

Computer Networks
RA1911030010014
Experiment - 8
Implementation Of File Transfer Protocol

Aim: To create an Implementation Of the File Transfer Protocol.

Server Code:

```
/**
 * RA1911030010014 - FTP Server
 */
#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>

#define PORT 8014

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(PORT);

    sd = socket(AF_INET, SOCK_STREAM, 0);
    bind(sd, (struct sockaddr *)&servaddr, sizeof(servaddr));
    listen(sd, 5);
    printf("Server Is Running on Port %d\n", PORT);
    ad = accept(sd, (struct sockaddr *)&cliaddr, &clilen);
    while (1)
    {
        bzero(buff, sizeof(buff));
        bzero(file, sizeof(file));
        recv(ad, buff, sizeof(buff), 0);
        fp = fopen(buff, "r");
        stat(buff, &x);
```

```
        size = x.st_size;
        fread(file, sizeof(file), 1, fp);
        send(ad, file, sizeof(file), 0);
    }
}
```

Client Code:

```
/**
 * RA1911030010014 - FTP Client
 */
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>

#define PORT 8014

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(PORT);

    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("%s\n", file);
    }
    return 0;
}
```

Output:

client.c

```
1  /**
2   * RA1911030010014 - FTP Client
3   */
4   #include <sys/types.h>
5   #include <sys/socket.h>
6   #include <netinet/in.h>
7   #include <arpa/inet.h>
8   #include <netdb.h>
9   #include <stdio.h>
10  #include <string.h>
11  #include <unistd.h>
12
13  #define PORT 8014
14
15  int main(int argc, char *argv[])
16  {
17      int sd, cd;
18      struct sockaddr_in servaddr, cliaddr;
19      socklen_t clilen;
20      char buff[100], file[10000];
21      struct hostent *h;
22
23      h = gethostbyname(argv[1]);
24      bzero(&servaddr, sizeof(servaddr));
25      servaddr.sin_family = h->h_addrtype;
26      memcpy((char *)&servaddr.sin_addr.s_addr, h->h_addr_list[0], h->h_length);
27      servaddr.sin_port = htons(PORT);
28
29      sd = socket(AF_INET, SOCK_STREAM, 0);
30      cd = connect(sd, (struct sockaddr *)&servaddr, sizeof(servaddr));
31
32      while (1)
33      {
34          printf("%s\n", "Enter the File Name:");
35          scanf("%s", buff);
36          send(sd, buff, strlen(buff) + 1, 0);
37          printf("%s\n", "File Output :");
38          recv(sd, file, sizeof(file), 0);
39          printf("%s\n", file);
40      }
41      return 0;
42  }
43  }
```

43:1 C and C++ Spaces: 4

server.c

```
1  /**
2   * RA1911030010014 - FTP Server
3   */
4   #include <sys/types.h>
5   #include <sys/socket.h>
6   #include <sys/stat.h>
7   #include <arpa/inet.h>
8   #include <netinet/in.h>
9   #include <netdb.h>
10  #include <unistd.h>
11  #include <stdio.h>
12  #include <string.h>
13
14  #define PORT 8014
15
16  int main(int argc, char *argv[])
17  {
18      int sd, ad, size;
19      struct sockaddr_in servaddr, cliaddr;
20      socklen_t clilen;
21      clilen = sizeof(cliaddr);
22      struct stat x;
23      char buff[100], file[10000];
24      FILE *fp;
25
26      bzero(&servaddr, sizeof(servaddr));
27      servaddr.sin_family = AF_INET;
28      servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
29      servaddr.sin_port = htons(PORT);
30
31      sd = socket(AF_INET, SOCK_STREAM, 0);
32      bind(sd, (struct sockaddr *)&servaddr, sizeof(servaddr));
33      listen(sd, 5);
34      printf("Server Is Running on Port %d\n", PORT);
35      ad = accept(sd, (struct sockaddr *)&cliaddr, &clilen);
36      while (1)
37      {
38          bzero(buff, sizeof(buff));
39          bzero(file, sizeof(file));
40          recv(ad, buff, sizeof(buff), 0);
41          fp = fopen(buff, "r");
42          stat(buff, &x);
43          size = x.st_size;
44          fread(file, sizeof(file), 1, fp);
45          send(ad, file, sizeof(file), 0);
46      }
47  }
```

14:18 C and C++ Spaces: 4

client.c

```
client.c:35:24: warning: incompatible implicit declaration of built-in function 'strlen'
client.c:35:24: note: include <string.h> or provide a declaration of 'strlen'
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ clear
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ gcc -o client client.c
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ ./client 127.0.0.1
Enter the File Name:
note.txt
File Output :
some-note-transfer
via-ftp
Enter the File Name:

```

2:8 Text Spaces: 4

server.c

```
[-Wformat=]
printf("Server Is Running on Port %s\n", PORT);
~^
%d
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ clear
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ gcc -o server server.c
RA1911030010018:~/environment/Sep_13/RA1911030010014 $ ./server
Server Is Running on Port 8014

```

AWS (not connected)

Result:

The required code for the Implementation Of File Transfer Protocol was written in the AWS Cloud9 environment and successfully compiled.