



计算机网络实验报告

实验五：文件传输实验（选做）

数据科学与计算机学院 17大数据与人工智能

17341015 陈鸿峰

一、实验目的

学习利用套接字传送文件。

二、实验工具

Telnet或SecureCRT

三、实验环境

本机为Ubuntu 18.04 (LTS) + gcc 7.3.0

四、实验内容

利用数据表示实验和Echo实验实现以下功能：

1. 运行服务器端程序，输入接收文件的文件夹，然后等待客户端连接，并接收客户端发来的文件并保存，直到客户端关闭连接。有重名文件时，文件名增加序号保存。
2. 客户端先连接服务器，每次输入一个文件名（包含路径）就传输它，直到输入exit时退出并关闭套接字。

五、实验结果

运行成功后，服务器截屏：

```
chhzh123@DESKTOP-PV2UBJL: /mnt/d/Assignments/ComputerNetworking/Lab5-File-Transmission$ ./server
输入接收文件夹: /mnt/e/Test
等待连接...
连接成功!
正接收文件README.md...
接收完毕!
正接收文件README.md...
接收完毕!
正接收文件index.html...
接收完毕!
正接收文件client.c...
接收完毕!
正接收文件server.c...
接收完毕!
正接收文件README.md...
接收完毕!
程序结束!
按任意键继续...
chhzh123@DESKTOP-PV2UBJL: /mnt/d/Assignments/ComputerNetworking/Lab5-File-Transmission$
```

客户端运行截屏：

```

chhzh123@DESKTOP-PV2UBJL: /mnt/d/Assignments/ComputerNetworking/Lab5-File-Transmission$ ./client
正在连接...
连接成功!
输入文件名: README.md
正在传送...
传送结束!
输入文件名: /mnt/d/README.md
正在传送...
传送结束!
输入文件名: /mnt/d/index.html
正在传送...
传送结束!
输入文件名: client.c
正在传送...
传送结束!
输入文件名: server.c
正在传送...
传送结束!
输入文件名: README.md
正在传送...
传送结束!
输入文件名: exit
程序结束!
按回车键继续...
chhzh123@DESKTOP-PV2UBJL: /mnt/d/Assignments/ComputerNetworking/Lab5-File-Transmission$

```

目标文件夹:

备份盘 (E:) > Test			
名称	修改日期	类型	大小
client.c	2019/5/21 22:38	C 文件	3 KB
index.html	2019/5/21 22:37	Chrome HTML D...	1 KB
README(2).md	2019/5/21 22:37	MD 文件	1 KB
README(3).md	2019/5/21 22:39	MD 文件	4 KB
README.md	2019/5/21 22:36	MD 文件	1 KB
server.c	2019/5/21 22:38	C 文件	4 KB

除了下面的服务器和客户端程序外，还自己写了Makefile文件方便编译。

服务器程序:

```

1  #include <sys/types.h>
2  #include <sys/socket.h>
3  #include <netinet/in.h>
4  #include <arpa/inet.h>
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <string.h>
8  #include <unistd.h>
9
10 #define BUF_LEN 100000
11
12 int main(int argc, char *argv[])
13 {
14     struct sockaddr_in fsin;          /* the from address of a client */
15     int msock, ssock;                 /* master & slave sockets */
16     char *service = "50500";

