Федеральное агентство связи

Сибирский Государственный Университет Телекоммуникаций и Информатики

Кафедра ТС и ВС

Лабораторная работа № 2 **“ Клиент-серверные приложения”**

По дисциплине: **Сетевое программное обеспечение**

**Выполнил:** Тиванов Данил Евгеньевич

Группа: ИА-832

**Проверил:** Лошкарев Александр Васильевич

**Задание:**

1. Написать клиент-серверную программу на основе транспортного протокола UDP. Реализовать: подтверждение приема для каждой датаграммы, сохранение целостности всей информации. Клиент передает файл или сообщение (несколькими датаграммами), сервер принимает.
2. Продемонстрировать реализованные возможности программ согласно заданию, при одновременной передаче файлов от нескольких клиентов к серверу. Например, при запуске сервера указать - какие пакеты и сколько раз будут потеряны. Результат правильности приема выводить на экран.
3. Реализация - на языке C/C++, консольные приложения.

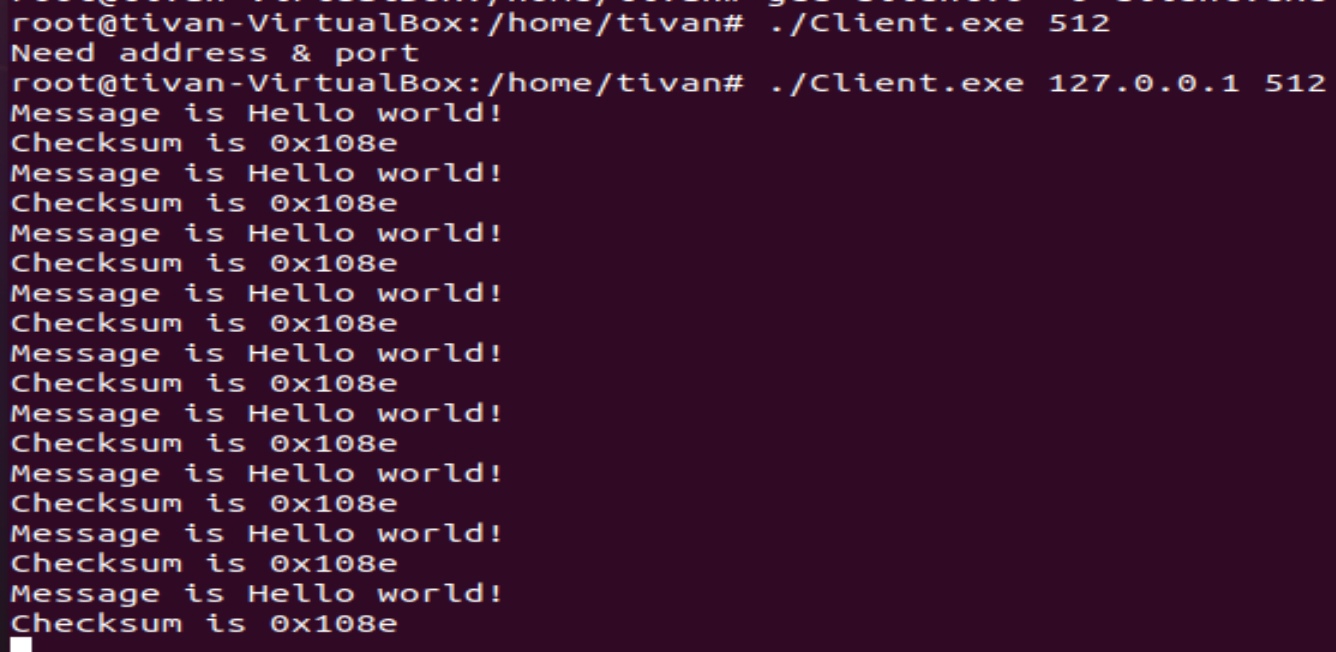
Серверная программа должна находить номер свободного порта и выводить его на экран. При запуске клиентской программы задавать со строки IP адрес сервера и порт (функции bind, getsockname).

