Федеральное агентство по образованию

Государственное образовательное учреждение высшего профессионального образования

Нижегородский государственный архитектурно-

строительный университет (ННГАСУ)

*Факультет инженерно-экологических систем и сооружений  
Кафедра информационных систем и технологий*

КУРСОВАЯ РАБОТА  
по дисциплине: «Инфокоммуникационные системы и сети»

На тему: «Разработка онлайн-игры»

Выполнил студент 3 курса гр.ИС-29: Камышев Д.Д.

Проверил старший преподаватель: Морозов Н.С.

**Содержание**

[**Введение** 3](#_Toc124980806)

[**Задание** 3](#_Toc124980807)

[**Требования к ПО** 4](#_Toc124980808)

[**Теоретическая часть** 5](#_Toc124980809)

[**IP-адресация** 5](#_Toc124980810)

[**Протокол связи (TCP/IP)** 7](#_Toc124980811)

[**Топология сети тестирования** 11](#_Toc124980812)

[**Листинг программы** 12](#_Toc124980813)

[**Серверная часть приложения** 12](#_Toc124980814)

[**Клиентская часть приложения** 13](#_Toc124980815)

[**Выводы** 14](#_Toc124980816)

[**Список литературы** 15](#_Toc124980817)

# **Введение**

## **Задание**

* Изучить систему маршрутизации;
* Изучить систему стека TCP/IP;
* Произвести настройку протокола транспортного уровня;
* Произвести настройку сокетов серверной и клиентской части игры;

## **Требования к ПО**

Для запуска и работы в программе необходимо иметь:

* Установленный Python v2022.20.2 (или выше);
* Операционная система Windows 8 [x64 рекомендуется] (или выше);
* Visual Studio Code 2023 (v1.74.3 или выше);

# **Теоретическая часть**

## **IP-адресация**

**IP-адрес** (от англ. Internet Protocol) - уникальный числовой идентификатор устройства в компьютерной сети, работающей по протоколу IP.  
**IP**- маршрутизируемый протокол сетевого уровня стека TCP/IP. Именно IP стал тем протоколом, который объединил отдельные компьютерные сети во всемирную сеть Интернет. Неотъемлемой частью протокола является адресация сети.

Проще говоря, IP-адрес – это идентификатор, позволяющий передавать информацию между устройствами в сети: он содержит информацию о местоположении устройства и обеспечивает его доступность для связи. IP-адреса позволяют различать компьютеры, маршрутизаторы и веб-сайты в интернете и являются важным компонентом работы интернета.

IP-адрес представляет собой серию из 32 двоичных бит (единиц и нулей). Так как человек невосприимчив к большому однородному ряду чисел, такому как этот 11100010101000100010101110011110 (здесь, к слову, 32 бита информации, так как 32 числа в двоичной системе), было решено разделить ряд на четыре 8-битных байта и получилась следующая последовательность: 11100010.10100010.00101011.10011110. Это не сильно облегчило жизнь и было решение перевести данную последовательность в, привычную нам, последовательность из четырёх чисел в десятичной системе, то есть 226.162.43.158. 4 разряда также называются октетами. Данный IP адрес определяется протоколом IPv4. По такой схеме адресации можно создать более 4 миллиардов IP-адресов.

Логический 32-битный IP-адрес представляет собой иерархическую систему и состоит из двух частей. Первая идентифицирует сеть, вторая — узел в сети. Обе части являются обязательными.

Например, если IP-адрес узла – 170.118.20.57, то первые три октета (170.118.20) представляют собой сетевую часть адреса, а последний октет (.57) является идентификатором узла. Такая система называется иерархической адресацией, поскольку сетевая часть идентифицирует сеть, в которой находятся все уникальные адреса узлов. Маршрутизаторам нужно знать только путь к каждой сети, а не расположение отдельных узлов.

При IP-адресации в одной физической сети могут существовать несколько логических сетей, если сетевая часть адреса их узла отличается. Пример. Три узла в одной физической локальной сети имеют одинаковую сетевую часть в своем IP-адресе (192.168.50), а три других узла — другую сетевую часть (192.168.70). Три узла с одной сетевой частью в своих IP-адресах имеют возможность обмениваться данными друг с другом, но не могут обмениваться информацией с другими узлами без использования маршрутизации. В данном случае имеем одну физическую сеть и две логические IP-сети.

Любая сеть с IP-адресацией построена на основе TCP/IP – модели, включающей в себя стек протоколов, применяемых при передаче данных по сети. Основными протоколами являются TCP и IP, но имеется и масса других вариантов.

При подключении к сети через протокол динамической настройки узла (DHCP / Dynamic Host Configuration Protocol) все параметры стека TCP/IP автоматически устанавливаются на устройстве. Узлу назначается динамический IP-адрес, который меняется на другой при переподключении устройства. Диапазон IP-адресов указывается на сервере DHCP. Иначе дело обстоит со статическим адресом. Статический IP-адрес присваивается вручную и не изменяется при переподключении к сети. Этот тип присваивания используется на устройствах, доступ к которым должен производится по одному адресу (например, на серверах).

