

Final Project of Computer Networking

I. Implementation

(i) Server-side

Server-side 要做的事情是與 Client 連接，然後送出资訊告訴 Client 可以執行哪些動作，最後接收 Client 傳過來的資訊並做處理。

在連接方面，我的實作與範例 code 有所不同，我將 accept function 放於 While-loop 之前，並把 While-loop 內的 closesocket function 移至 loop 之後，讓 Client 與 Server 不需要在每次傳遞封包後就重新連接。

在判斷、資料傳送與資料儲存方面，我總共使用了 4 個 char array，分別是 menu[MAX_SIZE], all_message[MAX_SIZE], send_buf[MAX_SIZE], rec_buf[MAX_SIZE]。send_buf 是用來記錄要傳送至 Client 的訊息，並使用 send function 將資料 send_buf 內容傳至 Client-side（使用方式如下圖）。

```
send_buf[0] = '\0';
strcat(send_buf, "All messages:\n");
strcat(send_buf, all_message);
strcat(send_buf, menu);
send(clientSocket, send_buf, sizeof(send_buf), 0);
```

rec_buf 則是用來接收 Client 傳過來的訊息。並可以使用 strncmp 去判斷要執行哪一個動作（使用方式如下圖）。

```
recv(clientSocket, rec_buf, sizeof(rec_buf), 0);
if(!strncmp(rec_buf, "1", 1)){ //if choose 1 in menu -> show all message.
```

menu 的內容是固定的，它會存著 menu 訊息，再藉由 send_buf 將 menu 資訊傳至 Client（menu 的內容與使用如下圖）。

```
char menu[MAX_SIZE] = "\n\n---Menu---\n\n1. Read all messages.\n2. Write a new message.\nPlease type \"1\" or \"2\" to select an option.\n";
```

最後一個 all_message 是用來記錄曾經輸入過的 message。在 menu 選擇 option 2 之後，可以在 Client-side 輸入訊息，Server-side 會將接收的訊息放入 all_message（如下圖）。

```
// Send menu to client
send(clientSocket, menu, strlen(menu), 0);
```

如果選擇 option 1，會將 all_message 的內容接於 send_buf，並於之後傳至 Client-side（如下圖）。

```
recv(clientSocket, rec_buf, sizeof(rec_buf), 0);  
  
strncat(all_message, rec_buf, strlen(rec_buf));  
strcat(send_buf, "All messages:\n");  
strcat(send_buf, all_message);
```

(ii) Client-side

Client-side 這邊是負責主要的操控。在連接 Server 後，會將 Server 傳過來的資訊顯示於螢幕上，方便使用者知道當下應該要做什麼操作，並在完成操作後將資料傳至 Server 端。

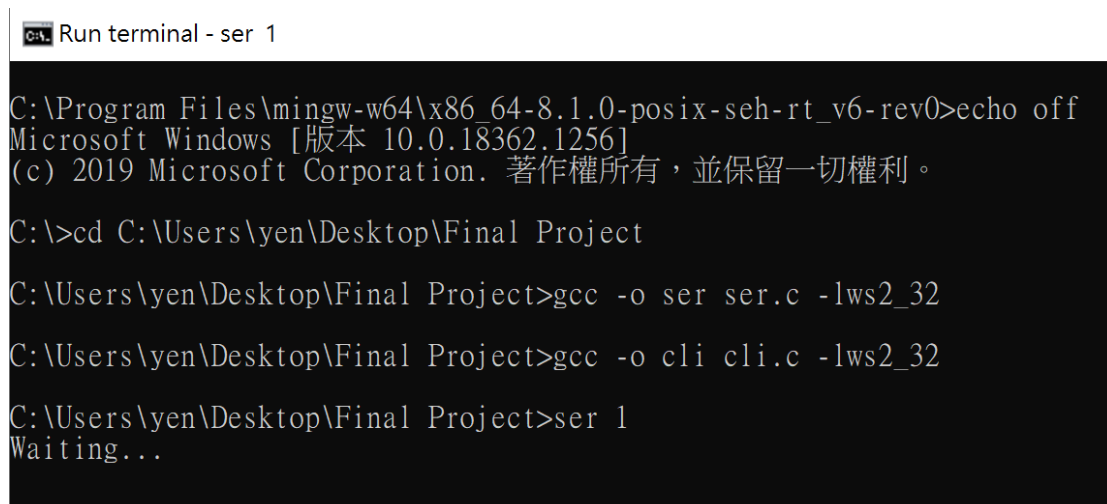
與 Server-side 相同，我將連接用的 accept function 放在 While-loop 之前讓兩端先做連接。

在 While-loop 裡，Client-side 要做的事情只有印出 Server 傳過來的訊息與發送訊息給 Server（如下圖）。

```
while (1) {  
    recv(serverSocket, rec_buf, sizeof(rec_buf), 0);  
    printf("%s\n", rec_buf);  
  
    fflush(stdin);  
    scanf("%[^\\n]", send_buf);  
    // Send the data to the server, and receive it back  
    send(serverSocket, send_buf, strlen(send_buf), 0);  
}
```

