Assignment – 2

2. Write a file transfer program for TCP client and server (Mininet is must). Total 40 marks

- A topology with at least three hosts h1, h2, h3. h1 is the server, h2, h3 are clients. 5 marks
- A file should be uploaded by h2 to h1 and downloaded by the client h3 from h1 20 marks
- Calculate the time required to upload and download 10 marks
- Change the download/upload rate of a given file (change the buffer size) 5 marks

Solution:

Server.c Program

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/time.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#include <sys/socket.h>


#define MAX_LINE 50

#define LINSTENPORT 7788

#define SERVERPORT 8877

#define BUFFSIZE 50


void writefile(int sockfd, FILE *fp);

void sendfile(FILE *fp, int sockfd);


ssize_t total=0;


int main(int argc, char *argv[])

{

   struct timeval start_time;

   struct timeval end_time;


   int sockfd = socket(AF_INET, SOCK_STREAM, 0);
```

```
if (sockfd == -1)
{
    perror("Can't allocate sockfd");
    exit(1);
}

struct sockaddr_in clientaddr, serveraddr;
memset(&serveraddr, 0, sizeof(serveraddr));
serveraddr.sin_family = AF_INET;
serveraddr.sin_addr.s_addr = inet_addr("10.0.0.1");
serveraddr.sin_port = htons(SERVERPORT);

if (bind(sockfd, (const struct sockaddr *) &serveraddr, sizeof(serveraddr)) == -1)
{
    perror("Bind Error");
    exit(1);
}

if (listen(sockfd, LINSTENPORT) == -1)
{
    perror("Listen Error");
    exit(1);
}

printf("\nServer Listening on port %d \n", SERVERPORT);

while(1){
    socklen_t addrlen = sizeof(clientaddr);
    int connfd = accept(sockfd, (struct sockaddr *) &clientaddr, &addrlen);
    if (connfd == -1)
    {
        perror("Connect Error");
        exit(1);
    }
    // close(sockfd);
```

```
char operation[BUFFSIZE] = {0};
recv(connfd, operation, BUFFSIZE, 0);

char filename[BUFFSIZE] = {0};

if(strcmp(operation, "upload") == 0){
    if (recv(connfd, filename, BUFFSIZE, 0) == -1) {
    perror("Can't receive filename");
    exit(1);
}

    FILE *fp = fopen(filename, "wb");
    if (fp == NULL)
    {
        perror("Can't open file");
        exit(1);
    }

    char addr[INET_ADDRSTRLEN];
    printf("\n----------------------------------------------------------------\n");
    printf("Uploading file: %s by %s\n", filename, inet_ntop(AF_INET, &clientaddr.sin_addr, addr, INET_ADDRSTRLEN));
    gettimeofday(&start_time, NULL);
    writefile(connfd, fp);
    gettimeofday(&end_time, NULL);
    fclose(fp);

    printf("Upload Success, Total Bytes = %ld\n", total);
    printf("Time taken: %ld secs \n\n", end_time.tv_sec - start_time.tv_sec);
}

if(strcmp(operation, "download") == 0){
    if (recv(connfd, filename, BUFFSIZE, 0) == -1) {
        perror("Can't receive filename");
        exit(1);
```

```
            }

            FILE *fp = fopen(filename, "rb");
            if (fp == NULL)
            {
                perror("Can't open file");
                exit(1);
            }

            char addr[INET_ADDRSTRLEN];
            printf("\n--------------------------------------------------------------\n");
            printf("Downloading file: %s by %s\n", filename, inet_ntop(AF_INET, &clientaddr.sin_addr, addr, INET_ADDRSTRLEN));

            gettimeofday(&start_time, NULL);
            sendfile(fp, connfd);
            gettimeofday(&end_time, NULL);
            fclose(fp);

            printf("File download success, Total Bytes = %ld\n", total);
            printf("Time taken: %ld secs \n\n", end_time.tv_sec - start_time.tv_sec);
        }

        close(connfd);
    }

    return 0;
}

// write file fn def
void writefile(int sockfd, FILE *fp)
{
    ssize_t n;
    char buff[MAX_LINE] = {0};
    while ((n = recv(sockfd, buff, MAX_LINE, 0)) > 0)
    {
                      total+=n;
        if (n == -1)
        {
            perror("Receive File Error");
            exit(1);
        }

        if (fwrite(buff, sizeof(char), n, fp) != n)
        {
            perror("Write File Error");
            exit(1);
        }
        memset(buff, 0, MAX_LINE);
    }
}

