Computer Networks Lab
Week-8
Sep 14, 2021

Venkata Naga Sai Ram Nomula
RA1911033010021
L2 - SWE

GIVEN REQUIREMENTS:
There are two hosts, Client and Server. The Client sends the name of the file it needs from the Server and the Server sends the contents of the file to the Client, where it is stored in a file.

TECHNICAL OBJECTIVE:
To implement an FTP application, where the Client on establishing a connection with the Server sends the name of the file it wishes to access remotely. The Server then sends the contents of the file to the Client, where it is stored.

METHODOLOGY:

Server:

- Include the necessary header files.
- Create a socket using the socket function with family AF_INET, type as SOCK_STREAM.
- Initialize server address to 0 using the bzero function.
- Assign the sin_family to AF_INET, sin_addr to INADDR_ANY, sin_port to dynamically assigned port number.
- Bind the local host address to the socket using the bind function.
- Listen on the socket for connection requests from the client.
- Accept connection requests from the Client using the accept function.
- Within an infinite loop, receive the file name from the Client.
- Open the file, read the file contents to a buffer and send the buffer to the Client.

Client:

- Include the necessary header files.
- Create a socket using the socket function with family AF_INET, type as SOCK_STREAM.
- Initialize server address to 0 using the bzero function.
- Assign the sin_family to AF_INET.
- Get the server IP address and the Port number from the console.
- Using the gethostbyname function, assign it to a hostent structure, and assign it to sin_addr of the server address structure.
- Within an infinite loop, send the name of the file to be viewed to the Server.
- Receive the file contents, store it in a file and print it on the console.

Code:

Server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<sys/stat.h>
#include<arpa/inet.h>
#include<netinet/in.h>
#include<netdb.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
int main(int argc,char *argv[]) {
int sd,ad,size;
struct sockaddr_in servaddr,cliaddr;
socklen_t clilen;
clilen=sizeof(cliaddr);
struct stat x;
char buff[100],file[10000];
FILE *fp;
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(9999);
```

```
sd=socket(AF_INET,SOCK_STREAM,0);
bind(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
listen(sd,5);
printf("%s\n","Server Is Running....");
ad=accept(sd,(struct sockaddr*)&cliaddr,&clilen);
while(1) {
bzero(buff,sizeof(buff));
bzero(file,sizeof(file));
recv(ad,buff,sizeof(buff),0);
printf("\nFile Reached %s",buff);
fp=fopen(buff,"r");
stat(buff,&x);
size=x.st_size;
fread(file,sizeof(file),1,fp);
printf("\n%s",file);
send(ad,file,sizeof(file),0);
}
}
```

Client.c:

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<stdio.h>
#include<unistd.h>
#include<string.h>
int main(int argc,char *argv[]) {
int sd,cd;
struct sockaddr_in servaddr,cliaddr;
socklen_t clilen;
char buff[100],file[10000];
struct hostent *h;
```

```
h=gethostbyname(argv[1]);
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=h->h_addrtype;
memcpy((char *)&servaddr.sin_addr.s_addr,h->h_addr_list[0],h->h_length);
servaddr.sin_port=htons(9999);
sd=socket(AF_INET,SOCK_STREAM,0);
cd=connect(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
while(1)
{
printf("%s\n","Enter the File Name :");
scanf("%s",buff);
send(sd,buff,strlen(buff)+1,0);
printf("%s\n","File Output :");
recv(sd,file,sizeof(file),0);
printf("\nfile uploaded to server");
int val;
printf("\nEnter 9 to view uploaded file, else press 9 to upload new file: ");
scanf("%d",&val);
if(val == 9)
printf("\n%s",file);
else if(val == 0)
continue;
else
break;
}
return 0;
}
```

server.c

```c
#include<sys/types.h>
#include<sys/socket.h>
#include<sys/stat.h>
#include<arpa/inet.h>
#include<netinet/in.h>
#include<netdb.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
int main(int argc,char *argv[]) {
int sd,ad,size;
struct sockaddr_in servaddr,cliaddr;
socklen_t clilen;
clilen=sizeof(cliaddr);
struct stat x;
char buff[100],file[10000];
FILE *fp;
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(9999);
sd=socket(AF_INET,SOCK_STREAM,0);
bind(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
listen(sd,5);
printf("%s\n","Server Is Running....");
ad=accept(sd,(struct sockaddr*)&cliaddr,&clilen);
while(1) {
bzero(buff,sizeof(buff));
bzero(file,sizeof(file));
recv(ad,buff,sizeof(buff),0);
printf("\nFile Reached %s",buff);
fp=fopen(buff,"r");
stat(buff,&x);
size=x.st_size;
fread(file,sizeof(file),1,fp);
printf("\n%s",file);
send(ad,file,sizeof(file),0);
}
}
```

