ASSIGNMENT-6

Name: Sayantan Roy

Roll Number: GCECTB-R18-3025

Subject: Computer Network Lab

Dept: CSE

Year:3rd

ASSIGNMENT -6

UDP TIME CLIENT SERVER PROGRAM IN C

UDP DATETIME CLIENT SERVER PROGRAM

SYSTEM USED :

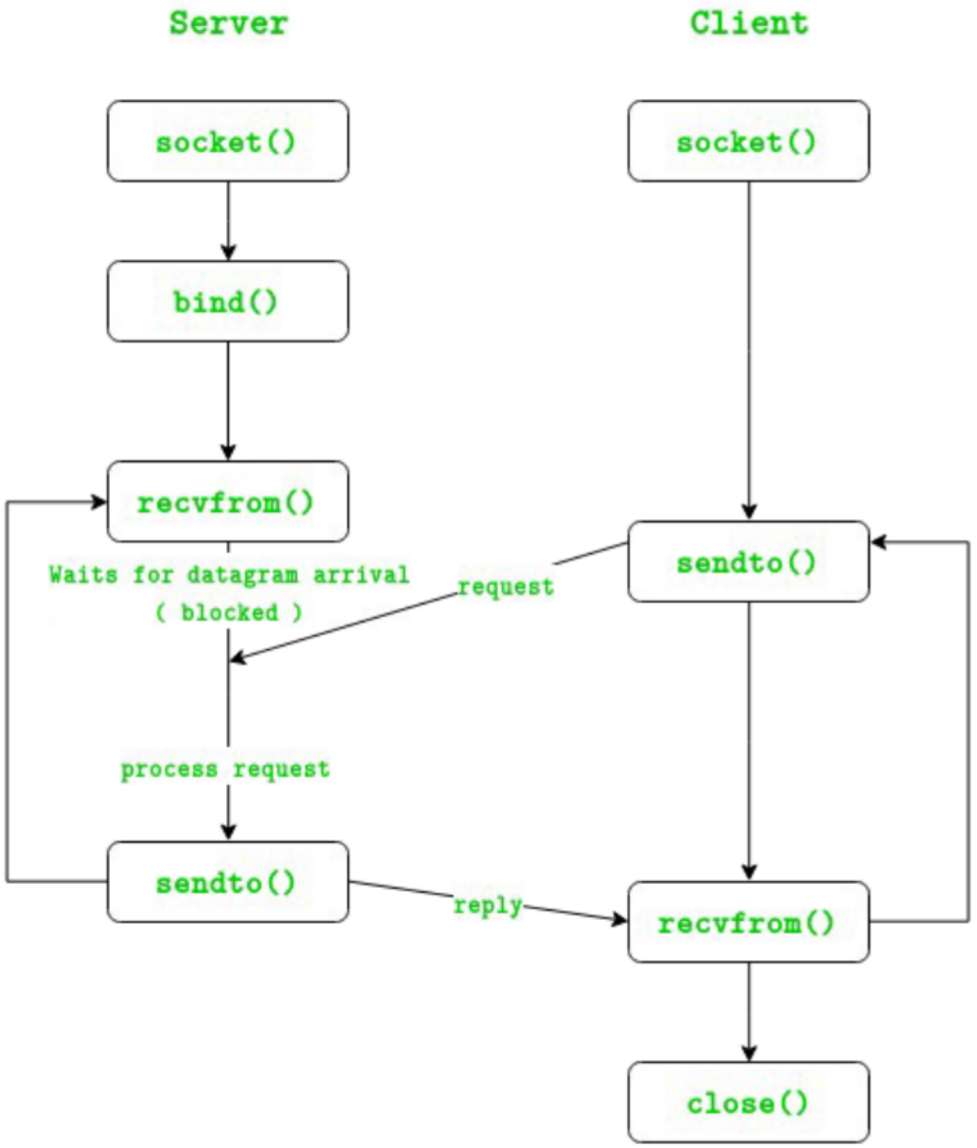
OPERATING SYSTEM : kali linux 2020.1
KERNEL: 4.19.153
GUI: DEFAULT XSESSION (XFCE4)

system owned and project done by :Sayantan Roy

UDP DATETIME CLIENT SERVER PROGRAM

Theory:

In UDP, the client does not form a connection with the server like in TCP and instead just sends a datagram. Similarly, the server need not accept a connection and just waits for datagrams to arrive. Datagrams upon arrival contain the address of sender which the server uses to send data to the correct client.