// send file for download

void sendfile(FILE *fp, int sockfd)
{
    int n;
    char sendline[MAX_LINE] = {0};
    while ((n = fread(sendline, sizeof(char), MAX_LINE, fp)) > 0)
    {
                      total+=n;
        if (n != MAX_LINE && ferror(fp))
        {
            perror("Read File Error");
            exit(1);
        }

        if (send(sockfd, sendline, n, 0) == -1)
        {
            perror("Can't send file");
    exit(1);
    }
    memset(sendline, 0, MAX_LINE);
  }
}
```

Client.c program - the program to upload file to server

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <libgen.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <time.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <sys/socket.h>

#define MAX_LINE 50
#define LINSTENPORT 7788
#define SERVERPORT 8877
#define BUFFSIZE 50

void sendfile(FILE *fp, int sockfd);
ssize_t total=0;
int main(int argc, char* argv[])
{
  char buff[BUFFSIZE] = {0};
  struct timeval start_time;
  struct timeval end_time;

  if (argc != 3)
  {
    perror("usage:./client u <filepath>");
    exit(1);
  }

  int sockfd = socket(AF_INET, SOCK_STREAM, 0);
  if (sockfd < 0)
  {
    perror("Can't allocate sockfd");
    exit(1);
  }

  struct sockaddr_in serveraddr;
  memset(&serveraddr, 0, sizeof(serveraddr));
  serveraddr.sin_family = AF_INET;
  serveraddr.sin_port = htons(SERVERPORT);
  serveraddr.sin_addr.s_addr = inet_addr("10.0.0.1");


  if (connect(sockfd, (const struct sockaddr *) &serveraddr, sizeof(serveraddr)) < 0)
  {
    perror("Connect Error");
    exit(1);
  }


  // getting the user operation
  strcpy(buff, argv[1]);
  send(sockfd,buff, BUFFSIZE, 0);

  char *filename = basename(argv[2]);
  if (filename == NULL)
  {
    perror("Can't get filename");
    exit(1);
  }
```

```
    strncpy(buff, filename, strlen(filename));
    if (send(sockfd, buff, BUFFSIZE, 0) == -1)
    {
        perror("Can't send filename");
        exit(1);
    }
    printf("\nUploading File... \"%s\"\n", filename);

    FILE *fp = fopen(argv[2], "rb");
    if (fp == NULL)
    {
        perror("Can't open file");
        exit(1);
    }

    gettimeofday(&start_time, NULL);
    sendfile(fp, sockfd);
    gettimeofday(&end_time, NULL);

    //puts("Send Success");

    printf("Upload Success, Total Bytes = %ld\n", total);
    printf("Time taken: %ld sec\n", end_time.tv_sec - start_time.tv_sec);
    printf("Upload rate: %ld B/s (%.2lf MB/s)\n\n", total/(end_time.tv_sec - 
start_time.tv_sec),(double)(total/(end_time.tv_sec - start_time.tv_sec))/1000000);

    fclose(fp);
    close(sockfd);
    return 0;
}

void sendfile(FILE *fp, int sockfd)
{
    int n;
    char sendline[MAX_LINE] = {0};
```

```
  while ((n = fread(sendline, sizeof(char), MAX_LINE, fp)) > 0)
  {
          total+=n;
    if (n != MAX_LINE && ferror(fp))
    {
      perror("Read File Error");
      exit(1);
    }

    if (send(sockfd, sendline, n, 0) == -1)
    {
      perror("Can't send file");
      exit(1);
    }
    memset(sendline, 0, MAX_LINE);
  }
}
```

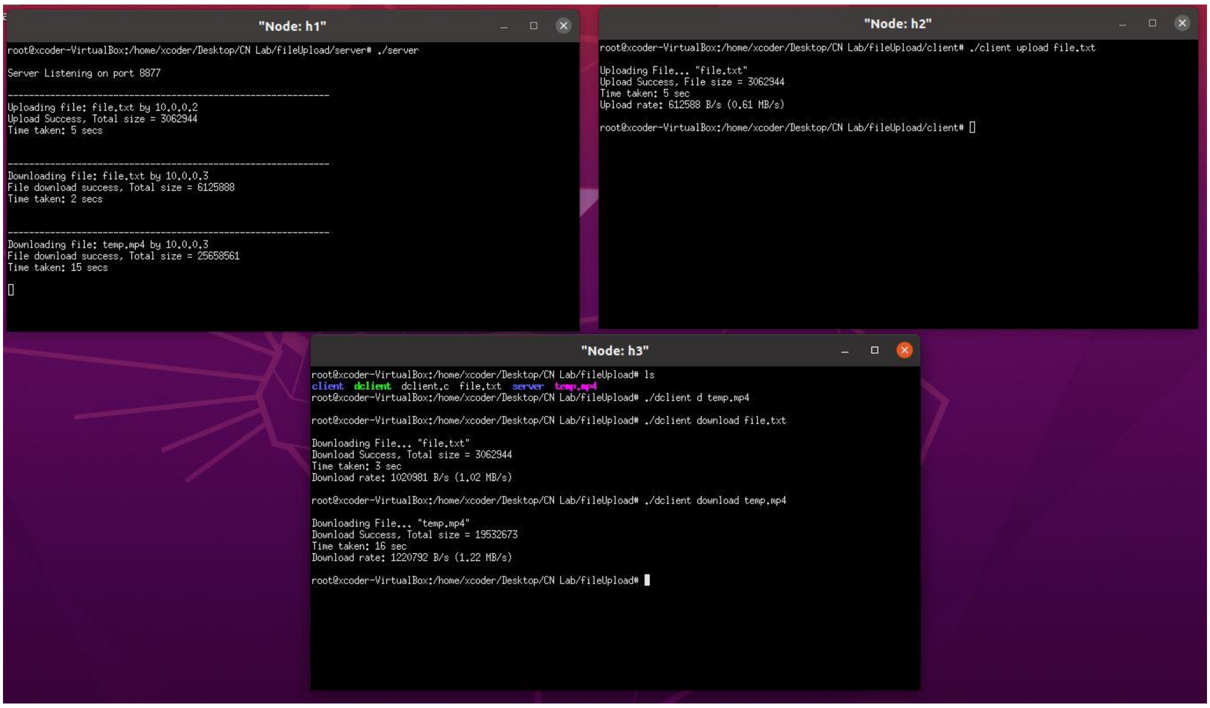
dclient.c - the program to download file

