

Networks Lab Experiment 3

Singam Sai Bala Subrahmanyam

B180522CS

TCP server code

```
kali [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Feb 7 13:38
kali@kali: ~
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#include<sys/types.h>           //libraries for definitions of socket functions
#include<sys/socket.h>
#include<unistd.h>

#include<netinet/in.h>          //contains structures needed to store addresses
#include<arpa/inet.h>

int main()
{
    // create server socket
    int server_socket = socket(PF_INET, SOCK_STREAM, 0);           //socket(communication domain, communication type, protocol) returns descriptor

    //define server address
    struct sockaddr_in server_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(8080);           //The htons function can be used to convert an IP port number in host byte order to the IP port number in network byte order.
    server_address.sin_addr.s_addr = inet_addr("192.168.2.0");     //The inet_addr() function converts the Internet host address cp from IPv4 numbers-and-dots notation into binary data in network byte order.

    //bind the socket to specified IP and port
    bind(server_socket, (struct sockaddr *) &server_address, sizeof(server_address));

    //listen to connections
    listen(server_socket, 5);

    //accept connections
    int client_socket = accept(server_socket, NULL, NULL);

    //sending msg
    char server_message[256];
    strcpy(server_message, "connected to server\n");
    send(client_socket, server_message, sizeof(server_message), 0);

    //closing socket
    close(server_socket);

    return 0;
}
```

