**Socket Programming作業格式**

資工三乙- 406262163黃品翰

## 環境

- 使用Virtual Box 6.0版本

- 虛擬機網路利用橋接介面卡(bridge)連接

- Server環境

- Ubuntu 16.04 server版

- ipv4: 192.168.43.18

- Client環境

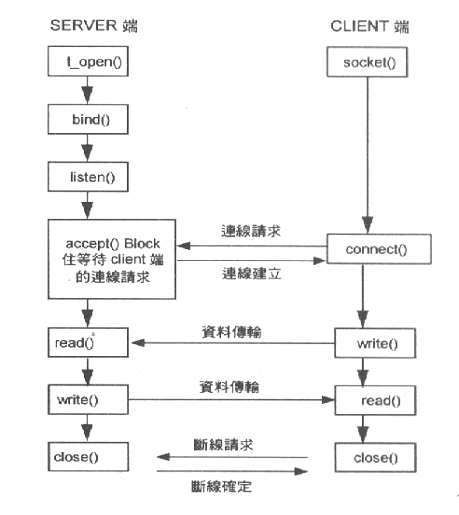
- Kali 2018.04

- ipv4: 192.168.43.26

- Ubuntu 18.04

- ipv4: 192.168.43.140

## 架構



### 題目

1. Client/Srever程式(1個Client對1個Server)

- 說明

- 右側為Ubuntu16.04當作server(使用ssh連線進虛擬機)

- server ip: 192.168.43.18/port 4444

- 左側為Ubuntu18.04當作client(本機terminal)

- client ip: 192.168.43.140/port 4444

- test.txt(server)內容, get.txt(client拿的)內容

- Here is the Flag La~~~~~~~~~~~~~~

- 過程

- client向server請求資料

- server去本機拿test.txt進行串流

- client開檔(get.txt)準備接收

- accept後開始進行傳輸

- 傳完之後直接將結果print出

### 示意圖

* 請洽client\_1.gif

2. Client/Srever程式(多個Client對1個Server)(30%)

- 說明

- 右側為Ubuntu16.04當作server(使用ssh連線進虛擬機)

- server ip: 192.168.43.18/port 4444

- 左上(client)與上題一樣，當作第一台client

- client1 ip: 192.168.43.140/port4444

- 左下為Kali 2018.04當作第二台client

- client2 ip: 192.168.43.26/port 4444

- test.txt(server)內容, get.txt(ubuntu16.04/client1), get2.txt(kali/client2拿的)內容

- Here is the Flag La~~~~~~~~~~~~~~

- 過程

- 過程與前題相同

- 兩邊交互請求時仍然可以回傳

- 傳完之後直接將結果print出

### 示意圖

* 請洽client\_1\_2.gif

## Code

### Server\_1.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <sys/types.h>
6. #include <sys/socket.h>
7. #include <netinet/in.h>
9. #define SEND\_FILE "test.txt"

12. //socket的server,client連線
13. **struct** sockaddr\_in serverInfo,clientInfo;
15. //inputBuffer 接收資料
16. //回傳給client端
17. **char** inputBuffer[256] = {};
18. **char** message[] = {"Good!!! Send success~\n"};
20. **void** setting\_server(**int** sockfd)
21. {
22. /\*
23. serverInfo server資訊
24. clientInfo client資訊
25. addrlen client資訊長度
26. bzero 初始化，將struct(server)涵蓋的bits設為0
27. bind() 把自家地址綁到socket上
28. listen(sockfd,x) 規定上線人數x
29. \*/
30. bzero(&serverInfo,**sizeof**(serverInfo));
31. serverInfo.sin\_family = PF\_INET;
32. serverInfo.sin\_addr.s\_addr = INADDR\_ANY;
33. serverInfo.sin\_port = htons(4444);
34. bind(sockfd,(**struct** sockaddr \*)&serverInfo,**sizeof**(serverInfo));
35. listen(sockfd,5);
36. }
38. **void** recv\_msg(**int** sockfd)
39. {
40. **int** forClientSockfd = 0;
41. **int** addrlen = **sizeof**(clientInfo);
42. //同意接收clientf請求
43. forClientSockfd = accept(sockfd,(**struct** sockaddr\*) &clientInfo, &addrlen);
44. //回傳msg
45. send(forClientSockfd,message,**sizeof**(message),0);
46. //接收msg
47. recv(forClientSockfd,inputBuffer,**sizeof**(inputBuffer),0);
48. printf("Server\_recv\_msg:\n");
49. printf("%s\n",inputBuffer);
50. }