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <libgen.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <time.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <sys/socket.h>

#define MAX_LINE 50
#define LINSTENPORT 7788
#define SERVERPORT 8877
#define BUFFSIZE 50
```

```
}

void writefile(int sockfd, FILE *fp);
ssize_t total=0;

int main(int argc, char* argv[])
{
    char buff[BUFFSIZE] = {0};
    struct timeval start_time;
    struct timeval end_time;

    if (argc != 3)
    {
        perror("usage:./client u <filepath>");
        exit(1);
    }

    int sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0)
    {
        perror("Can't allocate sockfd");
        exit(1);
    }

    struct sockaddr_in serveraddr;
    memset(&serveraddr, 0, sizeof(serveraddr));
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_port = htons(SERVERPORT);
    serveraddr.sin_addr.s_addr = inet_addr("10.0.0.1");


    if (connect(sockfd, (const struct sockaddr *) &serveraddr, sizeof(serveraddr)) < 0)
    {
        perror("Connect Error");
        exit(1);
    }
    // getting the user operation
    strcpy(buff, argv[1]);
    send(sockfd,buff, BUFFSIZE, 0);

    char *filename = basename(argv[2]);
    if (filename == NULL)
    {
        perror("Can't get filename");
        exit(1);
    }

    strncpy(buff, filename, strlen(filename));
    if (send(sockfd, buff, BUFFSIZE, 0) == -1)
    {
        perror("Can't send filename");
        exit(1);
    }

    printf("\nDownloading File... \"%s\" \n", filename);

    FILE *fp = fopen(filename, "wb");
    if (fp == NULL)
    {
        perror("Can't open file");
        exit(1);
    }

    gettimeofday(&start_time, NULL);
    writefile(sockfd, fp);
    gettimeofday(&end_time, NULL);

    printf("Download Success, Total Bytes = %ld\n", total);
    printf("Time taken: %ld sec\n", end_time.tv_sec - start_time.tv_sec);
    printf("Download rate: %ld B/s (%.2lf MB/s)\n\n", total/(end_time.tv_sec -
start_time.tv_sec),(double)(total/(end_time.tv_sec - start_time.tv_sec))/1000000);
```

```
   fclose(fp);

   close(sockfd);

   return 0;

}



// write file fn def

void writefile(int sockfd, FILE *fp)

{

   ssize_t n;

   char buff[MAX_LINE] = {0};

   while ((n = recv(sockfd, buff, MAX_LINE, 0)) > 0)

   {

                total+=n;

      if (n == -1)

      {

         perror("Receive File Error");

         exit(1);

      }



      if (fwrite(buff, sizeof(char), n, fp) != n)

      {

         perror("Write File Error");

         exit(1);

      }

      memset(buff, 0, MAX_LINE);

   }

}
```

Output:



```
root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload/server# ./server

Server Listening on port 8877

-------------------------------------------------------------
Uploading file: file.txt by 10.0.0.2
Upload Success, Total size = 3062944
Time taken: 5 secs


-------------------------------------------------------------
Downloading file: file.txt by 10.0.0.3
File download success, Total size = 6125888
Time taken: 2 secs


-------------------------------------------------------------
Downloading file: temp.mp4 by 10.0.0.3
File download success, Total size = 25658561
Time taken: 15 secs

```

```
root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload/client# ./client upload file.txt

Uploading File... "file.txt"
Upload Success, File size = 3062944
Time taken: 5 sec
Upload rate: 612588 B/s (0.61 MB/s)

root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload/client# 
```

```
root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload# ls
client  dclient  dclient.c  file.txt  server  temp.mp4
root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload# ./dclient d temp.mp4

root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload# ./dclient download file.txt

Downloading File... "file.txt"
Download Success, Total size = 3062944
Time taken: 3 sec
Download rate: 1020981 B/s (1.02 MB/s)

root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload# ./dclient download temp.mp4

Downloading File... "temp.mp4"
Download Success, Total size = 19532673
Time taken: 16 sec
Download rate: 1220792 B/s (1.22 MB/s)

root@xcoder-VirtualBox:/home/xcoder/Desktop/CN Lab/fileUpload# 
```