Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота №2

Мережеве програмування в середовищі Unix

Тема: Мережеві адреси та імена

Виконав
студент групи ІП-11:
Панченко С. В.

Перевірив:
Сімоненко А.В.

Київ 2025

ЗМІСТ

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Практичне освоєння роботи з мережевими адресами та іменами в середовищі Unix, зокрема використання функцій POSIX для конвертації текстових представлень мережевих адрес та портів у структури адрес сокетів і навпаки.

2 ЗАВДАННЯ

Розробити програму, яка виконує наступне:

1. Отримує текстове представлення мережевої адреси або мережеве ім’я та/або текстове представлення номера порту або ім’я сервісу в аргументах командного рядка. Програма повинна підтримувати аргумент, який дозволяє вказувати тільки текстові представлення мережевих адрес та номерів портів. Також має бути аргумент для вказування версії IP.
2. Конвертує отриманні дані в структури адрес сокетів.
3. Виводить текстове представлення отриманих структур адрес сокетів. Інформацію про протокол треба виводити в числовому значення та іменем. Має бути аргумент командного рядка, який вказує виводити відповідне мережеве ім’я для мережевої адреси, та аргумент, який вказує виводити ім’я сервісу для номера порту.

3 ВИКОНАННЯ

Розглянемо приклади використання:

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 google.com
Address: 216.58.208.206 (par10s21-in-f206.1e100.net)
Port: 0
Protocol: 6 (TCP)
```

Рисунок 3.1 - Базове використання

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 facebook.com 80
Address: 157.240.224.35 (edge-star-mini-shv-01-iev1.facebook.com)
Port: 80
Protocol: 6 (TCP)

(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$
```

Рисунок 3.2 - Адреса з портом

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -6 facebook.com
Address: 2a03:2880:f165:81:face:b00c:0:25de (edge-star-mini6-shv-01-iev1.facebook.com)
Port: 0
Protocol: 6 (TCP)

(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$
```

Рисунок 3.3 - Використання IPv6

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -n google.com
Address: 142.250.203.142
Port: 0
Protocol: 6 (TCP)
```

Рисунок 3.4 — Показати тільки числові адреси

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -s google.com 80
Address: 142.250.203.142 (waw07s06-in-f14.1e100.net)
Port: 80 (http)
Protocol: 6 (TCP)
```

Рисунок 3.5 — Показати імена сервісів для портів

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -6 -n -s google.com 443
Address: 2a00:1450:401b:80e::200e
Port: 443 (https)
Protocol: 6 (TCP)
```

Рисунок 3.6 — Комбінація опцій

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 142.250.203.142
Address: 142.250.203.142 (waw07s06-in-f14.1e100.net)
Port: 0
Protocol: 6 (TCP)
```

Рисунок 3.7 —Використання IP-адреси замість доменного імені

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 142.250.203.142 80
Address: 142.250.203.142 (waw07s06-in-f14.1e100.net)
Port: 80
Protocol: 6 (TCP)
```

Рисунок 3.8 — IP-адреса з портом

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 google.com ftp
Address: 216.58.208.206 (par10s21-in-f14.1e100.net)
Port: 21
Protocol: 6 (TCP)
```

Рисунок 3.9 — Використання імені сервісу замість номера порту

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -6 2a00:1450:401b:800::200e
Address: 2a00:1450:401b:800::200e (waw02s05-in-x0e.1e100.net)
Port: 0
Protocol: 6 (TCP)
```

Рисунок 3.10 — IPv6-адреса

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 -h
Usage: ./lab_1 [options] <address> [port]
Options:
  -4               Use IPv4 (default)
  -6               Use IPv6
  -n               Show numeric addresses only
  -s               Show service names for ports
  -h               Show this help message
```

Рисунок 3.11 — Отримання допомоги

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 localhost
Address: 127.0.0.1 (localhost)
Port: 0
Protocol: 6 (TCP)
```

Рисунок 3.12 — Локальний хост

```
(base) sideshowbobgot@localhost:~/university/network_programming_eight_semester/lab_1/cmake-build-debug$ ./lab_1 nonexistent.domain
getaddrinfo error: Name or service not known
```

Рисунок 3.13 — Спроба отримати інформацію про неіснуючий домен

ДОДАТОК А ТЕКСТИ ПРОГРАМНОГО КОДУ

Тексти програмного коду

(Найменування програми (документа))

Жорсткий диск

(Вид носія даних)

(Обсяг програми (документа), арк.)

