



TP4

Teleinformática

Sockets

Gustavo Marin
Alejandro Coello

1 de septiembre de 2019

1. Objetivo

Enviar un fichero entre dos computadoras utilizando un socket TCP.

2. Requerimientos

El servidor deberá calcular throughput (i.e., el número de kilobytes por segundo – KB/s) que recibió y el porcentaje bytes recibidos del total esperados. Para simplificar, se deberá utilizar una aproximación del tiempo entre varios `recv(...)` que sean mayores o iguales a 1 segundo

3. Implementación

En el servidor se obtiene el tamaño del fichero a recibir y se recibirán datos hasta alcanzar el tamaño del fichero, esto se realizara mediante un **while**, dentro el cual se tiene la recepción de datos, la escritura en el fichero, un control del tiempo, el cálculo del porcentaje recibido y la impresión de la velocidad de transmisión y el porcentaje.

3.1. Cálculo del porcentaje recibido

Para calcular el porcentaje recibido se realiza una división entre la cantidad de bytes recibidos entre el total a recibir y se multiplica por 100

```
per = ((total_byt * 100) / fileSize);
```

3.2. Cálculo de la velocidad de transmisión

Para calcular la velocidad de transmisión se inicia la cuenta de un tiempo antes de entrar al ciclo **while** y al acabar la recepción, este tiempo se ira acumulando y cuando sea mayor a 1 segundo se realizara la división entre los bytes recibidos hasta ese momento entre el tiempo transcurrido.

```
speed = (byps / 1000) / total_time;
```

3.3. Escritura del fichero

Para realizar la escritura del fichero se utiliza **fwrite**, se escribira sobre un fichero previamente creado con **fopen**.

```
// Creacion del fichero  
FILE* pFile = fopen(fileName, "w+");  
// Escritura del fichero  
fwrite(buffer, sizeof(char), byt_rcv, pFile);
```

4. Resultados

A continuación se adjuntaran imágenes de la aplicación funcionando.

En la siguiente imagen se muestra la ejecución del servidor con todos los requerimientos funcionando, además se puede observar el formato de introducción de argumentos, nombre del fichero, puerto y tamaño del buffer.

```
gustavo@gustavo-Inspiron-15-3567: ~/Documents/Teleinformatica/Sockets 70x18
gustavo@gustavo-Inspiron-15-3567:~/Documents/Teleinformatica/Sockets$
./server video.mp4 8888 4096
Trying to receive 5253880 Bytes...
Elapsed time 1.0020 s Velocidad 183.636 KB/s bytes 188950
Expected: 4096 Read: 601 Porcentaje 3 %
Elapsed time 1.0006 s Velocidad 75.951 KB/s bytes 267759
Expected: 4096 Read: 1460 Porcentaje 5 %
Elapsed time 1.0130 s Velocidad 57.257 KB/s bytes 327606
Expected: 4096 Read: 1460 Porcentaje 6 %
Elapsed time 1.0077 s Velocidad 48.626 KB/s bytes 377855
Expected: 4096 Read: 1460 Porcentaje 7 %
Elapsed time 1.0114 s Velocidad 41.528 KB/s bytes 421665
Expected: 4096 Read: 1460 Porcentaje 8 %
Elapsed time 1.0184 s Velocidad 37.314 KB/s bytes 461408
Expected: 4096 Read: 1460 Porcentaje 8 %
Elapsed time 1.0263 s Velocidad 34.102 KB/s bytes 497394
Expected: 4096 Read: 1460 Porcentaje 9 %
```