Arguments :

domain –Specifies the communication
domain (AF_INET for IPv4/ AF_INET6 for IPv6)

type –Type of socket to be created
(SOCK_STREAM for TCP / SOCK_DGRAM for UDP)

protocol –Protocol to be used by socket.
0 means use default protocol for the address family.

bind : assigns address to the unbound socket

sockfd –File descriptor of socket.

addr –Structure in which address to be binded to is specified

Functions:

socket(int domain, int type, int protocol)

Creates an unbound socket in the specified domain.
Returns socket file descriptor.

SERVERSIDE PROGRAM:

Filename:udpdatetimeserver.c

```
#include <sys/socket.h>

#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>

#define S_PORT 43454
#define C_PORT 43455
#define ERROR -1
#define IP_STR "127.0.0.1"

int main(int argc, char const *argv[]) {
int sfd, num;
time_t current_time;
struct sockaddr_in servaddr, clientaddr;
sfd = socket(AF_INET, SOCK_DGRAM,IPPROTO_UDP);
if (sfd == ERROR) {
perror("Could not open a socket");
```

```
return 1;

}

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

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=htonl(INADDR_ANY);

servaddr.sin_port=htons(S_PORT);


memset((char *) &clientaddr, 0, sizeof(clientaddr));

clientaddr.sin_family=AF_INET;

clientaddr.sin_addr.s_addr=inet_addr(IP_STR);

clientaddr.sin_port=htons(C_PORT);


if((bind(sfd,(struct sockaddr *)&servaddr,sizeof(servaddr)))!=0) {

perror("Could not bind socket");

return 2;

}


printf("Server is running on %s:%d\n", IP_STR, S_PORT);

while(1) {

recvfrom(sfd, &num, sizeof(num), 0, (struct sockaddr *)&clientaddr, (socklen_t *)&clientaddr);

current_time = time(NULL);

printf("Client at %s:%d asked for time: %s\n", inet_ntoa(clientaddr.sin_addr),
ntohs(clientaddr.sin_port), ctime(&current_time));

sendto(sfd, &current_time, sizeof(current_time), 0, (struct sockaddr *)&clientaddr,
sizeof(clientaddr));

}
```

return 0;

}

CLIENT SIDE PROGRAM :

