Міністэрства адукацыі Рэспублікі Беларусь

Установа адукацыі

«БЕЛАРУСКІ ДЗЯРЖАЎНЫ УНІВЕРСІТЭТ
ІНФАРМАТЫКІ І РАДЫЁЭЛЕКТРОНІКІ»

Інжынерна-эканамічны факультэт
Кафедра эканамічнай інфарматыкі

ЛАБАРАТОРНАЯ РАБОТА №2

на тэму

«СТВАРЭННЕ ПАСЛЯДОЎНАГА СЕРВЕРА БЕЗ УСТАЛЯВАННЯ ЛАГІЧНАГА ЗЛУЧЭННЯ UDP»

Падрыхтаваў ст. гр. 210101
Астроўскі Я. А.

Праверыў выкладчык
Карбіт П. А.

Мінск 2024

Мэта работы

Вывучыць метады стварэння сервераў без усталявання лагічнага злучэння *UDP*, выкарыстоўваючы алгарытм паслядоўнай апрацоўкі запросаў.

Заданне

Распрацаваць прылажэнне, якое рэалізуе архітэктуру «кліент-сервер». Неабходна рэалізаваць паслядоўны сервер без устанаўлення лагічнага злучэння (UDP). Логіку ўзаемадзеяння кліента і сервера рэалізаваць наступным чынам: кліент ўводзіць з клавіятуры радок знакаў і пасылае яе серверу. Прыкмета заканчэння ўводу радка-націск клавішы «Увод». Сервер, атрымаўшы гэты радок, павінен вызначыць даўжыню уведзенага радку, і, калі даўжыня кратная 4, то выдаляюцца ўсе лікі, якія дзеляцца на 4. Кліент атрымлівае ператвораны радок і колькасць такіх лікаў.

Серверная частка

```
#include <iostream>
#pragma comment (lib, "ws2_32.lib")
#include <winsock2.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#pragma warning(disable: 4996)

using namespace std;

int main()
{
    WSADATA wsaData;
    WORD wVersionRequested = MAKEWORD(2, 2);
    if (WSAStartup(wVersionRequested, &wsaData) != 0) {
        cout << "Error with loading library" << endl;
        exit(1);
    }

    SOCKADDR_IN ad;
    int sizeooflocal = sizeof(ad);
    ad.sin_addr.s_addr = inet_addr("127.0.0.1");
    ad.sin_port = htons(1228);
    ad.sin_family = AF_INET;

    SOCKET s = socket(AF_INET, SOCK_DGRAM, 0);
    bind(s, (SOCKADDR*)&ad, sizeooflocal);

    char buffer[1024];
```

```
int len;
SOCKADDR_IN client;
int sizeoofclient = sizeof(client);

while (true) {

    len = recvfrom(s, buffer, sizeof(buffer), 0,
 (SOCKADDR*)&client, &sizeoofclient);
    if (len == SOCKET_ERROR) {
        cout << "Error with receiving data" << endl;
        break;
    }
    buffer[len] = '\0';
    cout << "Received string: " << buffer << endl;

    char* readPtr = buffer;
    char* writePtr = buffer;
    int count = 0;

    while (*readPtr) {
        char* endPtr;
        long num = strtol(readPtr, &endPtr, 10);
        if (endPtr != readPtr) {
            if (num % 4 != 0) {
                while (readPtr != endPtr) {
                    *(writePtr++) = *(readPtr++);
                }
            }
            else {
                readPtr = endPtr;
                count++;
            }
        }
        else {
            *(writePtr++) = *(readPtr++);
        }
    }

    *writePtr = '\0';

    buffer[strlen(buffer)] = '\0';

    cout << "Transformed string: " << buffer << endl;
    cout << "Number of deleted numbers: " << count <<
 endl;


    char message[1024];
    sprintf(message, "%s | %d", buffer, count);
    if (sendto(s, message, strlen(message), 0,
 (SOCKADDR*)&client, sizeoofclient) == SOCKET_ERROR) {
```

```
            cout << "Error with sending data" << endl;
            break;
        }
        cout << "Sent message: " << message << endl;
    }

    closesocket(s);
    WSACleanup();
    return 0;
}
```

Кліентская частка

```
#include <iostream>
#pragma comment (lib, "ws2_32.lib")
#include <winsock2.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

#pragma warning(disable: 4996)

using namespace std;

int main()
{
    WSADATA wsaData;
    WORD wVersionRequested = MAKEWORD(2, 2);
    if (WSAStartup(wVersionRequested, &wsaData) != 0) {
      cout << "Error with loading library" << endl;
      exit(1);
    }

    SOCKADDR_IN add;
    add.sin_addr.s_addr = inet_addr("127.0.0.1");
    add.sin_port = htons(1228);
    add.sin_family = AF_INET;

    SOCKET s = socket(AF_INET, SOCK_DGRAM, 0);

    char buffer[1024];
    int len;
    char message[1024];
    SOCKADDR_IN server;
    int sizeofserver = sizeof(server);

    while (true) {
      cout << "Enter a string: ";
      cin.getline(buffer, sizeof(buffer));
      len = strlen(buffer);
```

```
        if (len % 4 != 0) {
            cout << "The string is not divisible by 4" <<
    endl;
            continue;
        }

        if (sendto(s, buffer, len, 0, (SOCKADDR*)&add,
    sizeoofserver) == SOCKET_ERROR) {
            cout << "Error with sending data" << endl;
            break;
        }
        cout << "Sent string: " << buffer << endl;

        len = recvfrom(s, message, sizeof(message), 0,
    (SOCKADDR*)&server, &sizeoofserver);
        if (len == SOCKET_ERROR) {
            cout << "Error with receiving data" << endl;
            break;
        }
        message[len] = '\0';
        cout << "Received message and number of deleted
    numbers: " << message << endl;
    }

    closesocket(s);
    WSACleanup();
    return 0;
}
```