МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

**Лабораторная работа №3**

дисциплина «Сети ЭВМ и телекоммуникации»

тема: «**Программирование протокола IP с использованием**

**библиотеки Winsock**»

Выполнил: студент группы ВТ-31

Ковалёв И. Д.

Проверил: Федотов Е. А.

Белгород 2020

**Цель работы:** изучить принципы и характеристику протокола IP и разработать программу для приема/передачи пакетов с использованием библиотеки Winsock.

**Теоретические сведения**

Internet Protocol или IP (англ. internet protocol – межсетевой протокол) – маршрутизируемый сетевой протокол сетевого уровня семейства TCP/IP. Протокол IP используется для негарантированной доставки данных, разделяемых на так называемые пакеты от одного узла сети к другому. Это означает, что на уровне этого протокола (третий уровень сетевой модели OSI) не даётся гарантий надёжной доставки пакета до адресата. В частности, пакеты могут прийти не в том порядке, в котором были отправлены, продублироваться (когда приходят две копии одного пакета - в реальности это бывает крайне редко), оказаться повреждёнными (обычно повреждённые пакеты уничтожаются) или не прибыть вовсе. Гарантии безошибочной доставки пакетов дают протоколы более высокого (транспортного) уровня сетевой модели OSI - например, TCP - который использует IP в качестве транспорта.

В современной сети Интернет используется IP четвёртой версии, также известный как IPv4. В протоколе IP этой версии каждому узлу сети ставится в соответствие IP-адрес длиной 4 октета (8-битных байта). При этом компьютеры в подсетях объединяются общими начальными битами адреса. Количество этих бит, общее для данной подсети, называется маской подсети (ранее использовалось деление пространства адресов по классам — A, B, C; класс сети определялся диапазоном значений старшего октета и определял число адресуемых узлов в данной сети, сейчас используется бесклассовая адресация).

IP-пакет — форматированный блок информации, передаваемый по вычислительной сети. Соединения вычислительных сетей, которые не поддерживают пакеты, такие как традиционные соединения типа «точка-точка» в телекоммуникациях, просто передают данные в виде последовательности байтов, символов или битов. При использовании пакетного форматирования сеть может передавать длинные сообщения более надежно и эффективно.

Структура IP адреса. IP-адрес имеет длину 4 байта и обычно записывается в виде четырех чисел, представляющих значения каждого байта в десятичной форме, и разделенных точками, например:

128.10.2.30 – традиционная десятичная форма представления адреса,

10000000 00001010 00000010 00011110 - двоичная форма представления этого же адреса.

Служебные IP-адреса. Некоторые IP-адреса являются зарезервированными. Для таких адресов существуют следующие соглашения об их особой интерпретации:

1. Если все биты IP-адреса установлены в нуль, то он обозначает адрес данного устройства.

2. Если в поле номера сети стоят нули, то считается, что получатель принадлежит той же самой сети, что и отправитель.

3. Если все биты IP-адреса установлены в единицу, то пакет с таким адресом должен рассылаться всем узлам, находящимся в той же сети, что и отправитель. Такая рассылка называется ограниченным широковещательным сообщением (limitedbroadcast).

4. Если все биты номера узла установлены в нуль, то пакет предназначен для данной сети.

5. Если все биты в поле номера узла установлены в единицу, то пакет рассылается всем узлам сети с данным номером сети. Такая рассылка называется широковещательным сообщением (broadcast).

6. Если первый октет адреса равен 127, то адрес обозначает тот же самый узел. Такой адрес используется для взаимодействия процессов на одной и той же машине (например, для целей тестирования). Этот адрес имеет название возвратного (loopback).

**Задание к работе**

1. Разработать программу “Сервер”, которая принимает запросы от клиентов и посылает им в качестве ответа некоторое сообщение.

2. Разработать программу “Клиент”, которая посылает запрос серверу и “ждет” от него ответного сообщения.

3. Провести анализ функционирования разработанных программ (одновременная работа 2-х, 3-х и т.д. приложений на 2-х, 3-х и т.д. компьютерах ЛВС), сделать выводы.4. Провести сравнительный анализ протоколов IPX и SPX. Сделать выводы.

**Основные функции API, использованные в данной работе:**

**int WSAStartup(WORD wVersionRequired, LPWSADATA lpWSAData);**

**Назначение:** инициализирует Winsock dll. Возвращает 0, если инициализация прошла успешно, иначе – код ошибки.

**Void WSACleanup();**

**Назначение:** завершает работу с Winsock dll. Возвращает 0, если закрытие прошло успешно, иначе – код ошибки.

**SOCKET WSAAPI socket(int af, int type, int protocol);**

**Назначение:** создает сокет.

Af – семейство адресов. Был использован AF\_IPX.

Type – тип создаваемого сокета.

Protocol – используемый протокол.

**bind(s, (sockaddr\*)& clientAdress, sizeof(SOCKADDR\_IPX));**

**Назначение:** связывает локальный адрес с сокетом. В случае успеха возвращает 0, иначе – код ошибки.

S – дескриптор несвязанного сокета.

Name – указатель на структуру sockaddr.