```

```
17  char    buf[100];                /* buffer for file name      */
18  char    res[BUF_LEN];            /* buffer for file context   */
19  struct  sockaddr_in sin;         /* an Internet endpoint address */
20  int      alen;                   /* from-address length       */
21  char     *pts;                   /* pointer to time string    */
22
23  char path[100];
24  printf("输入接收文件夹: ");
25  scanf("%s",path);
26
27  printf("\n等待连接...\n");
28  // create socket
29  msock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
30
31  memset(&sin, '\0', sizeof(sin));
32  sin.sin_family = AF_INET;
33  sin.sin_addr.s_addr = INADDR_ANY;
34  sin.sin_port = htons((u_short)atoi(service));
35  bind(msock, (struct sockaddr *)&sin, sizeof(sin));
36
37  listen(msock, 5); // length-5 request queue
38  alen = sizeof(struct sockaddr);
39  ssock = accept(msock, (struct sockaddr *)&fsin, &alen);
40  printf("连接成功! \n\n");
41
42  while (1){
43      memset(buf, sizeof(buf), 0);
44      memset(res, sizeof(res), 0);
45      char tmppath[100];
46      strcpy(tmppath, path);
47
48      // receive file name
49      int cc = recv(ssock, buf, BUF_LEN, 0);
50      if (cc <= 0) // client closed
51          break;
52      else if (cc > 0) {
53          buf[cc] = '\0';
54
55          printf("正接收文件%s...\n", buf);
56          int cc = recv(ssock, res, BUF_LEN, 0);
57
58          // test if file exists
59          int cnt = 1;
60          strcat(tmppath, "/"); // must in Linux!
61          strcat(tmppath, buf);
```

```

62     char file_name[100];
63     while (1){
64         strcpy(file_name,tmppath);
65         if (cnt != 1) {
66             char* p = strchr(file_name,'.');
67             if (p != 0) { // not found
68                 char suffix[10];
69                 strcpy(suffix,p);
70                 *p = '\0';
71                 strcat(file_name,"(");
72                 char numstr[10];
73                 sprintf(numstr,"%d",cnt);
74                 strcat(file_name,numstr);
75                 strcat(file_name,"");
76                 strcat(file_name,suffix);
77             } else {
78                 strcat(file_name,"(");
79                 char numstr[10];
80                 sprintf(numstr,"%d",cnt);
81                 strcat(file_name,numstr);
82                 strcat(file_name,"");
83             }
84         }
85         if (access(file_name, F_OK ) == -1)
86             break;
87         cnt++;
88     }
89     FILE* fout = fopen(file_name,"wb");
90     fwrite(res,strlen(res),1,fout);
91     fclose(fout);
92     printf("接收完毕! \n\n");
93 }
94 }
95 close(ssock);
96 close(msock);
97 printf("程序结束! \n按任意键继续...\n");
98 getchar();
99 return 0;
100 }

```

客户端程序:

```

1 #include <sys/types.h>
2 #include <sys/socket.h>
3 #include <netinet/in.h>
4 #include <arpa/inet.h>

```

```
5 #include <stdio.h>
6 #include <unistd.h>
7 #include <stdlib.h>
8 #include <string.h>
9 #include <assert.h>
10
11 #define BUF_LEN 100000
12
13 int main(int argc, char *argv[])
14 {
15     char *host = "127.0.0.1"; /* server IP to connect */
16     char *service = "50500"; /* server port to connect */
17     struct sockaddr_in sin; /* an Internet endpoint address */
18     char buf[100]; /* buffer for file name */
19     char res[BUF_LEN]; /* buffer for file context */
20     int sock; /* socket descriptor */
21     int cc; /* recv character count */
22
23     printf("正在连接...\n");
24
25     // create socket
26     sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
27
28     memset(&sin, 0, sizeof(sin));
29     sin.sin_family = AF_INET;
30     sin.sin_addr.s_addr = inet_addr(host);
31     sin.sin_port = htons((u_short)atoi(service));
32     int ret = connect(sock, (struct sockaddr *)&sin, sizeof(sin));
33     printf("连接成功! \n\n");
34
35     while (1){
36         printf("输入文件名: ");
37         scanf("%s", buf);
38         if (strcmp(buf, "exit") == 0)
39             break;
40         // split string
41         char path[100], *file_name;
42         strcpy(path, buf);
43         char* str = buf;
44         char* p = strtok(&str, "/"); // must in Linux!
45         while (p != NULL)
46         {
47             file_name = p;
48             p = strtok(&str, "/");
49         }
```

```
50
51     cc = send(sock, file_name, strlen(file_name), 0);
52     assert(cc > 0);
53
54     printf("正在传送...\n");
55     FILE* fin = fopen(path, "rb");
56     assert(fin != NULL);
57     // get file size
58     fseek(fin, 0, SEEK_END);
59     long size = ftell(fin);
60     rewind(fin);
61
62     fread(res, size, 1, fin);
63     cc = send(sock, res, strlen(res), 0);
64
65     fclose(fin);
66     printf("传送结束! \n\n");
67 }
68
69 close(sock);
70 printf("\n程序结束! \n按回车键继续...\n");
71 getchar();
72 return 0;
73 }
```

六、实验体会

本次实验使用第一第二次实验的代码，只需将用户交互界面修改一下即可。同样注意文件命名的问题，第一次实现的时候采用C++进行字符串处理，现在改回使用C语言，发现也没有想象中的那么麻烦。