Figura 1: Servidor

En la siguiente imagen se muestra la ejecución del cliente, además del formato de la introducción de argumentos, archivo a enviar, IP del servidor, puerto y tamaño del buffer.

```
~/TP4
Gustavo Marin Ovando@DESKTOP-OL28QK1 ~/TP4
$ ./client file.mp4 192.168.1.70 8888 4096
Trying to send 5253880 Bytes to 192.168.1.70:8888...
Bytes sent: 4096, Total Bytes sent: 4096
Bytes sent: 4096, Total Bytes sent: 8192
Bytes sent: 4096, Total Bytes sent: 12288
Bytes sent: 4096, Total Bytes sent: 16384
Bytes sent: 4096, Total Bytes sent: 20480
Bytes sent: 4096, Total Bytes sent: 24576
Bytes sent: 4096, Total Bytes sent: 28672
Bytes sent: 4096, Total Bytes sent: 32768
Bytes sent: 4096, Total Bytes sent: 36864
Bytes sent: 4096, Total Bytes sent: 40960
Bytes sent: 4096, Total Bytes sent: 45056
Bytes sent: 4096, Total Bytes sent: 49152
Bytes sent: 4096, Total Bytes sent: 53248
Bytes sent: 4096, Total Bytes sent: 57344
Bytes sent: 4096, Total Bytes sent: 61440
```

Figura 2: Cliente

Appendices

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

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

    if (argc != 4) {
        printf("usage: %s fileName port bufferSize\n", argv[0]);
        return -1;
    }

    // Nombre del archivo a recibir
    char* fileName = argv[1];

    // Numero de puerto
    int port = atoi(argv[2]);

    // Tamaño del buffer
    int bufferSize = atoi(argv[3]);

    char* buffer = malloc(bufferSize);

    struct sockaddr_in stSockAddr;
    int SocketFD = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    if(-1 == SocketFD) {
        perror("can not create socket");
        exit(EXIT_FAILURE);
    }

    memset(&stSockAddr, 0, sizeof(struct sockaddr_in));

    stSockAddr.sin_family = AF_INET;
    stSockAddr.sin_port = htons(port);
```

```

stSockAddr.sin_addr.s_addr = INADDR_ANY;
if(-1 == bind(SocketFD, (const struct sockaddr *)&stSockAddr,
↪ sizeof(struct sockaddr_in))) {
    perror("error bind failed");
    close(SocketFD);
    exit(EXIT_FAILURE);
}

if(-1 == listen(SocketFD, 10)) {
    perror("error listen failed");
    close(SocketFD);
    exit(EXIT_FAILURE);
}

int ConnectFD = accept(SocketFD, NULL, NULL);

if(0 > ConnectFD) {
    perror("error accept failed");
    close(SocketFD);
    exit(EXIT_FAILURE);
}

long fileSize;
int fsize = recv(ConnectFD, &fileSize, sizeof(fileSize), 0);
if(fsize != sizeof(fileSize)) {
    printf("Error reading file size\n");
    exit(-1);
}
printf("Trying to receive %ld Bytes...\n", fileSize);

int totalBytesReceived = 0;

FILE* pFile = fopen(fileName, "w+");
// here ADD your code
int per, byt_rcv = 0, total_byt = 0, byps = 0;
float seconds, total_time, speed;
clock_t start = clock();
    while (total_byt < fileSize) {
        byt_rcv = recv(ConnectFD, buffer, bufferSize, 0);
        fwrite(buffer, sizeof(char), byt_rcv, pFile);
        clock_t end = clock();
        total_byt += byt_rcv;

```

```

    byps += byt_rcv;
    per = ((total_byt * 100) / fileSize);
    seconds = ((float)(end - start) / CLOCKS_PER_SEC);
    total_time += seconds;
    if (total_time >= 1) {
        speed = (byps / 1000) / total_time;
        printf("Elapsed time %.4f s\tVelocidad %.3f KB/s\tbytes %d\n",
            ↪ total_time, speed, total_byt);
        printf("Expected: %d\tRead: %d\tPorcentaje %d %%\n", bufferSize,
            ↪ byt_rcv, per);
        total_time = 0;
        byps = 0;
    }
}

fclose(pFile);

printf("%s written\nDONE\n", fileName);

shutdown(ConnectFD, SHUT_RDWR);

close(ConnectFD);
close(SocketFD);

return 0;
}

```