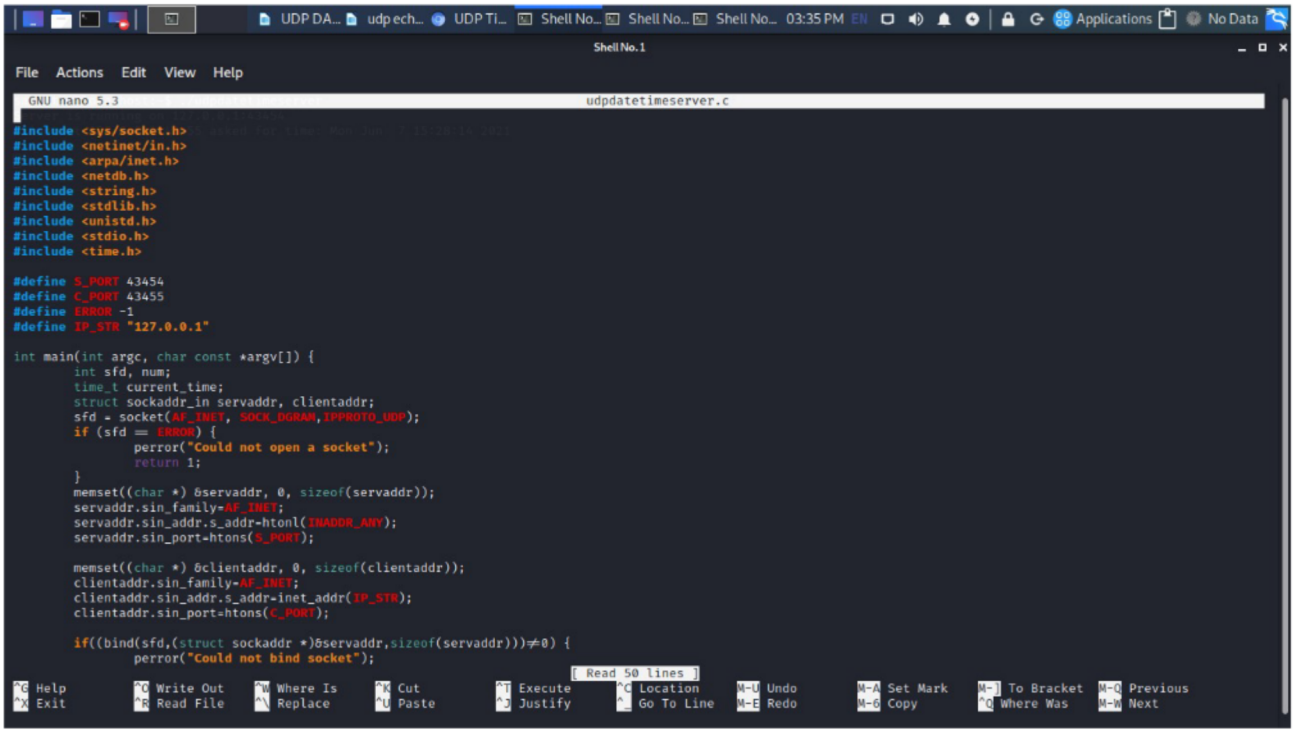
FILENAME:udpdatetimeclient.c

```
#include <sys/socket.h>
#include <netdb.h>
#include <string.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <time.h>

#define S_PORT 43454
#define C_PORT 43455
#define ERROR -1
#define IP_STR "127.0.0.1"

int main(int argc, char const *argv[]) {
	int sfd;
	int num = 1;
	time_t start_time, rtt, current_time;
	struct sockaddr_in servaddr, clientaddr;
	socklen_t addrlen;
	sfd = socket(AF_INET, SOCK_DGRAM,IPPROTO_UDP);
	if (sfd == ERROR) {
		perror("Could not open a socket");
		return 1;
	}
	memset((char *) &servaddr, 0, sizeof(servaddr));
	servaddr.sin_family=AF_INET;
	servaddr.sin_addr.s_addr=inet_addr(IP_STR);
	servaddr.sin_port=htons(S_PORT);

	memset((char *) &clientaddr, 0, sizeof(clientaddr));
	clientaddr.sin_family=AF_INET;
	clientaddr.sin_addr.s_addr=inet_addr(IP_STR);
	clientaddr.sin_port=htons(C_PORT);

	if((bind(sfd,(struct sockaddr *)&clientaddr,sizeof(clientaddr)))!=0) {
		perror("Could not bind socket");
		return 2;
	}
```

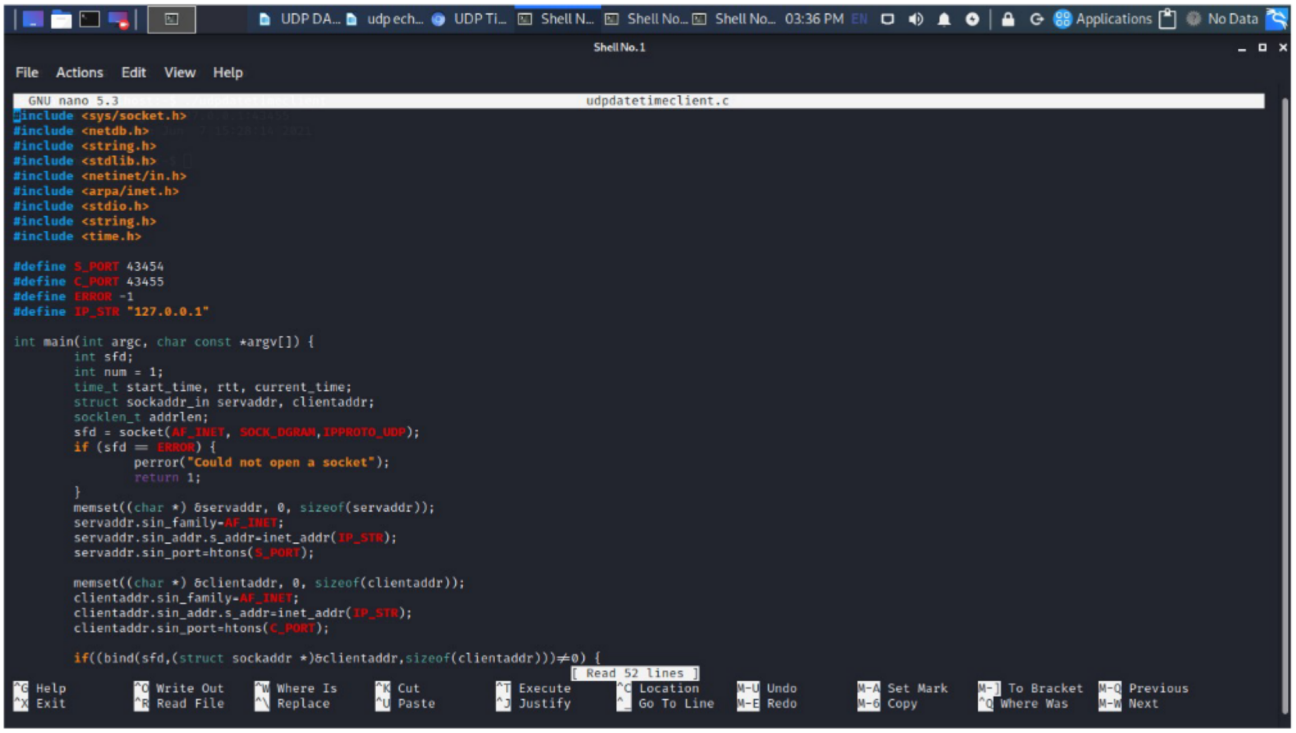
```
        printf("Client is running on %s:%d\n", IP_STR, C_PORT);
        start_time = time(NULL);
        sendto(sfd, &num, sizeof(num), 0, (struct sockaddr *)&servaddr, sizeof(servaddr));
        addrlen = sizeof(clientaddr);
        recvfrom(sfd, &current_time, sizeof(current_time), 0, (struct sockaddr *)&clientaddr,
&addrlen);
        rtt = time(NULL) - start_time;
        current_time += rtt / 2;
        printf("Server's Time: %s\n", ctime(&current_time));

        return 0;
}
```

