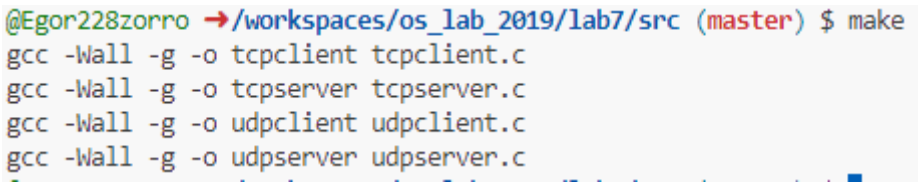
Лабораторная работа №7

Задание 1



Протокол TCP (Transmission Control Protocol)

TCP — это **протокол транспортного уровня**, обеспечивающий **надежную передачу данных** между двумя компьютерами.

Основные свойства:

- **Установление соединения**
- **Гарантия доставки** данных.
- **Контроль целостности** — проверка ошибок.
- **Сохранение порядка** пакетов.
- **Повторная передача** утерянных данных.

Протокол UDP (User Datagram Protocol)

UDP — это **простой протокол транспортного уровня**, который **не гарантирует доставку** данных.

Свойства:

- **Без установления соединения**
- **Нет гарантии доставки** или порядка.
- **Быстрая передача** данных (меньше задержек).

Нет контроля ошибок на уровне транспорта.

makefile:

```
CC = gcc
CFLAGS = -Wall -g

all: tcpclient tcpserver udpclient udpserver

tcpclient: tcpclient.c
    $(CC) $(CFLAGS) -o tcpclient tcpclient.c

tcpserver: tcpserver.c
    $(CC) $(CFLAGS) -o tcpserver tcpserver.c

udpclient: udpclient.c
    $(CC) $(CFLAGS) -o udpclient udpclient.c

udpserver: udpserver.c
    $(CC) $(CFLAGS) -o udpserver udpserver.c

clean:
    rm -f tcpclient tcpserver udpclient udpserver
```

Задание 2

1. Что делают оба приложения?

Эти приложения реализуют клиент-серверную архитектуру с использованием протоколов TCP и UDP.

2.Что произойдет, если tcpclient отправит сообщение незапущенному серверу?

Запускаем сервер В localhost:10050 и обрабатывать данные в буфере размером 100 байт.

./tcpserver 10050 100

```
@Egor228zorro →/workspaces/os_lab_2019/lab7/src (master) $ ./tcpserver 10050 100
```

В новом терминале запускаем подключение на стороне клиента:

./tcpclient 127.0.0.1 10050 100

```
@Egor228zorro →/workspaces/os_lab_2019/lab7/src (master) $ ./tcpserver 10050 100
connection established
Hello
Good
hey
```

Если мы попробую отправить сообщение незапущенному серверу через tcpclient,то произодет ошибка. Без работающего сервера клиент не сможет отправить сообщение, так как в TCP протоколе необходимо сначала установить соединение.

```
@Egor228zorro →/workspaces/os_lab_2019/lab7/src (master) $ ./tcpclient 127.0.0.1 10050 100
connect: Connection refused
@Egor228zorro →/workspaces/os_lab_2019/lab7/src (master) $
```

3. Что произойдет, если udpclient отправит сообщение незапущенному серверу?

Если запустим сервер и отправим сообщение:

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab7/src (master) $ ./udpserver 20001 1024
SERVER starts...
REQUEST sfsfsf
      FROM 127.0.0.1 : 42313
REQUEST cccc
      FROM 127.0.0.1 : 42313
REQUEST cccc
      FROM 127.0.0.1 : 42313
REQUEST Hello
      FROM 127.0.0.1 : 42313
```

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab7/src (master) $ ./udpclient 127.0.0.1 20001 1024
Enter string
sfsfsf
REPLY FROM SERVER= sfsfsf

cccc
REPLY FROM SERVER= cccc
f

cccc
REPLY FROM SERVER= cccc
f

Hello
REPLY FROM SERVER= Hello
```

