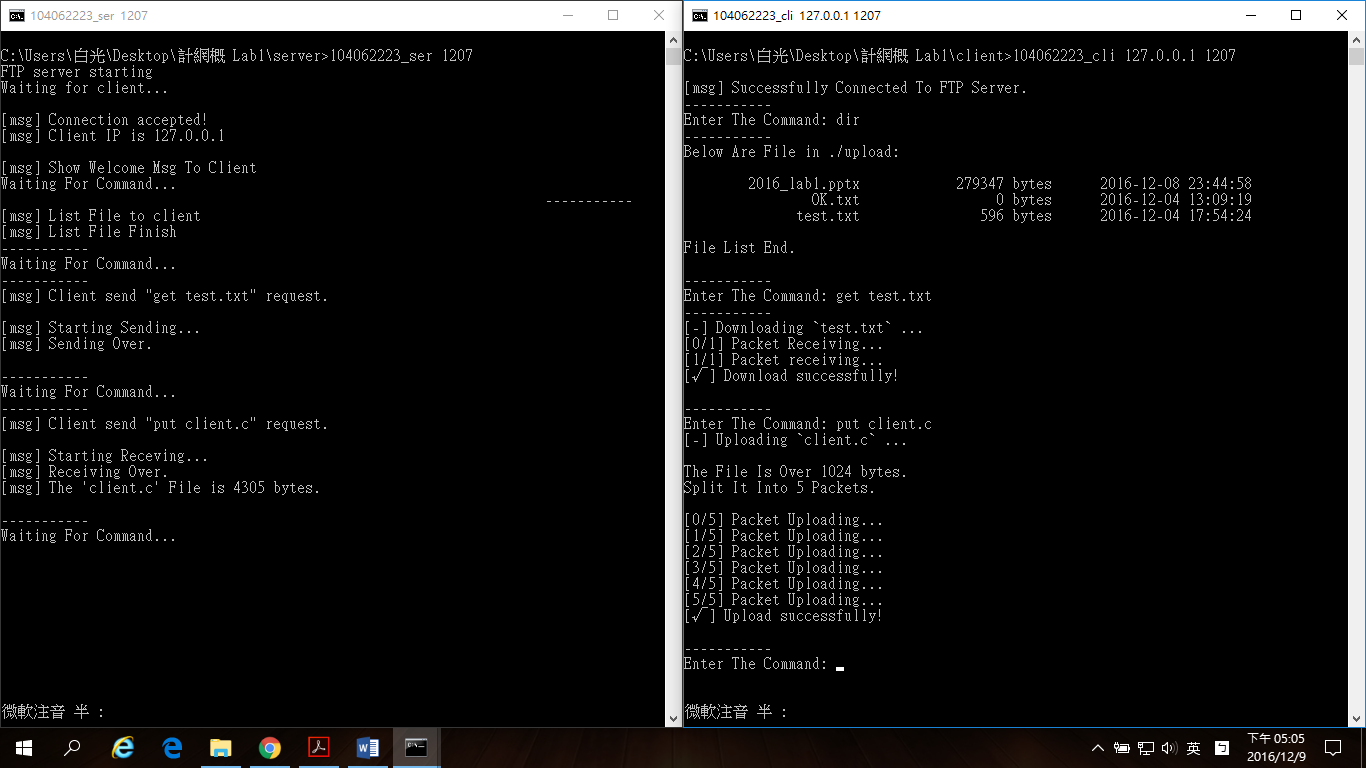
**void File\_download\_handler (SOCKET sockfd, char filename[])**