"tcp_server.c" 41L, 1399B

TCP client code

```
kali_1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Feb 7 13:37
kali1@kali1: ~
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#include<sys/types.h>           //libraries for definitions of socket functions
#include<sys/socket.h>
#include<unistd.h>

#include<netinet/in.h>          //contains structures needed to store addresses
#include<arpa/inet.h>

int main()
{
    //create a socket
    int socketid = socket(PF_INET, SOCK_STREAM, 0);           //socket(communication domain, communication type, protocol) returns descriptor

    //specify an address for socket
    struct sockaddr_in server_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(8080);           //The htons function can be used to convert an IP port number in host byte order to the IP port number in network byte order.
    server_address.sin_addr.s_addr = inet_addr("192.168.2.0");     //The inet_addr() function converts the Internet host address cp from IPv4 numbers-and-dots notation into binary data in network byte order.

    //Establish connection using connect()
    int connect_status = connect(socketid, (struct sockaddr *) &server_address, sizeof(server_address));
    if(connect_status<0)
        printf("connection failed\n");

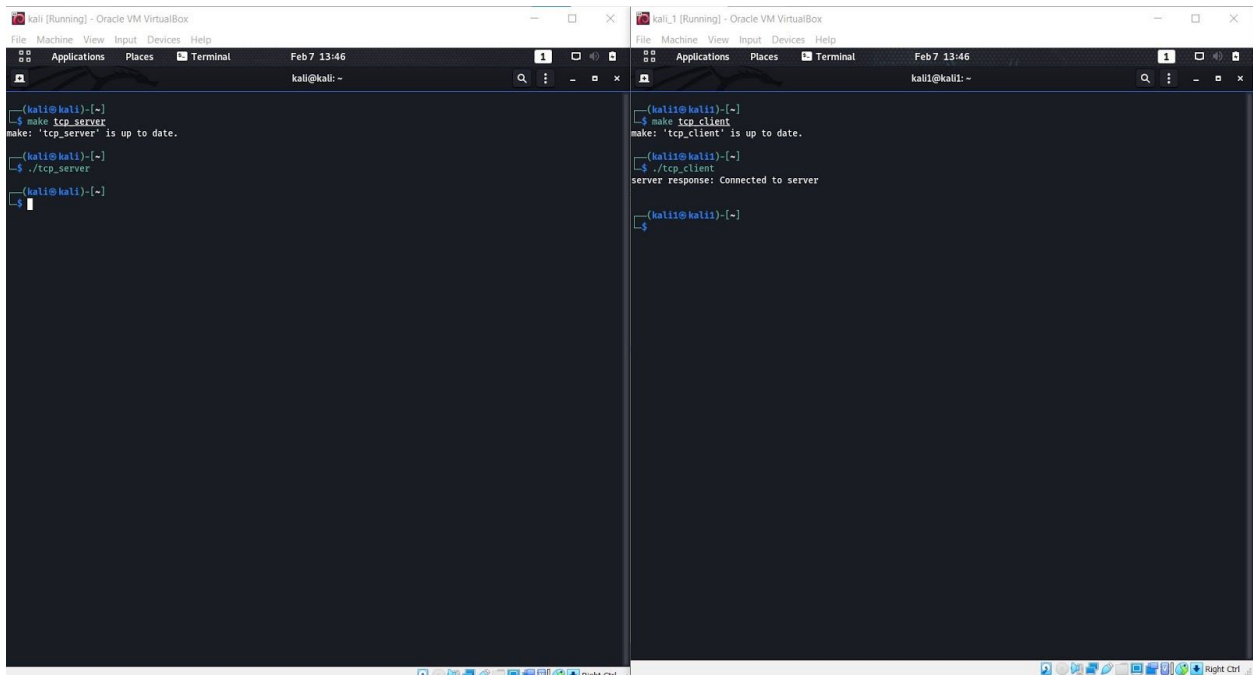
    //receiving data
    char server_response[256];
    recv(socketid, server_response, sizeof(server_response), 0);
    printf("server response: %s\n", server_response);

    //close the socket
    close(socketid);

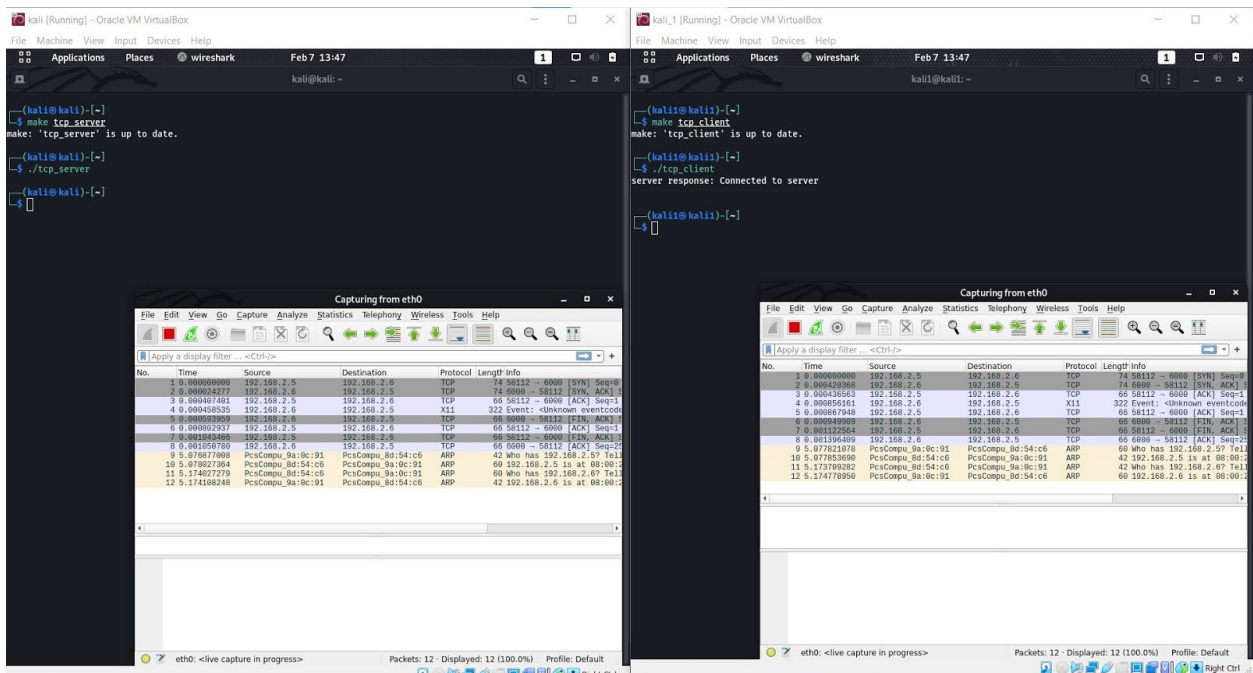
    return 0;
}
```

"tcp_client.c" 38L, 1360B

TCP communication



TCP wireshark capture during communication



UDP server code

```
kali [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Terminal
Feb 7 13:36
kali@kali: ~
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#include<sys/types.h> //libraries for definitions of socket functions
#include<sys/socket.h>
#include<unistd.h>

#include<netinet/in.h> //contains structures needed to store addresses
#include<arpa/inet.h>

int main()
{
    // create server socket
    int server_socket = socket(PF_INET, SOCK_DGRAM, 0); //socket(communication domain, communication type, protocol) returns descriptor

    //define server address
    struct sockaddr_in server_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(451); //The htons function can be used to convert an IP port number in host byte order to the IP port number in network byte order.
    server_address.sin_addr.s_addr = inet_addr("192.168.2.4"); //The inet_addr() function converts the Internet host address cp from IPv4 numbers-and-dots notation into binary data in network byte order.

    //bind the socket to specified IP and port
    bind(server_socket, (struct sockaddr *) &server_address, sizeof(server_address));

    //store client address
    struct sockaddr_storage client_address;
    socklen_t addr_size = sizeof(client_address);

    //receiving msg
    char client_message[256];
    recvfrom(server_socket, client_message, sizeof(client_message), 0, (struct sockaddr *) &client_address, &addr_size);
    printf("client message: %s", client_message);

    //sending msg
    char server_message[256];
    printf("server response:\n");
    fgets(server_message, 256, stdin);
    sendto(server_socket, server_message, sizeof(server_message), 0, (struct sockaddr *) &client_address, addr_size);

    //closing socket
    close(server_socket);

    return 0;
}

"udp_server.c" 451, 16528
38,32-40 All
```

