

Computer Networking

Subham Sagar Paira 1841017020

CSIT A

19th November 2020

EXPERIMENT 3

Objective 1

To implement echo server-client using UDP sockets.

Code:

server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<unistd.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include<arpa/inet.h>
#define MAXLINE 1024
int main(int argc,char **argv)
{
    int sockfd;
    int n;
    socklen_t len;
    char msg[1024];
    struct sockaddr_in servaddr,cliaddr;
    sockfd=socket(AF_INET,SOCK_DGRAM,0);

    //bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=INADDR_ANY;
    servaddr.sin_port=htons(5035);
    printf("\n\n Binded");
    bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

    printf("\n\n Listening...");
    printf("\n ");
    len=sizeof(cliaddr);
    n=recvfrom(sockfd,msg,MAXLINE,0,(struct sockaddr*)&cliaddr,&len);

    printf("\n Client's Message : %s\n",msg);
    if(n<6)
        perror("send error");
    sendto(sockfd,msg,n,0,(struct sockaddr*)&cliaddr,len);

    return 0;
}
```

client.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<arpa/inet.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdio.h>
#define MAXLINE 1024
int main(int argc,char* argv[])
{
    int sockfd;
    int n;
    socklen_t len;
    char sendline[1024],recvline[1024];
    struct sockaddr_in servaddr;
    strcpy(sendline,"");

    printf("\n Enter the message : ");
    //scanf("%s",sendline);
    fgets(sendline,1024,stdin);
    sockfd=socket(AF_INET,SOCK_DGRAM,0);

    //bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
    servaddr.sin_port=htons(5035);

    //connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
    len=sizeof(servaddr);
    sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr*)&servaddr,len);
    n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);
    recvline[n]=0;

    printf("\n Server's Echo : %s\n\n",recvline);

    return 0;
}
```

Output

```

friday BR master ... udp echo_client_server client la
friday BR master ... udp echo_client_server client gcc client.c -o client
friday BR master ... udp echo_client_server client ./client

Enter the message : Hey,Server!_This_is_client.
Server's Echo : Hey,Server!_This_is_client.

friday BR master ... udp echo_client_server client _

friday BR master ... udp echo_client_server server la
friday BR master ... udp echo_client_server server gcc server.c -o server
friday BR master ... udp echo_client_server server ./server

Binded
Listening...
Client's Message : Hey,Server!_This_is_client.

friday BR master ... udp echo_client_server server _

```

Objective 2

To implement a UDP program where client sends a string to the server. Server reverse the string and display at the client end.

Code

server.c

```

#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<unistd.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include<arpa/inet.h>
#define MAXLINE 1024

void strrev(char *s)
{
    int length, c;
    char *begin, *end, temp;
    length = strlen(s);
    begin = s;
    end = s;

    for(c = 0; c < length - 1; c++)
        end++;

    for (c = 0; c < length/2; c++)
    {
        temp = *end;
        *end = *begin;
        *begin = temp;
        begin++;
        end--;
    }
}

int main(int argc, char **argv)
{
    int sockfd;
    int n;

```

```

socklen_t len;
char msg[1024];
struct sockaddr_in servaddr,cliaddr;
sockfd=socket(AF_INET,SOCK_DGRAM,0);

//bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=INADDR_ANY;
servaddr.sin_port=htons(5035);
printf("\n\n Binded");
bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

printf("\n\n Listening...");
printf("\n ");
len=sizeof(cliaddr);
n=recvfrom(sockfd,msg,MAXLINE,0,(struct sockaddr*)&cliaddr,&len);

printf("\n Client's Message : %s\n",msg);
printf("\n Reversing the string...\n");
strrev(msg);
if(n<6)
perror("send error");
sendto(sockfd,msg,n,0,(struct sockaddr*)&cliaddr,len);
printf(" Sent\n\n");
return 0;
}

```

client.c

```

#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<arpa/inet.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdio.h>
#define MAXLINE 1024
int main(int argc,char* argv[])
{
    int sockfd;
    int n;
    socklen_t len;
    char sendline[1024],recvline[1024];
    struct sockaddr_in servaddr;
    strcpy(sendline,"");

    printf("\n Enter the message : ");
    //scanf("%s",sendline);
    fgets(sendline,1024,stdin);
    sockfd=socket(AF_INET,SOCK_DGRAM,0);