Namelen – размер структуры sockaddr.

**int WSAAPI sendto(SOCKET s, const char \*buf, int len, int flags, const sockaddr \*to, int tolen);**

**Назначение:** отправляет данные data по определенному адресу to\*.

**Анализ протокола IP**

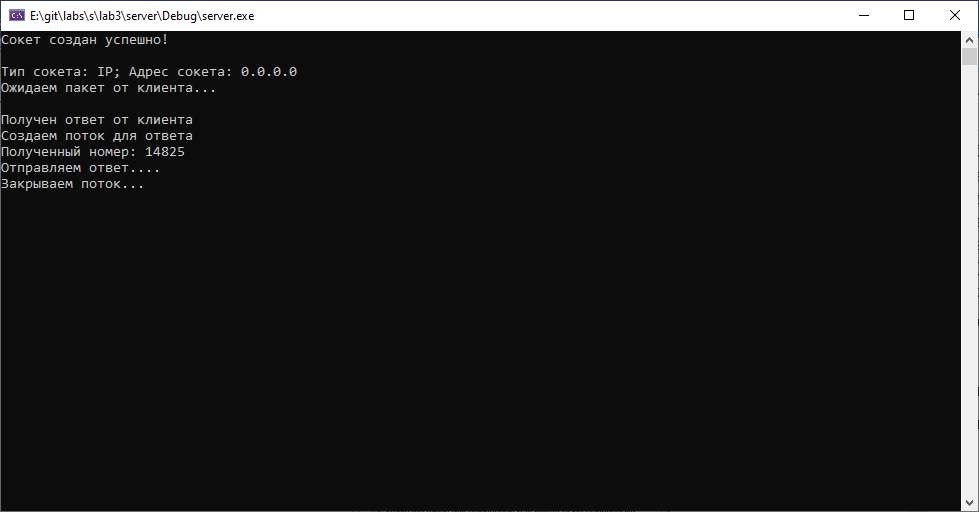
В процессе выполнения работы с сервера отправлялся файл размером 498235 байт.

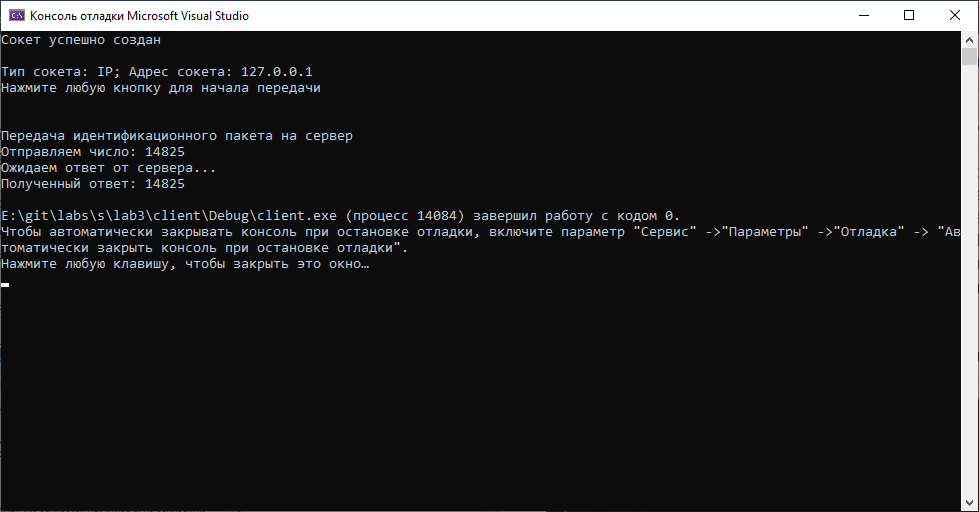
|  |  |  |
| --- | --- | --- |
|  | **Время, сек** | **Принято байт** |
| **1 – 1** | **6,785** | 497912 |
| **2 -2** | **8,28** | 496441 |
| **3 -3** | **11,074** | 487509 |

**Вывод:** В процессе выполнения лабораторной работы были изучены принципы и характеристики протокола IP, а также была разработана программа для приема/передачи пакетов с использованием библиотеки Winsock.

В процессе тестирования разработанных программ, было выявлено, что протокол IP не гарантирует передачу данных без потерь. Также, количество клиентов/серверов влияет на время передачи данных и процент потерь.

