Madhurya Mozumder

RA1911028010036

CSE-CC

I2

Experiment 4 UDP Echo Client Server Communication

## Aim:

To implement an UDP Echo Client-Server application, where the Client on establishing a connection with the Server, sends a string to the Server. The Server reads the String, prints it and echoes it back to the Client

## Procedure:

* Create a folder
* Create two files named with server and client
* Write codes on it
* Now select file for terminal
* Check whether the file is present or not using ls command for reference
* Now put command as cc updclient.c –o udpclient for client
* Now put command as gcc udpserver.c –o udpserver for server
* Now you can able to communicate with each other.

## Server

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

#define PORT 8080

#define MAXLINE 1024

// Driver code int main() {

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from server"; struct sockaddr\_in servaddr, cliaddr;

// Creating socket file descriptor

if ( (sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 ) { perror("socket creation failed");

exit(EXIT\_FAILURE);

}

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

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

// Filling server information servaddr.sin\_family = AF\_INET; // IPv4 servaddr.sin\_addr.s\_addr = INADDR\_ANY; servaddr.sin\_port = htons(PORT);

// Bind the socket with the server address

if ( bind(sockfd, (const struct sockaddr \*)&servaddr, sizeof(servaddr)) < 0 )

{

perror("bind failed"); exit(EXIT\_FAILURE);

}

int len, n;

len = sizeof(cliaddr); //len is value/resuslt

n = recvfrom(sockfd, (char \*)buffer, MAXLINE, MSG\_WAITALL, ( struct sockaddr \*) &cliaddr, &len);

buffer[n] = '\0';

printf("Client : %s\n", buffer);

sendto(sockfd, (const char \*)hello, strlen(hello), MSG\_CONFIRM, (const struct sockaddr \*) &cliaddr,

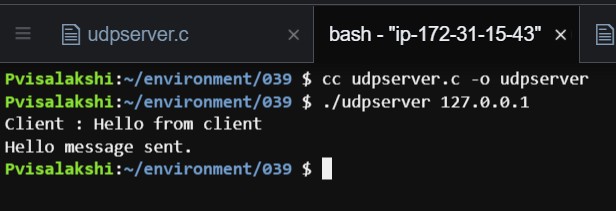
len);

printf("Hello message sent.\n");

return 0;

}

Output:



## Client

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

#define PORT 8080

#define MAXLINE 1024

// Driver code int main() {

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from client"; struct sockaddr\_in servaddr;

// Creating socket file descriptor

if ( (sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 ) { perror("socket creation failed");

exit(EXIT\_FAILURE);

}

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

// Filling server information servaddr.sin\_family = AF\_INET; servaddr.sin\_port = htons(PORT);

servaddr.sin\_addr.s\_addr = INADDR\_ANY; int n, len;

sendto(sockfd, (const char \*)hello, strlen(hello), MSG\_CONFIRM, (const struct sockaddr \*) &servaddr,

sizeof(servaddr)); printf("Hello message sent.\n");

n = recvfrom(sockfd, (char \*)buffer, MAXLINE, MSG\_WAITALL, (struct sockaddr \*) &servaddr, &len);

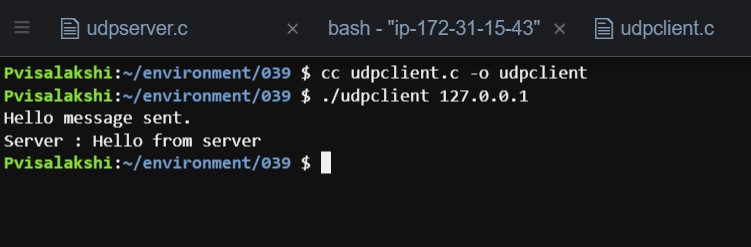
buffer[n] = '\0';

printf("Server : %s\n", buffer);

close(sockfd); return 0;

}

Output:



## Result:

Thus, the UDP ECHO client server communication is established by sending the message from the client to the server and server prints it and echoes the message back to the client.