МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №11**

**з дисципліни «Операційні системи»**

**на тему: «Створення серверних і клієнтських програм з**

**використанням сокетів.»**

Виконав:

Студент гр. ПЗ2011

Кулик С.В.

Прийняв:

Андрющенко В.О.

Дніпро, 2023

**Тема:** Створення серверних і клієнтських програм з використанням сокетів.

**Мета:**

- ознайомитися з сокетами в LINUX

- отримати практичні навички програмування серверних і клієнтських застосувань

- отримати практичні навички використання сокетів.

**Короткі теоретичні відомості**

Сокет - це кінцева точка мережевих комунікацій. Він є чимось на зразок "порталу", через який можна відправляти байти в зовнішній світ. Додаток просто пише дані в сокет; їх подальша буферизація, відправка і транспортування здійснюється використовуваним стеком протоколів і мережевою апаратурою. Читання даних з сокета походить аналогічним чином.

У програмі сокет ідентифікується дескриптором - це просто змінна типа int. Програма отримує дескриптор від операційної системи при створенні сокета, а потім передає його сервісам socket API для вказівки сокета, над яким необхідно виконати те або інша дія.

**Завдання**

Розробити серверну та клієнтську програму з використанням сокетів.

**Текст програми**

**Файл server.cpp**

#include <iostream>

#include <unistd.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <string.h>

using namespace std;

int main()

{

// Створення сокету

int serverSocket = socket(AF\_INET, SOCK\_STREAM, 0);

if (serverSocket == -1)

{

perror("Socket creation failed");

return 1;

}

// Задання адреси сервера

struct sockaddr\_in serverAddress;

memset(&serverAddress, 0, sizeof(serverAddress));

serverAddress.sin\_family = AF\_INET;

serverAddress.sin\_addr.s\_addr = INADDR\_ANY; // Всі доступні IP-адреси

serverAddress.sin\_port = htons(8080); // Порт 8080

// Прив'язка сокету до адреси сервера

if (bind(serverSocket, (struct sockaddr\*)&serverAddress, sizeof(serverAddress)) == -1)

{

perror("Bind failed");

return 1;

}

// Прослуховування вхідних з'єднань

if (listen(serverSocket, 5) == -1)

{

perror("Listen failed");

return 1;

}

cout << "Server started. Waiting for incoming connections..." << endl;

while (true)

{

// Приймання з'єднання

struct sockaddr\_in clientAddress;

socklen\_t clientAddressLength = sizeof(clientAddress);

int clientSocket = accept(serverSocket, (struct sockaddr\*)&clientAddress, &clientAddressLength);

if (clientSocket == -1)

{

perror("Accept failed");

return 1;

}

char clientIP[INET\_ADDRSTRLEN];

inet\_ntop(AF\_INET, &(clientAddress.sin\_addr), clientIP, INET\_ADDRSTRLEN);

cout << "Client connected: " << clientIP << ":" << ntohs(clientAddress.sin\_port) << endl;

// Обробка з'єднання з клієнтом

char buffer[1024];

while (true)

{

// Отримання повідомлення від клієнта

int bytesRead = recv(clientSocket, buffer, sizeof(buffer), 0);

if (bytesRead == -1)

{

perror("Receive failed");

break;

}

else if (bytesRead == 0)

{

cout << "Client disconnected" << endl;

break;

}

// Відправка відповіді клієнту

int bytesSent = send(clientSocket, buffer, bytesRead, 0);

if (bytesSent == -1)

{

perror("Send failed");

break;

}

}

// Закриття з'єднання з клієнтом

close(clientSocket);

}

// Закриття серверного сокету

close(serverSocket);

return 0;

}

**Файл client.cpp**

#include <iostream>

#include <unistd.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <string.h>

using namespace std;

int main()

{

// Створення сокету

int clientSocket = socket(AF\_INET, SOCK\_STREAM, 0);

if (clientSocket == -1)

{

perror("Socket creation failed");

return 1;

}

// Задання адреси сервера

struct sockaddr\_in serverAddress;

memset(&serverAddress, 0, sizeof(serverAddress));

serverAddress.sin\_family = AF\_INET;

serverAddress.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); // Адреса сервера (в даному випадку - локальна адреса)

serverAddress.sin\_port = htons(8080); // Порт 8080

// З'єднання з сервером

if (connect(clientSocket, (struct sockaddr\*)&serverAddress, sizeof(serverAddress)) == -1)

{

perror("Connect failed");

return 1;

}

cout << "Connected to the server." << endl;

// Взаємодія з сервером

char message[1024];

while (true)

{

cout << "Enter a message: ";

cin.getline(message, sizeof(message));

// Надсилання повідомлення серверу

int bytesSent = send(clientSocket, message, strlen(message), 0);

if (bytesSent == -1)

{

perror("Send failed");

break;

}

// Отримання відповіді від сервера

char buffer[1024];

int bytesRead = recv(clientSocket, buffer, sizeof(buffer), 0);

if (bytesRead == -1)

{

perror("Receive failed");

break;

}

else if (bytesRead == 0)

{

cout << "Server closed the connection" << endl;

break;

}

// Виведення відповіді сервера

buffer[bytesRead] = '\0';

cout << "Server response: " << buffer << endl;

}

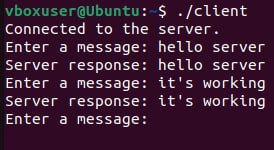
// Закриття сокету клієнта

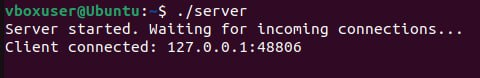
close(clientSocket);

return 0;

}

**Результати програми та тестові прилади**





**Висновки**

В результаті вивчення теми були отримані практичні навички програмування серверних та клієнтських застосувань з використанням сокетів, а також навички використання сокетів для взаємодії між різними процесами в мережі.