**Результаты:**

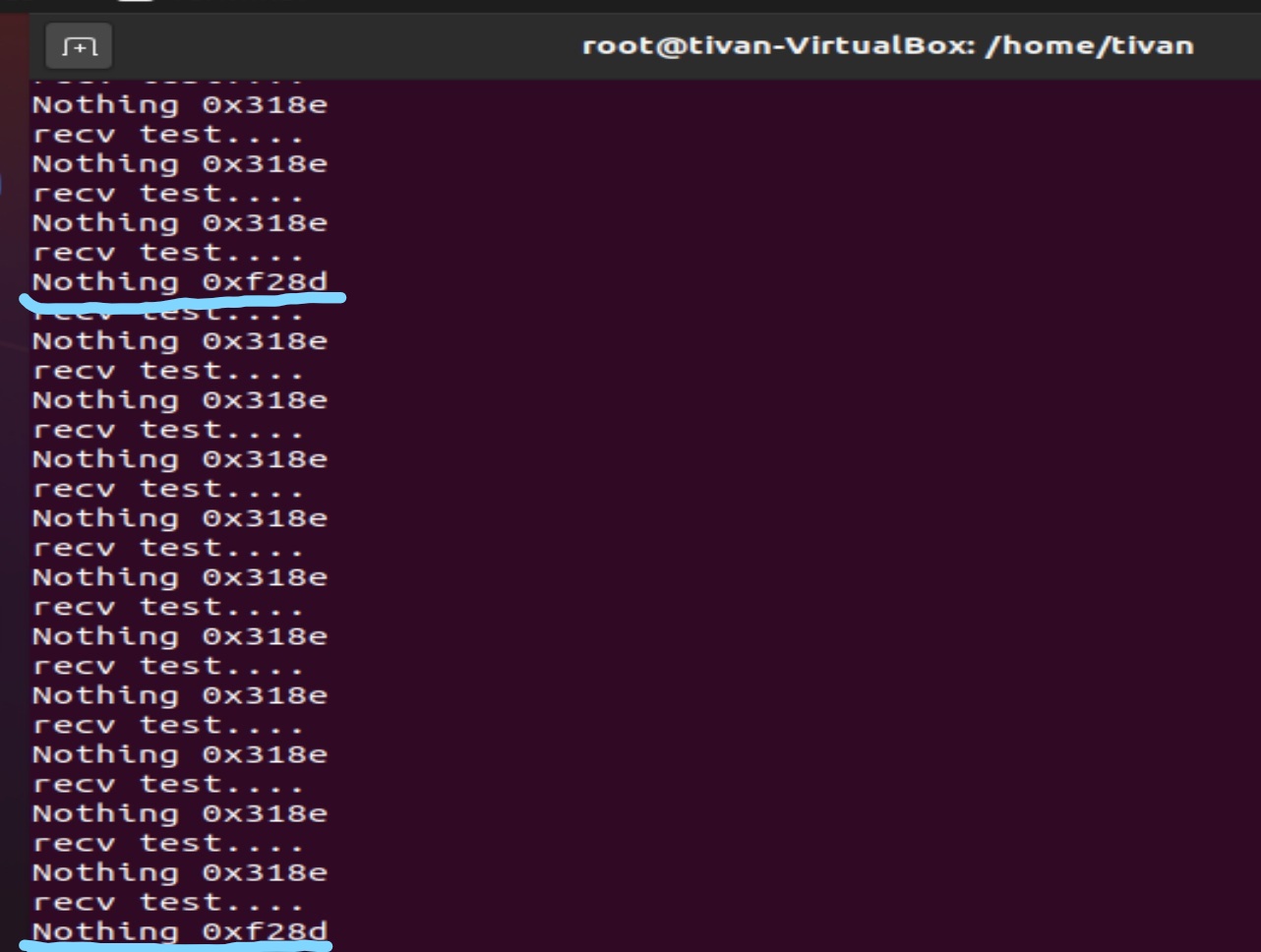
Запуск сервера и получение контрольной суммы 0x318e от клиента:



Запуск клиента и отправка контрольной суммы 0x108e:



10 процентный шанс на контрольную сумму 0xf28d



**Код программы:**

Server.c

#include <stdio.h>

#include <errno.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <unistd.h>

char arr[6]; // 65536 - max

unsigned short **checksum**(unsigned short \*ptr, int nbytes)

{

register long sum;

unsigned short oddbyte;

register short answer;

sum=0;

while(nbytes>1) {

sum+=\*ptr++;

nbytes-=2;

}

if(nbytes==1) {

oddbyte=0;