**Скриншоты разработанных программ**





**Листинг программ, реализующих получение ответа клиента серверу:**

1. **Client.cpp:**
2. // client.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.
3. //
4. #define \_CRT\_SECURE\_NO\_WARNINGS
5. #include <windows.h>
6. #include <winsock.h>
7. #include <stdlib.h>
8. #include <stdio.h>
9. #include <signal.h>
10. #include <ctime>
11. #include <iostream>
12. #pragma comment (lib, "Ws2\_32.lib")
13. #pragma comment (lib, "Mswsock.lib")
14. #pragma comment (lib, "AdvApi32.lib")
15. void intHandler(int dummy) {
16. exit(0);
17. }
18. int main() {
19. setlocale(LC\_ALL, "Russian");
20. signal(SIGINT, intHandler);
21. time\_t t;
22. srand((unsigned)time(&t));
23. int data = rand();
24. WSADATA wsaData = { 0 };
25. int iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);
26. int sock = socket(AF\_INET, SOCK\_DGRAM, IPPROTO\_IP);
27. if (sock == SOCKET\_ERROR) {
28. perror("Ошибка создания сокета\n");
29. return 1;
30. };
31. printf("Сокет успешно создан\n");
32. struct sockaddr\_in addr;
33. addr.sin\_family = AF\_INET;
34. unsigned short port = 0x4444;
35. addr.sin\_port = htons(port);
36. char address[] = "127.0.0.1";
37. addr.sin\_addr.s\_addr = inet\_addr(address);
38. printf("\nТип сокета: IP; Адрес сокета: %s\n", address);
39. printf("Нажмите любую кнопку для начала передачи\n");
40. getchar();
41. getchar();
42. printf("Передача идентификационного пакета на сервер\nОтправляем число: %d\n", data);
43. int result = sendto(sock, (char\*)(&data), sizeof(int), 0, (SOCKADDR\*)&addr, sizeof(addr));
44. if (result == SOCKET\_ERROR) {
45. printf("\nОтправка не удалась. Код ошибки: %d\n", WSAGetLastError());
46. closesocket(sock);
47. WSACleanup();
48. return 1;
49. }
50. printf("Ожидаем ответ от сервера...\n");
51. int echo;
52. struct sockaddr\_in senderAddress;
53. int senderAddressSize = sizeof(senderAddress);
54. result = recvfrom(sock, (char\*)(&echo), sizeof(int), 0, (SOCKADDR\*)&senderAddress, &senderAddressSize );
55. if (result == SOCKET\_ERROR) {
56. printf("\nПроизошла ошибка при получении ответа с сервера. Код ошибки: %d\n", WSAGetLastError());
57. closesocket(sock);
58. WSACleanup();
59. return 1;
60. };
61. printf("Полученный ответ: %d\n", echo);
62. return 0;
63. }

**2. Server.cpp:**

// server.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <windows.h>

#include <winsock.h>

#include <stdlib.h>

#include <stdio.h>

#include <sys/types.h>

#include <signal.h>

#include <iostream>

#pragma comment (lib, "Ws2\_32.lib")

#pragma comment (lib, "Mswsock.lib")

#pragma comment (lib, "AdvApi32.lib")

typedef struct \_threadArg {

struct sockaddr\_in adr;

int data;

} threadArg;

DWORD WINAPI sendEcho(LPVOID arg) {

printf("\nПолучен ответ от клиента\n");

printf("Создаем поток для ответа\n");

printf("Полученный номер: %d\n", ((threadArg\*)arg)->data);

printf("Отправляем ответ....\n");

int sock = socket(AF\_INET, SOCK\_DGRAM, IPPROTO\_IP);

if (sock == SOCKET\_ERROR) {

perror("Ошибка создания сокета\n");

return 1;

};

int ret\_val = sendto(sock, (char\*)(&((threadArg\*)arg)->data), sizeof(int), 0, (SOCKADDR\*)&((threadArg\*)arg)->adr, sizeof(((threadArg\*)arg)->adr));

if (ret\_val == SOCKET\_ERROR) {

printf("Произошла ошибка при отправке. Код ошибки: %d\n", WSAGetLastError());

closesocket(sock);

WSACleanup();

return 1;

}

printf("Закрываем поток...\n");

closesocket(sock);

return 0;

}

void intHandler(int dummy) {

exit(0);

}

int main() {

setlocale(LC\_ALL, "Russian");

signal(SIGINT, intHandler);

WSADATA wsaData = { 0 };

int iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);

int sock = socket(AF\_INET, SOCK\_DGRAM, IPPROTO\_IP);

if (sock == SOCKET\_ERROR) {

perror("Ошибка создания сокета\n");

return 1;

};

printf("Сокет создан успешно!\n");

struct sockaddr\_in address;

address.sin\_family = AF\_INET;

unsigned short port = 0x4444;

address.sin\_port = htons(port);

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

printf("\nТип сокета: IP; Адрес сокета: %s\n", inet\_ntoa(address.sin\_addr));

if (bind(sock, (SOCKADDR\*)&address, sizeof(address)) < 0) {

return 2;

};

printf("Ожидаем пакет от клиента...\n");

int receivedData;

while (1) {

struct sockaddr\_in senderAddress;

int senderAddressSize = sizeof(senderAddress);

int ret\_val = recvfrom(sock, (char\*)(&receivedData), sizeof(int), 0, (SOCKADDR\*)&senderAddress, &senderAddressSize);

if (ret\_val == SOCKET\_ERROR) {

wprintf(L"Ошибка при получении пакета! Код ошибки: %d\n", WSAGetLastError());

closesocket(sock);

WSACleanup();

return 1;

}

else {

threadArg arg;

arg.data = receivedData;

arg.adr = senderAddress;

HANDLE thread = CreateThread(NULL, 0, sendEcho, &arg, 0, NULL);

};

};

return 0;

}