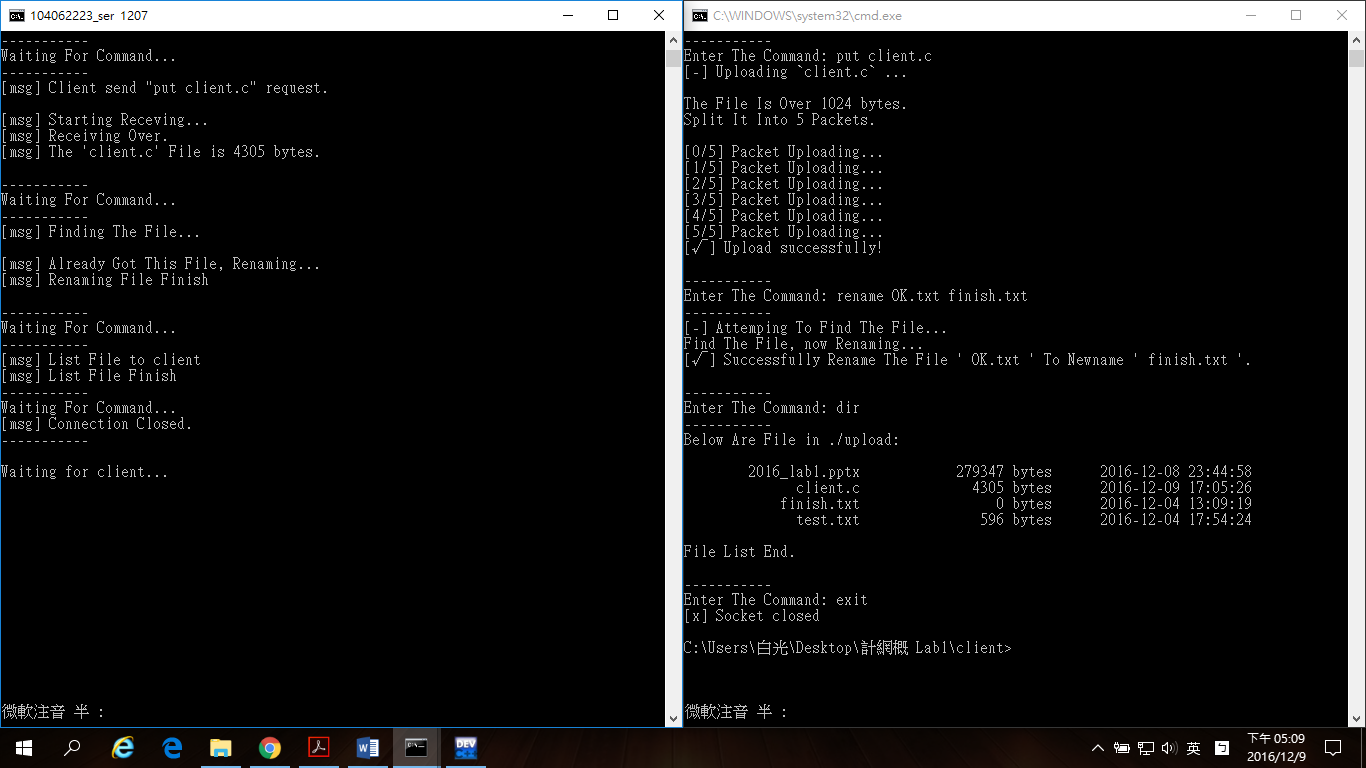
從server下載檔名為filename的檔案，當command為”get”時進入。基本上功能和server的File\_receving\_handler一模一樣，這裡不贅述相同的功能，不一樣的主要是顯示資訊，client會依據file size，算出總共會切成幾個packet，在接受時經由已接收的byte size來顯示[?/total num of pkts] Packet Receiving...，告訴使用者接收的進度。

**void File\_upload\_handler (SOCKET sockfd, char filename[])**

上傳檔名為filename的檔案，當command為”put”時進入。基本上功能和server的File\_sending\_handler也是一樣，這裡不贅述相同的功能，另外同download，client端會計算總共要切成幾個packet來上傳，並透過已傳送的byte size來顯示[?/total num of pkt] Packet Uploading...，告訴使用者上傳的進度。因為這裡client為主導方，成功和失敗的資訊由自己判斷後print out，而非由server send socket得知。

Screenshot：





所遇到的問題與心得：

問題最多的大概就是如何找到合適的function，過去寫程式很少接觸到File讀取、寫入的部分，像是fread, fwrite對我而言都極為陌生，更別提關於dirent.h和stat.h這兩個根本沒聽過的header檔(基本上除了winsock2.h的header檔都是Google相關功能而學來的)，像是dir的部分，光是找到Dir這個型別就花了不少時間，而後續的stat、localtime轉換又是另一段掙扎。

但這些function在實作真正能應用的程式上，想必都是相當重要且基本的函式，雖然因為時間的問題，大部分的函式我還是僅停留在「會用」，對實際的原理一知半解的情形，但這次的lab對於socket的使用方式，以及實際File相關操作的程式方面，確實收穫不少。