(19 Bytes) 7:13 C and C++ Spaces: 4

client.c

```c
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<stdio.h>
#include<unistd.h>
#include<string.h>
int main(int argc,char *argv[]) {
int sd,cd;
struct sockaddr_in servaddr,cliaddr;
socklen_t clilen;
char buff[100],file[10000];
struct hostent *h;
h=gethostbyname(argv[1]);
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=h->h_addrtype;
memcpy((char *)&servaddr.sin_addr.s_addr,h->h_addr_list[0],h->h_length)
servaddr.sin_port=htons(9999);
sd=socket(AF_INET,SOCK_STREAM,0);
cd=connect(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
while(1)
{
printf("%s\n","Enter the File Name :");
scanf("%s",buff);
send(sd,buff,strlen(buff)+1,0);
printf("%s\n","File Output :");
recv(sd,file,sizeof(file),0);
printf("\nfile uploaded to server");
int val;
printf("\nEnter 9 to view uploaded file, else press 9 to upload new fil
scanf("%d",&val);
if(val == 9)
printf("\n%s",file);
else if(val == 0)
continue;
else
break;
}
```

35:18 C and C++ Spaces: 4

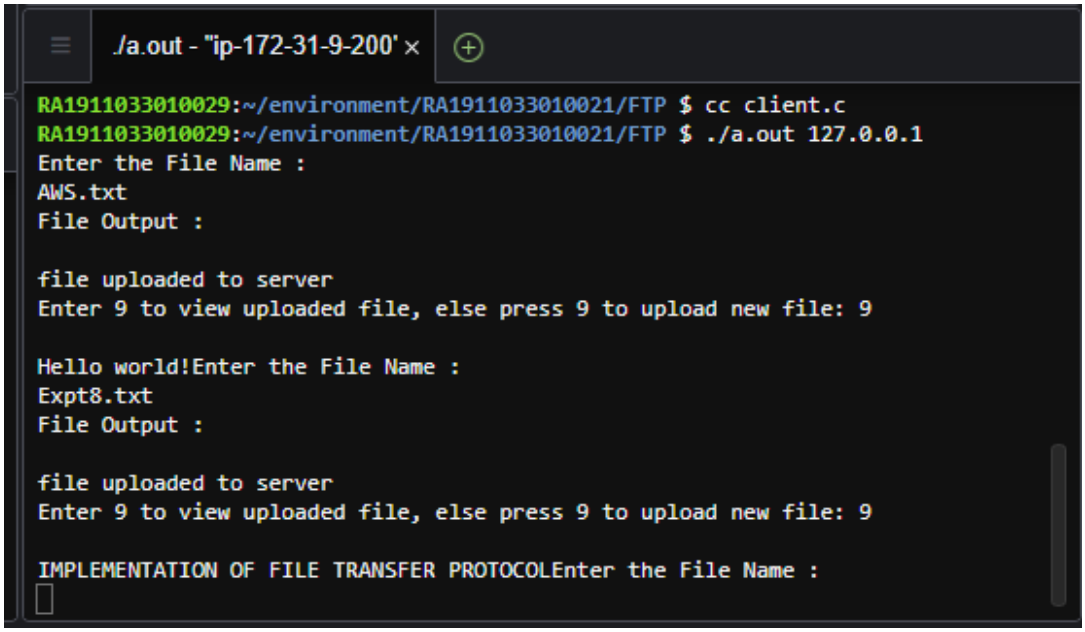
Result:

./a.out - "ip-172-31-9-200'

```
RA1911033010029:~/environment/RA1911033010021/FTP $ cc server.c
RA1911033010029:~/environment/RA1911033010021/FTP $ ./a.out
Server Is Running....

File Reached AWS.txt
Hello world!
Segmentation fault (core dumped)
RA1911033010029:~/environment/RA1911033010021/FTP $ cc server.c
RA1911033010029:~/environment/RA1911033010021/FTP $ ./a.out
Server Is Running....

File Reached AWS.txt
Hello world!
File Reached Expt8.txt
```


./a.out - "ip-172-31-9-200" ×
RA1911033010029:~/environment/RA1911033010021/FTP $ cc client.c
RA1911033010029:~/environment/RA1911033010021/FTP $ ./a.out 127.0.0.1
Enter the File Name :
AWS.txt
File Output :
file uploaded to server
Enter 9 to view uploaded file, else press 9 to upload new file: 9
Hello world!Enter the File Name :
Expt8.txt
File Output :
file uploaded to server
Enter 9 to view uploaded file, else press 9 to upload new file: 9
IMPLEMENTATION OF FILE TRANSFER PROTOCOLEnter the File Name :