## **Протокол связи (TCP/IP)**

**TCP/IP** — сетевая модель передачи данных, представленных в цифровом виде. Модель описывает способ передачи данных от источника информации к получателю. В модели предполагается прохождение информации через четыре уровня, каждый из которых описывается правилом (протоколом передачи). Наборы правил, решающих задачу по передаче данных, составляют стек протоколов передачи данных, на которых базируется Интернет. Название TCP/IP происходит из двух важнейших протоколов семейства — Transmission Control Protocol (TCP) и Internet Protocol (IP), которые были первыми разработаны и описаны в данном стандарте.

Механизм TCP предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым (в отличие от UDP) целостность передаваемых данных и уведомление отправителя о результатах передачи.

Когда осуществляется передача от компьютера к компьютеру через Интернет, TCP работает на верхнем уровне между двумя конечными системами, например, браузером и веб-сервером. TCP осуществляет надёжную передачу потока байтов от одного процесса к другому.

Протокол TCP/IP основан на OSI и так же, как предшественник, имеет несколько уровней, которые и составляют его архитектуру. Всего выделяют 4 уровня – канальный (интерфейсный), межсетевой, транспортный и прикладной.

* Канальный (сетевой интерфейс) Аппаратный уровень обеспечивает взаимодействие сетевого оборудования Ethernet и Wi-Fi. Он соответствует физическому из предыдущего стандарта OSI. Здесь задача состоит в кодировании информации, ее делению на пакеты и отправке по нужному каналу. Также измеряются параметры сигнала вроде задержки ответа и расстояния между хостами.
* Межсетевой (Internet Layer) Интернет состоит из множества локальных сетей, объединенных между собой как раз за счет протокола связи TCP/IP. Межсетевой уровень регламентирует взаимодействие между отдельными подсетями.

Маршрутизация осуществляется путем обращения к определенному IP-адресу с использованием маски. Если хосты находятся в одной подсети, маркируемой одной маской, данные передаются напрямую. В противном случае информация «путешествует» по целой цепочке промежуточных звеньев, пока не достигнет нужной точки. Назначение IP-адреса проводится по стандарту IPv4 или IPv6 (они не совместимы между собой).

* Транспортный уровень (Transport Layer) Следующий уровень отвечает за контроль доставки, чтобы не возникало дублей пакетов данных. В случае обнаружения потерь или ошибок информация запрашивается повторно. Такой подход дает возможность полностью автоматизировать процессы независимо от скорости и качества связи между отдельными участками интернета или внутри конкретной подсети. Протокол TCP отличается большей достоверностью передачи данных по сравнению с тем же UDP, который подходит только для передачи потокового видео и игровой графики. Там некритичны потери части пакетов, чего нельзя сказать о копировании программных файлов и документов. На этом уровне данные не интерпретируются.
* Прикладной уровень (Application Layer) Здесь объединены 3 уровня модели OSI – сеансовый, представления и прикладной. На него ложатся задачи по поддержанию сеанса связи, преобразованию данных, взаимодействию с пользователем и сетью. На этом уровне применяются стандарты интерфейса API, позволяющего передавать команды на выполнение определенных задач. Возможно и использование «производных» протоколов. Например, для открытия сайтов используется HTTPS, при отправке электронной почты – SMTP, для назначения IP-адресов – DHCP. Такой подход упрощает программирование, снижает нагрузку на сеть, увеличивает скорость обработки команд и передачи данных.

Процессы, работающие на прикладном уровне, «общаются» с транспортным, но они видны ему как «черные ящики» с зашифрованной информацией. Зато он понимает, на какой IP-адрес адресованы данные и через какой порт надо их принимать. Этого достаточно для точного распределения пакетов по сети независимо от месторасположения хостов. Порты с 0 до 1023 зарезервированы операционными системами, остальные, в диапазоне от 1024 до 49151, условно свободны и могут использоваться сторонними приложениями.

Комбинация IP-адреса и порта называется сокетом и используется при идентификации компьютера. Если первый критерий уникален для каждого хоста, второй обычно фиксирован для определенного типа приложений. Так, получение электронной почты проходит через 110 порт, передача данных по протоколу FTP – по 21, открытие сайтов – по 80.

Процесс начала сеанса TCP (также называемый «рукопожатие» (англ. handshake)), состоит из трёх шагов.

1. Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN.

* Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буфера и управляющие структуры памяти) для обслуживания нового клиента.
* В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED.
* В случае неудачи сервер посылает клиенту сегмент с флагом RST.

1. Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK.

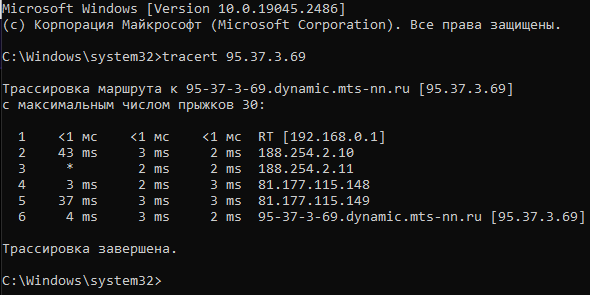
* Если клиент одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED.
* Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться.
* Если клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

1. Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED.