output:

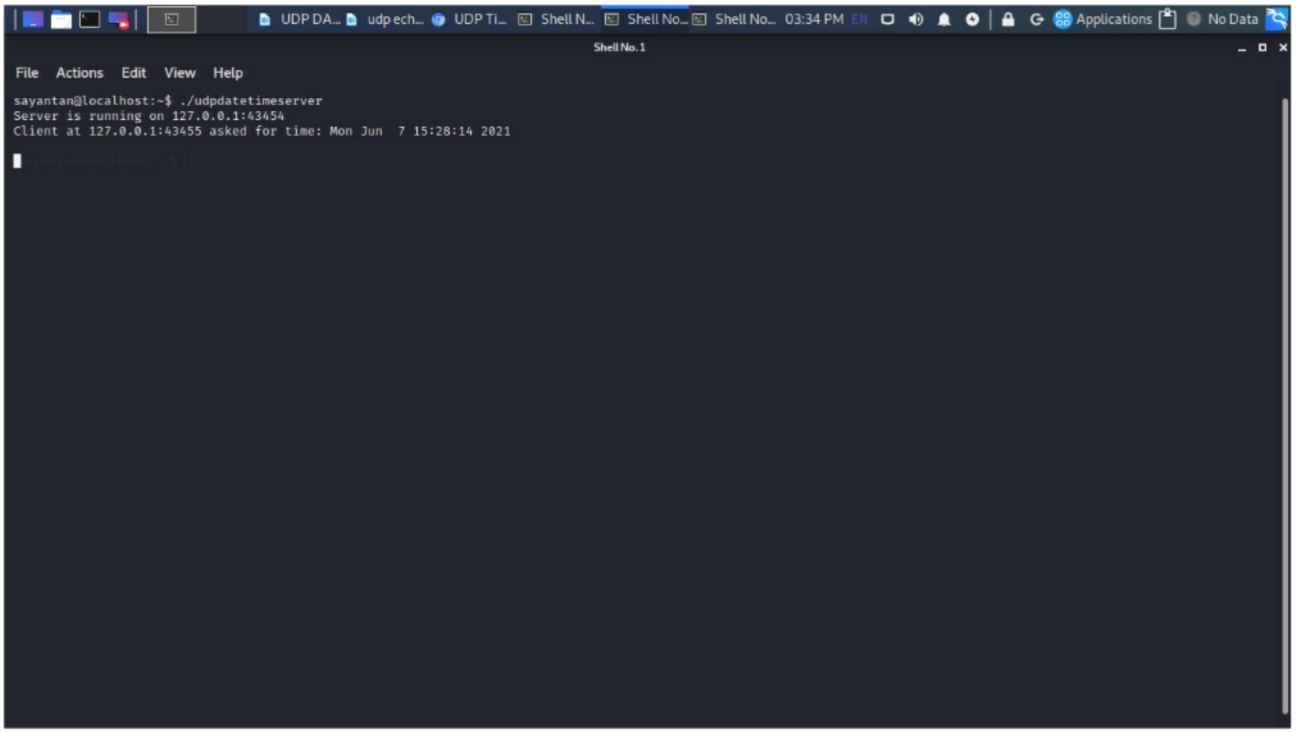
```
sayantan@localhost:~$ gcc udpdatetimeserver.c -o udpdatetimeserver
sayantan@localhost:~$ gcc udpdatetimeclient.c -o udpdatetimeclient


sayantan@localhost:~$ ./udpdatetimeserver
Server is running on 127.0.0.1:43454
Client at 127.0.0.1:43455 asked for time: Mon Jun  7 15:28:14 2021


sayantan@localhost:~$ ./udpdatetimeclient
Client is running on 127.0.0.1:43455
Server's Time: Mon Jun  7 15:28:14 2021

sayantan@localhost:~$
```