```

```

//bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=htons(5035);

//connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
len=sizeof(servaddr);
sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr*)&servaddr,len);
n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);
recvline[n]=0;

printf("\n Server's Response : %s\n\n",recvline);

return 0;
}

```

Output

The screenshot shows two terminal windows side-by-side. The left window, titled 'friday', is running the server program. It shows the compilation of 'server.c' and the execution of the server. The server outputs 'Binded', 'Listening...', and 'Client's Message : hello server'. It then reverses the string and sends the response 'revres olleh'. The right window, also titled 'friday', is running the client program. It shows the compilation of 'client.c' and the execution of the client. The client prompts 'Enter the message : ' and the user enters 'hello server'. The client then receives the response 'Server's Response : revres olleh'.

```

... | udp | reverse_string_server | server | gcc
server.c -o server
... | udp | reverse_string_server | server | ./se
rver

Binded
Listening...
Client's Message : hello server

Reversing the string...
Sent
... | udp | reverse_string_server | server | _

***

```

Objective 3

To implement a UDP client-server program to check if the given string is a pallindrome or not.

Code

server.c

```

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

```

```
#define MAXLINE 1024
```

```
int checkPallindrome(char* string)
```

```
{
    char *ptr, *rev;
    ptr = string;

    while (*ptr != '\0') {
        ++ptr;
    }
    --ptr;

    for (rev = string; ptr >= rev;) {
        if (*ptr == *rev) {
            --ptr;
            rev++;
        }
        else
            break;
    }

    if (rev > ptr)
        return 1;
    else
        return 0;
}
```

```
int main(int argc, char **argv)
```

```
{
    int sockfd;
    int n;
    socklen_t len;
    char msg[1024];
    struct sockaddr_in servaddr, cliaddr;
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);

    //bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(5035);
    printf("\n\n Binded");
    bind(sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr));

    printf("\n\n Listening...");
    printf("\n ");
    len = sizeof(cliaddr);
    n = recvfrom(sockfd, msg, MAXLINE, 0, (struct sockaddr*)&cliaddr, &len);

    printf("\n Client's Message : %s\n", msg);
    printf("\n Checking for Pallindrome...");

    msg[strlen(msg)-1] = '\0';
}
```

```

    int temp = checkPallindrome(msg);
    char res[1024];
    if(temp)
        strcpy(res, "String is pallindrome.");
    else
        strcpy(res, "String is not pallindrome.");

    if(n<6)
        perror("send error");
    sendto(sockfd,res,n,0,(struct sockaddr*)&cliaddr,len);
    printf(" Sent\n\n");
    return 0;
}

```

client.c

```

#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<arpa/inet.h>
#include<string.h>
#include<arpa/inet.h>
#include<stdio.h>
#define MAXLINE 1024
int main(int argc,char* argv[])
{
    int sockfd;
    int n;
    socklen_t len;
    char sendline[1024],recvline[1024];
    struct sockaddr_in servaddr;
    strcpy(sendline,"");

    printf("\n Enter the message : ");
    //scanf("%s",sendline);
    fgets(sendline,1024,stdin);
    sockfd=socket(AF_INET,SOCK_DGRAM,0);

    //bzero(&servaddr,sizeof(servaddr));
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
    servaddr.sin_port=htons(5035);

    //connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
    len=sizeof(servaddr);
    sendto(sockfd,sendline,MAXLINE,0,(struct sockaddr*)&servaddr,len);
    n=recvfrom(sockfd,recvline,MAXLINE,0,NULL,NULL);
    recvline[n]=0;
}

```

```

printf("\n Server's Response : %s\n\n",recvline);

return 0;
}

```

Output

```

udp | check_pallindrome_string_server | server
gcc server.c -o serverme_string_server | server
./serverheck_pallindrome_string_server | server

Binded
Listening...
Client's Message : malayalam

Checking for Pallindrome... Sent
udp | check_pallindrome_string_server | server

udp | check_pallindrome_string_server | client
gcc client.c -o clientme_string_server | client
./client me_string_server | client

Enter the message : malayalam

Server's Response : String is pallindrome.
udp | check_pallindrome_string_server | client

```

Source: Subham Sagar Paira

https://github.com/subhamsagar524/Computer-Networking/tree/master/server_client/udp

Submitted by-

Subham Sagar Paira
1841017020
CSIT A

**** End of Document ****