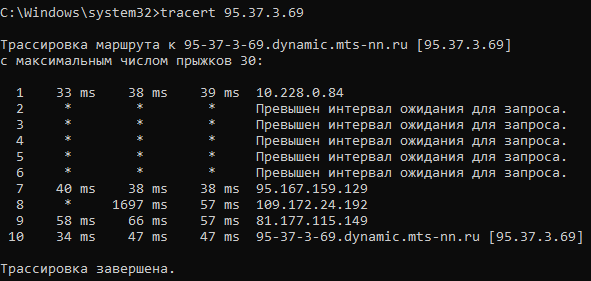
* В противном случае после тайм-аута он закрывает сокет и переходит в состояние CLOSED.

Процесс называется «трёхэтапным рукопожатием» (англ. three way handshake), так как, несмотря на то, что возможен процесс установления соединения с использованием четырёх сегментов (SYN в сторону сервера, ACK в сторону клиента, SYN в сторону клиента, ACK в сторону сервера), на практике для экономии времени используется три сегмента.

## **Топология сети тестирования**



Трассировка при выходе в сеть с использованием домашнего интернета



Трассировка при выходе в сеть с использованием мобильного интернета

# **Листинг программы**

## **Серверная часть приложения**

import socket

import threading

p1 = *dict*()

*def* Game(*choise*, *opp\_choise*):

    if choise == opp\_choise:

        return "Ничья"

    elif choise == 1 and opp\_choise == 2 \

            or choise == 2 and opp\_choise == 3 \

            or choise == 3 and opp\_choise == 1:

        return "Проигрыш"

    else:

        return "Победа"

*def* ConnClient(*server*, *thread\_ID*):

    decision = -2

    server.listen(1)

    conn, addr = server.accept()

    print('Connected:', addr)

    name = *str*(conn.recv(256).decode('utf-8'))  # Клиент после подключения сразу передает свой ник

    while True:

        while decision == -2:

            data = conn.recv(8).decode('utf-8')

            if data:

                decision = *int*(data)

        # данные получены, ник есть, считаем результат

        p1.update({name: decision})

        while len(p1.items()) != 2: # ждём пока данные появятся

            continue

        if *list*(p1.keys())[0] == name:

            me, opp = p1.values()

        else:

            opp, me = p1.values()

        res = Game(me, opp)

        conn.send(res.encode('utf-8')) # отправляем результат клиенту

        decision = -2

        p1.clear()

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

sock.bind(('localhost', 6969))

print("Server is working.")

t1 = threading.Thread(*target*=ConnClient, *args*=(sock, 1))

t2 = threading.Thread(*target*=ConnClient, *args*=(sock, 2))

t1.start()

print('t1 started')

t2.start()

print('t2 started')

t1.join()

print('t1 finished')

t2.join()

print('t2 finished')

p1.clear()

## **Клиентская часть приложения**

import socket

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

sock.connect(('localhost', 6969))

nickname = input('Enter your nickname:')

sock.send(nickname.encode('utf-8'))

while True:

    sock.send(input('Enter your fighter (1 - rock, 2 - paper, 3 - scissors):').encode('utf-8'))

    while True:

        data = sock.recv(256).decode('utf-8')

        if data:

            break

    print(data)

# **Выводы**

В результате выполненной работы была исследована система маршрутизации. Также использованием встроенных в операционную систему Windows 10 команд была произведена трассировка - составлен путь до ключевой, необходимой нам машины.

Была изучена система работы стека TCP/IP, уровни модели и особенности работы вышеуказанных протоколов.

При создании программного обеспечения была реализована связь TCP и произведена настройка протокола транспортного уровня, также настройка сокетов клиентской и серверной части приложения.

# **Список литературы**

1. Wikipedia[Электронный ресурс] URL: https://ru.wikipedia.org/wiki/IP-адрес
2. Kaspersky[Электронный ресурс]URL: <https://www.kaspersky.ru/resource-center/definitions/what-is-an-ip-address>
3. Habr[Электронный ресурс] URL: <https://habr.com/ru/post/350878/>
4. Wikipedia[Электронный ресурс]URL: <https://ru.wikipedia.org/wiki/IP>
5. ITandLife[Электронный ресурс]URL: <https://itandlife.ru/technology>/ /computer-networks/ip-adresaciya-klassy-ip-adresov-i-znachenie-maski-podseti/
6. Eternalhost[Электронный ресурс] URL: <https://eternalhost.net/blog/sistemnoe-administrirovanie/chto-takoe-ip-adres#p3>
7. Wikipedia[Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/TCP/IP>
8. TimeWeb[Электронный ресурс] URL: <https://timeweb.com/ru/community/articles/chto-takoe-tcp-ip>
9. Wikipedia[Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/Transmission_Control_Protocol>
10. «TCP/IP Сертификационный экзамен – экстерном (экзамен 70-059)» Эд Титтел, Курт Хадсон, Джеймс Майкл Стюарт. Издательство «Питер Ком» 1999