Студента групи ІП-11 4 курсу

Панченка С. В

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <unistd.h>

void print_usage(const char* program_name) {
    printf("Usage: %s [options] <address> [port]\n"
           "Options:\n"
           "  -4              Use IPv4 (default)\n"
           "  -6              Use IPv6\n"
           "  -n              Show numeric addresses only\n"
           "  -s              Show service names for ports\n"
           "  -h              Show this help message\n", program_name);
}

int main(int argc, char* argv[]) {
    int use_ipv4 = 1;
    int numeric_only = 0;
    int show_service_names = 0;
    int opt;

    while ((opt = getopt(argc, argv, "46nsh")) != -1) {
        switch (opt) {
            case '4': use_ipv4 = 1; break;
            case '6': use_ipv4 = 0; break;
            case 'n': numeric_only = 1; break;
            case 's': show_service_names = 1; break;
            case 'h':
                print_usage(argv[0]);
```

```
            return 0;
        default:
            fprintf(stderr, "Unknown option: %c\n", optopt);
            print_usage(argv[0]);
            return 1;
    }
}

if (optind >= argc) {
    fprintf(stderr, "Error: Address argument is required.\n");
    print_usage(argv[0]);
    return 1;
}

const char* address = argv[optind];
const char* port = (optind + 1 < argc) ? argv[optind + 1] : NULL;

struct addrinfo hints, *res, *p;
memset(&hints, 0, sizeof(hints));
hints.ai_family = use_ipv4 ? AF_INET : AF_INET6;
hints.ai_socktype = SOCK_STREAM;

int status = getaddrinfo(address, port, &hints, &res);
if (status != 0) {
    fprintf(stderr, "getaddrinfo error: %s\n", gai_strerror(status));
    return 1;
}

for (p = res; p != NULL; p = p->ai_next) {
    void* addr;
    char ipstr[INET6_ADDRSTRLEN];
```

```
        uint16_t port;

        if (p->ai_family == AF_INET) {
            struct sockaddr_in* ipv4 = (struct sockaddr_in*)p->ai_addr;
            addr = &(ipv4->sin_addr);
            port = ntohs(ipv4->sin_port);
        } else {
            struct sockaddr_in6* ipv6 = (struct sockaddr_in6*)p->ai_addr;
            addr = &(ipv6->sin6_addr);
            port = ntohs(ipv6->sin6_port);
        }

        inet_ntop(p->ai_family, addr, ipstr, sizeof(ipstr));

        printf("Address: %s", ipstr);
        if (!numeric_only) {
            char hostname[NI_MAXHOST];
                if (getnameinfo(p->ai_addr, p->ai_addrlen, hostname, NI_MAXHOST,
NULL, 0, NI_NAMEREQD) == 0) {
                printf(" (%s)", hostname);
            }
        }
        printf("\n");

        printf("Port: %d", port);
        if (show_service_names) {
            char servname[NI_MAXSERV];
                    if (getnameinfo(p->ai_addr, p->ai_addrlen, NULL, 0, servname,
NI_MAXSERV, NI_NAMEREQD) == 0) {
                printf(" (%s)", servname);
            }
```

```
        }
        printf("\n");

        printf("Protocol: %d (", p->ai_protocol);
        switch (p->ai_protocol) {
            case IPPROTO_TCP: printf("TCP"); break;
            case IPPROTO_UDP: printf("UDP"); break;
            default: printf("Unknown"); break;
        }
        printf(")\n\n");
    }

    freeaddrinfo(res);
    return 0;
}
```