Bài tập thực hành

Môn: TH Lập Trình Mạng

Chương 3: Mở đầu về Socket

1. MỤC TIÊU

Bài thực hành đưa ra với mục tiêu cho sinh viên làm quen với khái niệm Socket, cũng như thành thạo sử dụng các hàm trong các bộ thư viện liên quan đến lập trình Socket.

2. YÊU CẦU

- Kiến thức cơ bản về lập trình C
- Kiến thức cơ bản về Mạng máy tính
- Máy tính cài đặt Hệ điều hành Linux (khuyến khích distro Ubuntu)

3. BÀI THỰC HÀNH

Bài 1. Giới thiệu về Cấu trúc Địa chỉ Socket

Cấu trúc `sockadd_in` được sử dụng để xử lý các địa chỉ Internet. Sau đây là mô tả cấu trúc:

```
struct sockaddr_in {
    short int sin_family;  // Họ địa chỉ (AF_INET cho IPv4)
    unsigned short int sin_port; // Số cổng (sử dụng htons() để gán)
    struct in_addr sin_addr; // Địa chỉ Internet (cấu trúc in_addr)
    unsigned char sin_zero[8]; // Đệm để làm cho struct có kích thước giống
sockaddr
};
```

Yêu cầu 1: Viết một chương trình khởi tạo cấu trúc `sockadd_in` với một số cổng và địa chỉ IP (ví dụ: “192.168.1.1”).

```
 8      struct sockaddr_in server_addr;
 9      memset(&server_addr, 0, sizeof(server_addr));
10
11      server_addr.sin_family = AF_INET;
12      server_addr.sin_port = htons(8080);
13
14      if (inet_pton(AF_INET, ip_str, &server_addr.sin_addr) <= 0) {
15          printf("Invalid address/ Address not supported\n");
16          return -1;
17      }
```

- *Dòng 8: Khởi tạo cấu trúc **sockaddr_in**.*
- *Dòng 9: Đặt tất cả các giá trị trong cấu trúc bằng 0.*
- *Dòng 11: Đặt giao thức là IPv4 (AF_INET).*
- *Dòng 12: Đặt số cổng là 8080, sử dụng **htons** để chuyển đổi sang dạng big-endian.*
- *Dòng 14: Chuyển đổi địa chỉ IP từ dạng chuỗi sang nhị phân.*

Bài 2. Các hàm Chuyển đổi Địa chỉ

2.1 inet_aton và inet_ntoa:

- `inet_aton:` Chuyển đổi một chuỗi (ví dụ: "192.168.1.1") thành struct in_addr.
- `inet_ntoa:` Chuyển đổi một in_addr thành chuỗi địa chỉ IP dễ đọc.

2.2 inet_pton và inet_ntop:

- `inet_pton:` Chuyển đổi địa chỉ IP dạng văn bản (IPv4/IPv6) sang dạng nhị phân.
- `inet_ntop:` Chuyển đổi địa chỉ IP dạng nhị phân (IPv4/IPv6) về dạng văn bản dễ đọc.

Yêu cầu 2: Viết một chương trình thực hiện:
- Nhập địa chỉ IP dưới dạng chuỗi từ người dùng.
- Sử dụng `inet_pton` để chuyển địa chỉ sang dạng nhị phân.
- Chuyển địa chỉ nhị phân về dạng văn bản dễ đọc bằng `inet_ntop` và in ra.

- *Nhập địa chỉ IP dưới dạng chuỗi từ người dùng.*

```
7       char ip_str[INET_ADDRSTRLEN];
8       printf("Enter IP address (as a string): ");
9       fgets(ip_str, INET_ADDRSTRLEN, stdin);
10      ip_str[strcspn(ip_str, "\n")] = '\0';
```

Dòng 10: Xóa ký tự xuống dòng ("\n") ở cuối chuỗi.

- *Sử dụng `inet_pton` để chuyển địa chỉ sang dạng nhị phân.*

```c
    // Convert the IP address from string to binary form using inet_pton
    if (inet_pton(AF_INET, ip_str, &ip_addr) == 1) {
        // printf("inet_pton: Successfully converted IP address: %s\n", ip_str);
    } else {
        printf("inet_pton: Failed to convert IP address: %s\n", ip_str);
        exit(EXIT_FAILURE);
    }
```

- *Chuyển địa chỉ nhị phân về dạng văn bản dễ đọc bằng inet_ntop và in ra.*

```c
    // Convert the binary IPv4 address back to string form
    if (inet_ntop(AF_INET, &ip_addr, ip_str_converted, INET_ADDRSTRLEN)) {
        printf("inet_ntop: Converted back to string IP address: %s\n", ip_str_converted);
    } else {
        printf("inet_ntop: Failed to convert IP address back to string\n");
        exit(EXIT_FAILURE);
    }
```

Bài 3. I/O với Socket Stream

Hàm `read` và `write`:

- `read`: Đọc dữ liệu từ một mô tả tệp (file descriptor) socket vào bộ đệm.
- `write`: Ghi dữ liệu từ bộ đệm vào file descriptor socket.

Yêu cầu 3:

Tạo một chương trình client – server sử dụng tất cả các hàm đã học ở trên:

- Client đọc địa chỉ IP của server từ người dùng, chuyển đổi bằng `inet_pton`, và gửi thông điệp. **(file *ex3_client.c*)**
    - *Người dùng nhập địa chỉ IP của server*

```c
    // Enter server's IP address
    printf("Enter server's IP address: ");
    fgets(server_ip, sizeof(server_ip), stdin);
    server_ip[strcspn(server_ip, "\n")] = 0;
```

    - *Chuyển đổi địa chỉ bằng **inet_pton***

```c
    // Convert IP addresses: string to binary
    if (inet_pton(AF_INET, server_ip, &serv_addr.sin_addr) <= 0)
    {
        printf("Invalid address or address not supported\n");
        return -1;
    }
```

    - *Gửi thông điệp đến server*

```
47      // Send a message to server
48      write(sock, message, strlen(message));
```

- Server nhận thông điệp, chuyển đổi địa chỉ của client bằng `sock_ntop`, và gửi phản hồi. ***(file ex3_server.c)***
    - *Chuyển đổi địa chỉ của client bằng **sock_ntop** (sau đó in ra màn hình)*

```
54      // Print client's IP address
55      char client_ip[INET_ADDRSTRLEN];
56      inet_ntop(AF_INET, &address.sin_addr, client_ip, INET_ADDRSTRLEN);
57      printf("Client's IP address: %s\n", client_ip);
```

    - *Gửi phản hồi đến client*

```
59      // Response to client
60      write(new_socket, message, strlen(message));
```

Kết quả:

```
phuca@phuca-VirtualBox:~/network/week3$ ./ex3_server
Server listening on port 8080...
Message from client: Dinh Thi Hong Phuc
Client's IP address: 127.0.0.1
```

```
phuca@phuca-VirtualBox:~/network/week3$ ./ex3_client
Enter server's IP address: 127.0.0.1
Enter message: Dinh Thi Hong Phuc
Server response: Message from server
```