53. **void** send\_file(**char** \*filela, **int** sockfd)
54. {
55. //常數設定
56. **int** forClientSockfd = 0, len = 0;
57. **int** addrlen = **sizeof**(clientInfo);
58. **char** text[10000],c;
59. //同意接收client請求
60. forClientSockfd = accept(sockfd,(**struct** sockaddr\*) &clientInfo, &addrlen);
61. **if** (forClientSockfd == -1)
62. perror("Server forbidden");

65. //開檔案
66. **FILE** \*fp = fopen(filela, "r");
68. //讀字串至EOF
69. **while**((c = fgetc(fp)) != EOF){
70. text[len++] = c;
71. }
72. //尾端寫下'\0'
73. text[len] = '\0';
75. printf("[server] Send data to client(%d) \n", forClientSockfd);
76. write(forClientSockfd, &len, 4);
77. write(forClientSockfd, text, len);
78. //關閉client服務
79. close(forClientSockfd);
80. //關閉檔案
81. fclose(fp);
82. }
84. **int** main(**int** argc , **char** \*argv[])
85. {
86. //socket的建立
87. **int** sockfd = 0;
88. sockfd = socket(AF\_INET , SOCK\_STREAM , 0);
89. **if** (sockfd == -1)
90. printf("[X] Server Create Failed\n");
91. **else**
92. printf("[V] Server Create Success\n");

95. //設定server
96. setting\_server(sockfd);
98. //持續接收請求
99. **while**(1){
100. //同意client請求，並傳送file給client
101. send\_file(SEND\_FILE, sockfd);
102. }
103. **return** 0;
104. }

### Client\_1.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <sys/types.h>
6. #include <sys/socket.h>
7. #include <netinet/in.h>
8. #include <arpa/inet.h>
10. #define TARGET\_IP "192.168.43.18"
11. #define TARGET\_PORT 4444
12. #define SEND\_MSG "whatthebbbb"
13. #define FILE\_NAME "get.txt"

16. **void** connect\_to\_server(**char** \*ip, **int** port, **int** sockfd)
17. {
18. **struct** sockaddr\_in info;
20. //初始化，將struct涵蓋的bits設為0
21. bzero(&info,**sizeof**(info));
22. //BSD是AF, POSIX是PF
23. //sockaddr\_in為Ipv4結構
24. info.sin\_family = PF\_INET;
25. //IP address setting
26. info.sin\_addr.s\_addr = inet\_addr(ip);
27. //htons -> Host TO Network Short integer
28. //將port(int)轉換成port(network\_short\_int)
29. info.sin\_port = htons(port);
31. **int** err = connect(sockfd,(**struct** sockaddr \*)&info,**sizeof**(info));
32. **if**(err == -1)
33. printf("[X] Connection Falied\n");
34. **else**
35. printf("[V] Connection Success\n");
36. }

39. **void** send\_msg\_to\_server(**char** \*msg, **int** sockfd)
40. {
41. //接收資訊buffer
42. **char** recvmsg[100] = {};
43. //send,recv
44. /\*
45. sd: socketfd
46. buf: buffer塞收到的資料
47. len: buffer大小
48. flags: 預設為0
49. ret val:
50. - val=0
51. - 收到的資料總Byte
52. - val!=0
53. - 真的0Byte
54. - domain設定不一致
55. - 一方正常情況結束連線
56. \*/
57. send(sockfd, msg, strlen(msg), 0);
58. recv(sockfd, recvmsg, **sizeof**(recvmsg), 0);
59. printf("%s", recvmsg);
60. printf("---Close Socket---\n");
61. close(sockfd);
62. **return**;
63. }
65. **void** download(**char** \*filename, **int** sockfd)
66. {
67. **int** len=0;
68. **char** text[500];
69. printf("[client] Read data from server\n");
70. read(sockfd, &len, 4);
71. read(sockfd, text, len);
73. //開檔寫入從server來的file
74. **FILE** \*fp = fopen(filename, "w");
75. //將text裡的字串傳入
76. fprintf(fp, "%s", text);
77. //關閉client服務
78. close(sockfd);
79. //關閉file
80. fclose(fp);
81. }