\*((u\_char\*)&oddbyte)=\*(u\_char\*)ptr;

sum+=oddbyte;

}

sum = (sum>>16)+(sum & 0xffff);

sum = sum + (sum>>16);

answer=(short)~sum;

sprintf(arr, "%u", answer);

return(answer);

}

int main(int argc, char \*argv){

char Hz[100];

char csum\_client[6]; // 65536 - max

char hex[6];

int sock = socket(PF\_INET, SOCK\_DGRAM, IPPROTO\_UDP);

struct **sockaddr\_in** sa;

int bound, len;

ssize\_t recsize;

socklen\_t \*address\_len = NULL;

sa.sin\_family = AF\_INET;

sa.sin\_addr.s\_addr = htonl(INADDR\_ANY);

sa.sin\_port = htons( 512 );

if(bind(sock, (struct sockaddr \*)&sa, sizeof(sa)) < 0){

printf("BIND ERROR %s\n", strerror(errno));

return -1;

}

if(getsockname(sock, (struct sockaddr \*)&sa, (socklen\_t \*)&len) < 0){

printf("GETSOCKNAME %s\n", strerror(errno));

return -1;

}

printf("port: %hd\n", htons(sa.sin\_port));

bound = bind(sock, (struct sockaddr\*)&sa, sizeof(struct sockaddr));

while(1){

printf("recv test....\n");

recsize = recvfrom(sock, (void\*)Hz, 100, 0, (struct sockaddr\*)&sa, address\_len);

for(int i = 0; i < 5; i++){

csum\_client[i] = Hz[i + strlen(Hz) - 5];

}

Hz[strlen(Hz) - 5] = '\0';

unsigned short csum\_server = checksum(Hz, strlen(Hz));

if(atoi(arr) == atoi(csum\_client)){

sprintf(&hex, "%x", atoi(csum\_client));

printf("Datagram: %s\n", Hz);

printf("Checksum from client is 0x%s; Checksum from server: 0x%02x\n", hex, csum\_server);

}else{

printf("Nothing 0x%02x\n", csum\_server);

}

sleep(1);

}

}

Client.c

#include <stdio.h>

#include <errno.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <unistd.h>

#include <time.h>

char short\_arr[6]; // 65536 - max

unsigned short **checksum**(unsigned short \*ptr, int nbytes)

{

register long sum;

unsigned short oddbyte;

register short answer;

sum=0;

while(nbytes>1) {

sum+=\*ptr++;

nbytes-=2;

}

if(nbytes==1) {

oddbyte=0;

\*((u\_char\*)&oddbyte)=\*(u\_char\*)ptr;

sum+=oddbyte;

}

sum = (sum>>16)+(sum & 0xffff);

sum = sum + (sum>>16);

answer=(short)~sum;

sprintf(short\_arr, "%u", answer);

return(answer);

}

int **main**(int argc, char \*\*argv){

if(argc != 3){

printf("Need address & port\n");

return -1;

}

srand(time(NULL));

int sock;

struct **sockaddr\_in** sa;

int bytes\_sent;

char \*buffer = "Hello world!", datagram[4096];

int buffer\_length;

while(1){

buffer = "Hello world!";

buffer\_length = strlen( buffer ) + 1;

printf("Message is %s\nChecksum is 0x%02x\n", buffer, checksum(buffer, buffer\_length));

int r = rand() % 10; // 0 to 9

if(r == 0){ // 10% loss

buffer = "Hello\_world?";

}

sock = socket(PF\_INET, SOCK\_DGRAM, IPPROTO\_UDP);

if(sock == -1){

printf("ERROR CREATION NETWORK");

return 0;

}

sa.sin\_family = AF\_INET;

sa.sin\_addr.s\_addr = inet\_addr(argv[1]);

sa.sin\_port = htons(atoi(argv[2]));

strcpy(datagram, buffer);

// printf("1 %s\n", datagram);

strcat(datagram, short\_arr);

// printf("2 %s\n", datagram);

bytes\_sent = sendto(sock,

datagram,

strlen(datagram) + 1,

0,

(struct sockaddr\*)&sa,

sizeof(struct sockaddr\_in));

if(bytes\_sent < 0)

printf("ERROR SENDING POCKET: %s\n", strerror(errno));

sleep(1);

close(sock);

}

return 0;

}