UDP client code

```
kali_1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Terminal
Feb 7 13:36
kali1@kali1: ~
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#include<sys/types.h> //libraries for definitions of socket functions
#include<sys/socket.h>
#include<unistd.h>

#include<netinet/in.h> //contains structures needed to store addresses
#include<arpa/inet.h>

int main()
{
    //create a socket
    int socketid = socket(PF_INET, SOCK_DGRAM, 0); //socket(communication domain, communication type, protocol) returns descriptor

    //specify an address for socket
    struct sockaddr_in server_address;
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(451); //The htons function is used to convert an IP port number in host byte order to the IP port number in network byte order.
    server_address.sin_addr.s_addr = inet_addr("192.168.2.4"); //The inet_addr() function converts the Internet host address cp from IPv4 numbers-and-dots notation into binary data in network byte order.

    //sending data
    char client_msg[256];
    printf("enter your message:\n");
    fgets(client_msg, 256, stdin);
    sendto(socketid, client_msg, sizeof(client_msg), 0, (struct sockaddr *) &server_address, sizeof(server_address));

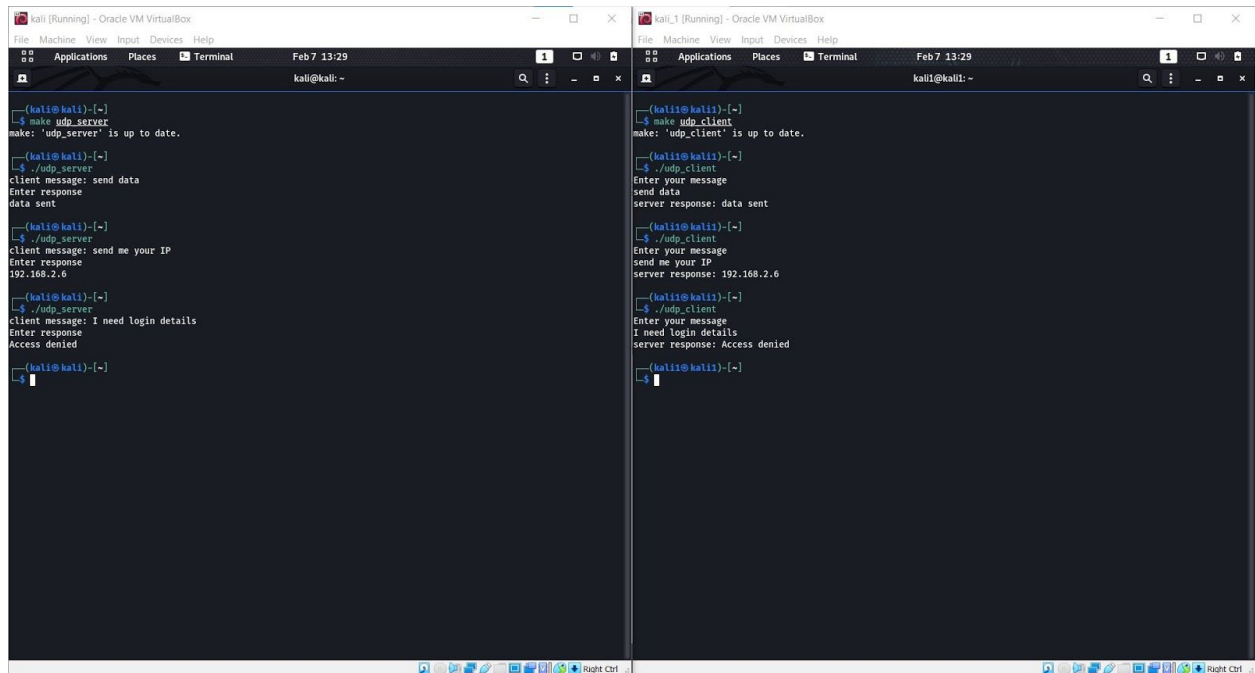
    //receiving data
    char server_response[256];
    recvfrom(socketid, server_response, sizeof(server_response), 0, NULL, NULL);
    printf("server response: %s", server_response);

    //close the socket
    close(socketid);

    return 0;
}

"udp_client.c" 391, 13928
26,29-36 All
```

UDP communication



UDP wireshark capture during communication