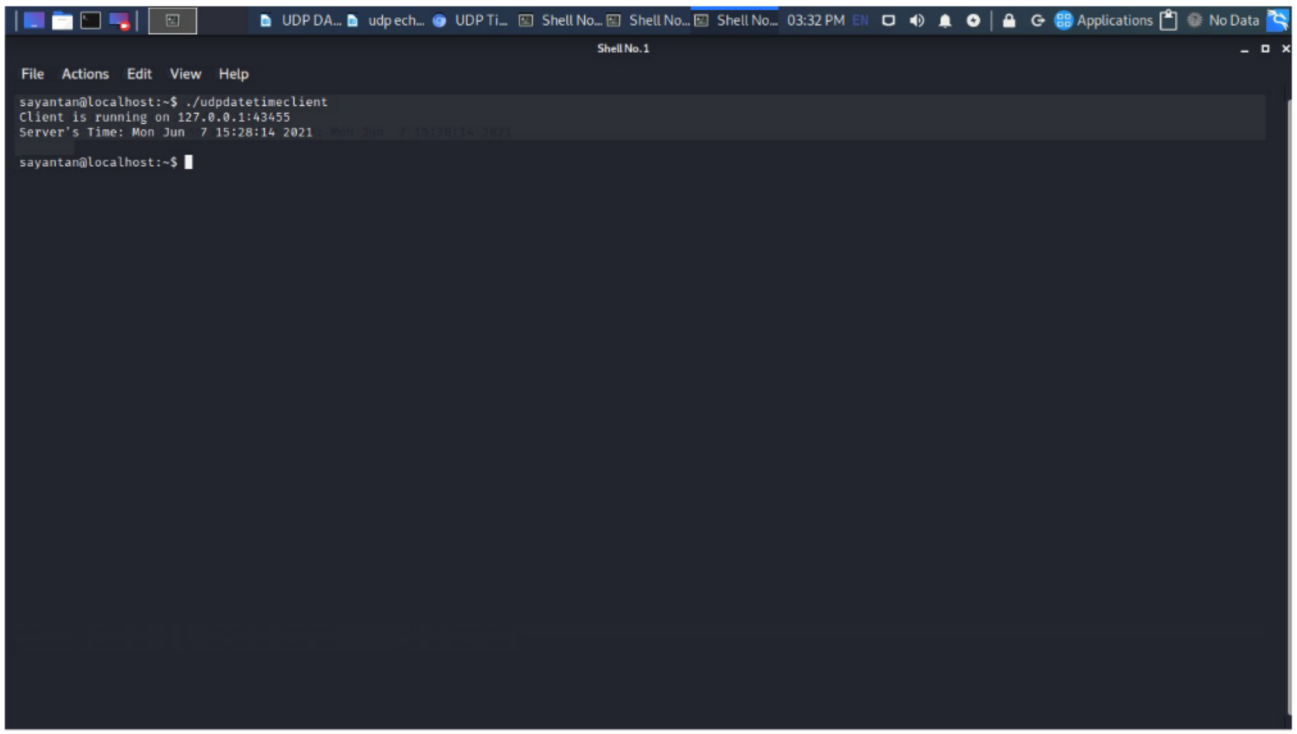
filename:udpdatetimeserver.c



filename: udpdatetimeclient.c



here the udp server program running and client asks for server's time from his side



client asks for the server's time and got the reply of server's time