85. **int** main(**int** argc, **char** \*argv[])
86. {
87. //AF\_INET 使用的是IPv4協定
88. //SOCK\_STREAM 序列化的連接導向位元流，可以做位元流傳輸
89. //讓kernel選擇type對應的默認協議
90. //socket創建失敗則會回傳-1
91. **int** sockfd = 0;
92. sockfd = socket(AF\_INET , SOCK\_STREAM , 0);
93. **if** (sockfd == -1)
94. printf("[X] Create Failed\n");
95. **else**
96. printf("[V] Create Success\n");

99. //socket的連線
100. connect\_to\_server(TARGET\_IP, TARGET\_PORT, sockfd);
101. // 送msg到server上
102. // send\_msg\_to\_server(SEND\_MSG, sockfd);
103. // 從server載檔案
104. download(FILE\_NAME, sockfd);
106. **return** 0;
107. }

### Client2.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <sys/types.h>
6. #include <sys/socket.h>
7. #include <netinet/in.h>
8. #include <arpa/inet.h>
10. #define TARGET\_IP "192.168.43.18"
11. #define TARGET\_PORT 4444
12. #define SEND\_MSG "whatthebbbb"
13. #define FILE\_NAME "get2.txt"

16. **void** connect\_to\_server(**char** \*ip, **int** port, **int** sockfd)
17. {
18. **struct** sockaddr\_in info;
20. //初始化，將struct涵蓋的bits設為0
21. bzero(&info,**sizeof**(info));
22. //BSD是AF, POSIX是PF
23. //sockaddr\_in為Ipv4結構
24. info.sin\_family = PF\_INET;
25. //IP address setting
26. info.sin\_addr.s\_addr = inet\_addr(ip);
27. //htons -> Host TO Network Short integer
28. //將port(int)轉換成port(network\_short\_int)
29. info.sin\_port = htons(port);
31. **int** err = connect(sockfd,(**struct** sockaddr \*)&info,**sizeof**(info));
32. **if**(err == -1)
33. printf("[X] Connection Falied\n");
34. **else**
35. printf("[V] Connection Success\n");
36. }

39. **void** send\_msg\_to\_server(**char** \*msg, **int** sockfd)
40. {
41. //接收資訊buffer
42. **char** recvmsg[100] = {};
43. //send,recv
44. /\*
45. sd: socketfd
46. buf: buffer塞收到的資料
47. len: buffer大小
48. flags: 預設為0
49. ret val:
50. - val=0
51. - 收到的資料總Byte
52. - val!=0
53. - 真的0Byte
54. - domain設定不一致
55. - 一方正常情況結束連線
56. \*/
57. send(sockfd, msg, strlen(msg), 0);
58. recv(sockfd, recvmsg, **sizeof**(recvmsg), 0);
59. printf("%s", recvmsg);
60. printf("---Close Socket---\n");
61. close(sockfd);
62. **return**;
63. }
65. **void** download(**char** \*filename, **int** sockfd)
66. {
67. **int** len=0;
68. **char** text[500];
69. printf("[client] Read data from server\n");
70. read(sockfd, &len, 4);
71. read(sockfd, text, len);
73. //開檔寫入從server來的file
74. **FILE** \*fp = fopen(filename, "w");
75. //將text裡的字串傳入
76. fprintf(fp, "%s", text);
77. //關閉client服務
78. close(sockfd);
79. //關閉file
80. fclose(fp);
81. }


85. **int** main(**int** argc, **char** \*argv[])
86. {
87. //AF\_INET 使用的是IPv4協定
88. //SOCK\_STREAM 序列化的連接導向位元流，可以做位元流傳輸
89. //讓kernel選擇type對應的默認協議
90. //socket創建失敗則會回傳-1
91. **int** sockfd = 0;
92. sockfd = socket(AF\_INET , SOCK\_STREAM , 0);
93. **if** (sockfd == -1)
94. printf("[X] Create Failed\n");
95. **else**
96. printf("[V] Create Success\n");

99. //socket的連線
100. connect\_to\_server(TARGET\_IP, TARGET\_PORT, sockfd);
101. // 送msg到server上
102. // send\_msg\_to\_server(SEND\_MSG, sockfd);
103. // 從server載檔案
104. download(FILE\_NAME, sockfd);
106. **return** 0;
107. }