Если попробуем отправить сообщение на стороне клиента на незапущенный сервер, со стороны клиента сообщение будет отправлено ,но не доставлено на сам сервер

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab7/src (master) $ ./udpclient 127
.0.0.1 20001 1024
Enter string
Hello
Why?
Egor
20
20001 1024
^C
```

Запускаем сервер(сообщение не доставлено):

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab7/src (master) $ ./udpserver 20001 1024
SERVER starts...
```

UDP (User Datagram Protocol) — это протокол, который не устанавливает соединение между клиентом и сервером, в отличие от TCP.

Когда клиент отправляет сообщение через UDP, он просто "бросает" пакет данных на указанный IP-адрес и порт. Протокол не проверяет, есть ли там сервер, готовый принять сообщение.

Это позволяет избежать накладных расходов на установление и поддержание соединения (как в TCP), но приводит к тому, что доставка данных не гарантируется.

UDP не проверяет, был ли пакет доставлен и обработан. Это сделано для упрощения и ускорения передачи данных, особенно для случаев, где потеря некоторых пакетов

допустима (например, потоковое видео или онлайн-игры).

Почему сервер работает "в неподключенном виде"?

1. Сервер в UDP просто ожидает:

UDP-сервер "слушает" на определенном IP-адресе и порту. Он не "устанавливает связь" с клиентом, как это происходит в TCP.

Сервер принимает данные, только если клиент отправляет их. Если данных нет, сервер просто "сидит и ждет".

2. Зачем это нужно?:

UDP используется в ситуациях, когда нужно минимизировать задержки, и потеря данных не является критичной:

DNS-запросы: Клиент отправляет запрос к серверу DNS и не ждет подтверждения доставки, а просто получает ответ.

Видеостриминг: Если несколько пакетов потеряются, видео продолжит воспроизводиться (возможно, с артефактами), но это лучше, чем ждать повторной отправки.

Онлайн-игры: В играх важно, чтобы данные о действиях игрока отправлялись быстро. Протокол TCP мог бы замедлить процесс из-за своих механизмов подтверждения доставки.

4.Что произойдет, если tcpclient отвалится во время работы с сервером?

Ничего не произойдет. Сервер и клиент обмениваются данными через сокет. Пока сервер не пытается использовать сокет для обмена данными (например, вызовы recv или send), он не узнает, что клиент отключился.

5.Что произойдет, если udpclient отвалится во время работы с сервером?

Ничего не произойдет

Сервер не узнает, что клиент отключился.

UDP не устанавливает соединения, поэтому сервер не отслеживает состояние клиента.

6.Что произойдет, если udpclient отправит сообщение на несуществующий / выключенный сервер?

Сообщение не будет доставлено, поскольку сервер не активен. Программа продолжит работу, и пользователь не получит никаких уведомлений о проблемах с отправкой.

7.Что произойдет,если tcpclient отправит сообщение на несуществующий /выключенный сервер?

Соединение с сервером не установится, поскольку сервер не слушает на указанном порту. Программа выдаст ошибку при попытке подключиться, и выполнение завершится с сообщением об ошибке.

8.В чем отличия UDP и TCP протоколов?

Соединение:

TCP: Ориентированный на соединение. Устанавливается надежное соединение перед передачей данных.

UDP: Без соединения. Данные отправляются без предварительного установления соединения.

Надежность:

TCP: Гарантирует доставку данных, проверяет целостность и порядок. Если данные потеряны, они будут повторно отправлены.

UDP: Не гарантирует доставку, порядок или целостность данных. Нет механизма повторной отправки.

Скорость:

TCP: Более медленный из-за дополнительных проверок и установления соединения.

UDP: Более быстрый, так как не требует установления соединения и дополнительных проверок.

Использование:

TCP: Используется для приложений, требующих надежности, таких как веб-браузеры и электронная почта.

UDP: Используется для приложений, где скорость важнее надежности, таких как потоковое видео и онлайн-игры.