

POST LAB

Server-Side Implementation

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <unistd.h>
#include <arpa/inet.h>

int main() {
    int server_socket;
    server_socket = socket(AF_INET, SOCK_DGRAM, 0);
    socklen_t addr_len = sizeof(struct sockaddr_in);

    struct sockaddr_in my_addr, client_addr;
    my_addr.sin_family = AF_INET;
    my_addr.sin_port = htons(8085);
    my_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

    int success = bind(server_socket, (struct sockaddr*)&my_addr,
sizeof(my_addr));
    if(success == 0){
        printf("%s\n", "Successfully Binded");
    }
    else if(success == -1){
        printf("%s\n", "Not Successfully Binded");
        return -1;
    }
}
```

```

int i = 0;
while (i < 2) {

    char *msg = "connected";

    char msg1[100];
    int success = recvfrom(server_socket, msg1, 100, 0, (struct sockaddr*)&client_addr, &addr_len);

    printf("%s\n", msg1);
    ++i;
}

return 0;
}

```

Client-Side Implementation

```

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <unistd.h>
#include <arpa/inet.h>

int main() {
    int network_socket;
    network_socket = socket(AF_INET, SOCK_DGRAM, 0);

    socklen_t addr_len = sizeof(struct sockaddr_in);

```

```

struct sockaddr_in server_addr;
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(8085);
server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

char *msg1 = "Arshaq Waseem";

int a = sendto(network_socket, msg1, 100, 0, (struct
sockaddr*)&server_addr, addr_len);

if(a == -1){
    printf("%s\n", "Not Sent");
    return -1;
}
else{
    printf("%s\n", msg1);
    printf("%s\n", "Sent");
}

close(network_socket); // Close the socket when done

return 0;
}

```

Output:

The screenshot shows a dual-terminal session on an Ubuntu 12.04 LTS VM running in VMware Workstation. The left terminal window displays the source code for a client application (client.c) using the gedit text editor. The right terminal window shows the execution of the client and server programs, demonstrating successful communication.

Left Terminal (gedit):

```
haider@ubuntu:~/Desktop$ gcc client.c -o client
haider@ubuntu:~/Desktop$ ./client
Arshaq Waseem
Sent
```

Right Terminal:

```
haider@ubuntu:~/Desktop$ gcc server.c -o server
haider@ubuntu:~/Desktop$ ./server
Successfully Bound
Arshaq Waseem
Sent
haider@ubuntu:~/Desktop$
```

The status bar at the bottom indicates the current tab width is 4, the current line is 18, and the current column is 38. A note at the bottom left says "To direct input to this VM, move the mouse pointer inside or press Ctrl+G."