II. Screenshot and explanations

下圖為 Server-side 啟動後第一步，還沒有與 Client 連接，維持在 waiting。此時只要 Client 端發送連接要求，兩邊就可以開始連線。



```
Run terminal - ser 1

C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0>echo off
Microsoft Windows [版本 10.0.18362.1256]
(c) 2019 Microsoft Corporation. 著作權所有，並保留一切權利。

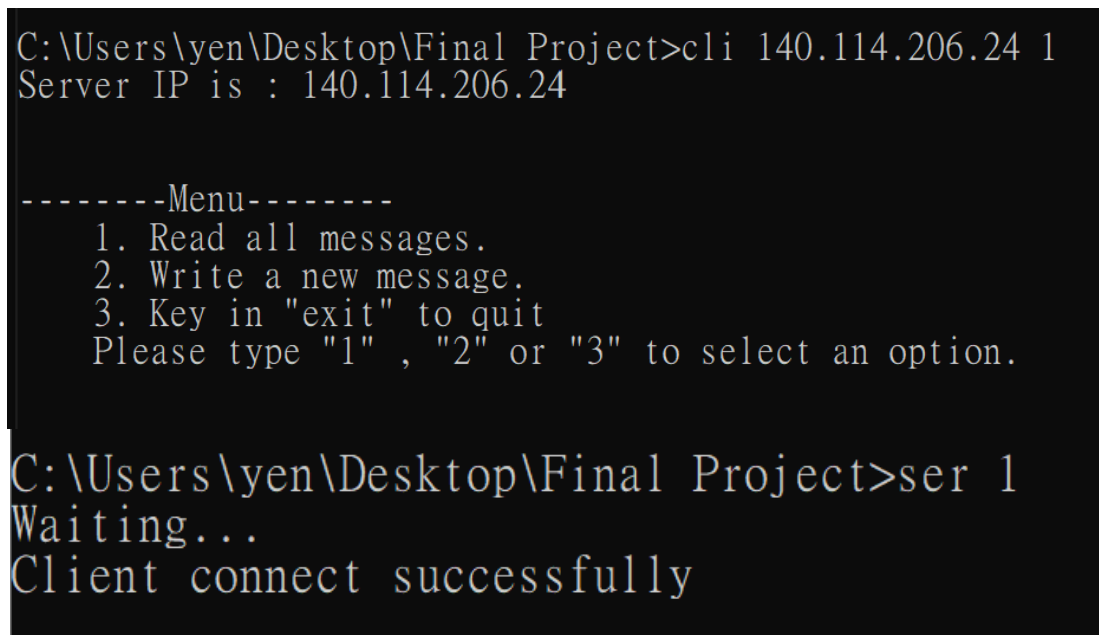
C:\>cd C:\Users\yen\Desktop\Final Project

C:\Users\yen\Desktop\Final Project>gcc -o ser ser.c -lws2_32

C:\Users\yen\Desktop\Final Project>gcc -o cli cli.c -lws2_32

C:\Users\yen\Desktop\Final Project>ser 1
Waiting...
```

接著 client-side 就送出連接的請求。在連接上之後 Client 的介面會出現從 Server 送過來的 menu，而 Server 介面則顯示 Connect Successfully。



```
C:\Users\yen\Desktop\Final Project>cli 140.114.206.24 1
Server IP is : 140.114.206.24

-----Menu-----
1. Read all messages.
2. Write a new message.
3. Key in "exit" to quit
Please type "1" , "2" or "3" to select an option.

C:\Users\yen\Desktop\Final Project>ser 1
Waiting...
Client connect successfully
```

接著測試各項功能是否正常，如果在 menu 輸入'1'就會依序練出目前存在的 message，並以 a,b,c...排序。如果在 menu 輸入'2'會請 Client 輸入 new message。如果輸入'exit'雙方就會斷開連接（下列依序列出實際測試狀況）。

```
2
Type a new message:

This is a test.

-----Menu-----
  1. Read all messages.
  2. Write a new message.
  3. Key in "exit" to quit
Please type "1" , "2" or "3" to select an option.
```

```
2
Type a new message:

Test message no.2

-----Menu-----
  1. Read all messages.
  2. Write a new message.
  3. Key in "exit" to quit
Please type "1" , "2" or "3" to select an option.
```

```
1
All messages:
a. This is a test.
b. Test message no.2

-----Menu-----
  1. Read all messages.
  2. Write a new message.
  3. Key in "exit" to quit
Please type "1" , "2" or "3" to select an option.
```

```
exit
Disconnected
```

```
C:\Users\yen\Desktop\Final Project>
```

III. Difficulties and solutions

這份作業遇到的最大問題是電腦環境的適應。因為平常使用的是 mac，在使用 windows 編譯很不習慣，而且在 windows 的 command line 中，沒有辦法使用 gcc compile 檔案。為了解決問題，上網找了許多插件像是 mingw 之類，在耗費多時後，終於成功在 windows 上操作。雖然環境不熟悉，但我認為身為一個資工系學生，快速適應不同電腦環境以及 command line 的使用是必備的技能，這次能有這樣的收穫十分可貴。

另外，這次用上了 winsocket2 這個函示庫。雖然大部分的 code 老師都已經提供，但畢竟以前沒有使用過，在一開始要看懂範例 code 時也花費了些許時間，但因為網路上資料庫內容充足且詳細，在之後的使用及改寫非常的順利。