Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных систем и технологий»

**Семейство протоколов TCP/IP. Сокеты в UNIX и работа с ними**

Лабораторная работа №11 по учебной дисциплине «Операционные системы»

По направлению 09.03.01 «Информатика и вычислительная техника»

Отчёт

Принял преподаватель:

\_\_\_\_\_\_\_\_\_ Ананьев В. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Выполнил студент группы 8091:

\_\_\_\_\_\_\_ Шляханов Д. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Великий Новгород

2020

**Цель работы:** объединить полученные в процессе выполнения предыдущих лабораторных работ знания в рамках одной комплексной программы.

**Задание:**

Написать программу-клиент и программу-сервер для выполнения следующей задачи: сервер запускается и ждёт запросов от клиентов; клиент подключается к серверу и может запрашивать файлы, находящиеся на сервере (в папке с программой-сервером), задавая полное имя файла; если файл найден, сервер передаёт файл клиенту, если не найден, то сервер возвращает сообщение о том, что файл не найден.

Сервер должен параллельно обслуживать несколько клиентов (для каждого свой процесс). Клиент может запрашивать файлы последовательно вводя имена. Выход из программы клиента можно осуществить по текстовой команде (например, «exit»). Сервер так же должен иметь возможность вводить команды (например, для закрытия программы-сервера).

Протокол транспортного уровня – TCP.

При запуске программы-сервера через параметры командной строки передаётся порт, на котором будет работать сервер. Если данный порт занят, программа-сервер должна получить порт по усмотрению системы. При успешном запуске сервер должен выдать на экран номер порта, на котором он работает.

При запуске программы-клиента через параметры командной строки передаётся IP-адрес и порт (сокет) программы-сервера в формате <IP-address>:<port> (например, 192.168.0.100:4242).

**Содержание файла server.c:**

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <string.h>

#include <stdio.h>

#include <errno.h>

#include <unistd.h>

#include <limits.h>

#include <stdlib.h>

#include <pthread.h>

struct thread\_args

{

int sockfd;

};

long getFileSize(FILE \*input)

{

long fileSizeBytes;

fseek(input, 0, SEEK\_END);

fileSizeBytes = ftell(input);

fseek(input, 0, SEEK\_SET);

return fileSizeBytes;

}

void \*client\_thread(void \*t\_args)

{

struct thread\_args \*args = (struct thread\_args \*)t\_args;

int newsockfd = args->sockfd;

while (1)

{

char name\_file[256];

long filesize;

int n = read(newsockfd, name\_file, 256);

if (n == 0)

{

close(newsockfd);

exit(0);

}

FILE \*fin = fopen(name\_file, "r");

if (fin == NULL)

{

filesize = -1;

write(newsockfd, &filesize, sizeof(filesize));

}

else

{

filesize = getFileSize(fin);

write(newsockfd, &filesize, sizeof(filesize));

char buffer[filesize];

fread(buffer, sizeof(char), filesize, fin);

fclose(fin);

write(newsockfd, buffer, sizeof(buffer));

}

}

}

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

{

int listener, sock;

int bytes\_read;

listener = socket(PF\_INET, SOCK\_STREAM, 0);

struct sockaddr\_in addr, cliaddr;

bzero(&addr, sizeof(addr));

addr.sin\_family = AF\_INET;

addr.sin\_port = htons(atoi(argv[1]));

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

if (bind(listener, (struct sockaddr \*)&addr, sizeof(addr)) < 0)

{

addr.sin\_port = 0;

if (bind(listener, (struct sockaddr \*)&addr, sizeof(addr)) < 0)

{

perror("Bind");

close(listener);

exit(2);

}

}

socklen\_t servlen = sizeof(addr);

getsockname(listener, (struct sockaddr \*)&addr, &servlen);

printf("Listening on port: %d\n", ntohs(addr.sin\_port));

listen(listener, 10);

if (fork() == 0)

{

while (1)

{

socklen\_t clilen = sizeof(cliaddr);

sock = accept(listener, (struct sockaddr \*)&cliaddr, &clilen);

struct thread\_args \*new\_thread\_args;

new\_thread\_args->sockfd = sock;

pthread\_t thread;

pthread\_create(&thread, NULL, client\_thread, new\_thread\_args);

}

}

else

{

printf("Ведите exit чтобы выйти\n");

char command[256];

while (1)

{

scanf("%s", command);

if (strcmp(command, "exit") == 0)

{

exit(0);

}

}

}

}

**Содержание файла client.c:**

#include <sys/types.h>

#include <sys/socket.h>

#include <sys/mman.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <fcntl.h>

#include <string.h>

#include <stdio.h>

#include <errno.h>

#include <unistd.h>

#include <limits.h>

#include <stdlib.h>

#include <pthread.h>

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

{

long file\_size, n;

char name\_file[256];

int port = atoi(strchr(argv[1], ':') + 1);

int pointer = strchr(argv[1], ':') - argv[1];

char ip[pointer + 1];

strncpy(ip, argv[1], pointer);

ip[pointer] = '\0';

int sock;

sock = socket(PF\_INET, SOCK\_STREAM, 0);

struct sockaddr\_in addr;

bzero(&addr, sizeof(addr));

addr.sin\_family = AF\_INET;

addr.sin\_port = htons(port);

inet\_aton(ip, &addr.sin\_addr);

if (connect(sock, (struct sockaddr \*)&addr, sizeof(addr)) < 0)

{

perror("Connect");

exit(2);

}

printf("Ведите название файла для скачивания или ведите exit чтобы выйти\n");

while (1)

{

scanf("%s", name\_file);

if (strcmp(name\_file, "exit") == 0)

{

exit(0);

}

else

{

write(sock, name\_file, strlen(name\_file) + 1);

long n = read(sock, &file\_size, sizeof(file\_size));

if (n == 0)

{

perror("Connect");

exit(1);

}

else if (file\_size == -1)

{

printf("Такого файла на сервере нет\n");

}

else

{

printf("Процесс скачивания...\n");

int downloaded\_file = open(name\_file, O\_RDWR | O\_CREAT, 0600);

ftruncate(downloaded\_file, file\_size);

char \*downloaded\_file\_data = (char \*)mmap(NULL, n, PROT\_WRITE | PROT\_READ, MAP\_SHARED, downloaded\_file, 0);

read(sock, downloaded\_file\_data, file\_size);

munmap(downloaded\_file\_data, file\_size);

close(downloaded\_file);

printf("Файл скачен\n");

}

}

}

close(sock);

}

**Вывод:** В процессе выполнения лабораторной работы я на практике закрепил знания о различных аспектах UNIX.