Practical No. 9

Title: To Create TCP/IP packets using include files

Objectives:

1. To learn how to make TCP and IP Packets.
2. To learn TCP Socket Programming.

Problem statement

Write a program using TCP socket for wired network for following

a. Say Hello to Each other (For all students)

b. File transfer (For all students)

c. Calculator (Arithmetic) (50% students)

d. Calculator (Trigonometry) (50% students)

Demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.

Procedure:

To use tcp.h and ip.h header files and create the packets.

```
#include <netinet/in.h>

#include <sys/socket.h>

#include <netinet/ip.h>

#include <netinet/tcp.h>

main()

{

int tcp_socket;

struct sockaddr_in peer;

struct send_tcp

{

struct iphdr ip;

struct tcphdr tcp;

} packet;

packet.ip.version = 4; /* version of IP used */

packet.ip.ihl = 5; /* Internet Header Length (IHL) */
```

packet.ip.tos = 0; /* Type Of Service (TOS) */

packet.ip.tot_len = htons(40); /* total length of the IP datagram

*/

packet.ip.id = 1; /* identification */ packet.ip.frag_off = 0; /* fragmentation flag */ packet.ip.ttl = 255; /* Time To Live (TTL) */ packet.ip.protocol = IPPROTO_TCP; /* protocol used (TCP in this case) */

packet.ip.check = 14536; /* IP checksum */

packet.ip.saddr = inet_addr("1.2.3.4"); /* source address */ packet.ip.daddr = inet_addr("127.0.0.1"); /* destination address */

packet.tcp.source = htons(2000); /* source port */ packet.tcp.dest = htons(80); /* destination port */ packet.tcp.seq = 1; /* sequence number */ packet.tcp.ack_seq = 2; /* acknowledgement number */ packet.tcp.doff = 5; /* data offset */

packet.tcp.res1 = 0; /* reserved for future use (must be 0) */ packet.tcp.fin = 0; /* FIN flag */

packet.tcp.syn = 1; /* SYN flag */ packet.tcp.rst = 0; /* RST flag */ packet.tcp.psh = 0; /* PSH flag */ packet.tcp.ack = 0; /* ACK flag */ packet.tcp.urg = 0; /* URG flag */ packet.tcp.res2 = 0; /* reserved (must be 0) */ packet.tcp.window = htons(512); /* window */ packet.tcp.check = 8889; /* TCP checksum */ packet.tcp.urg_ptr = 0; /* urgent pointer */

/* Packet done here....open the connection and send the packet */

peer.sin_family = AF_INET;

peer.sin_port = htons(80);

peer.sin_addr.s_addr = inet_addr("127.0.0.1");

tcp_socket = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);

sendto(tcp_socket, &packet, sizeof(packet), 0, (struct sockaddr *)&peer, sizeof(peer));

/* the 0 is for the routing flag */

close(tcp_socket);

}

TCP Socket Programming:-

There are a few steps involved in using sockets:

1. Create the socket
2. Identify the socket
3. On the server, wait for an incoming connection
4. On the client, connect to the server's socket
5. Send and receive messages
6. Close the socket

Socket Primitives:-

Sockets - Procedures

Primitive	Meaning
Socket	Create a new communication endpoint
Bind	Attach a local address to a socket
Listen	Announce willingness to accept connections
Accept	Block caller until a connection request arrives
Connect	Actively attempt to establish a connection
Send	Send some data over the connection
Receive	Receive some data over the connection
Close	Release the connection

Table:- Socket Primitives

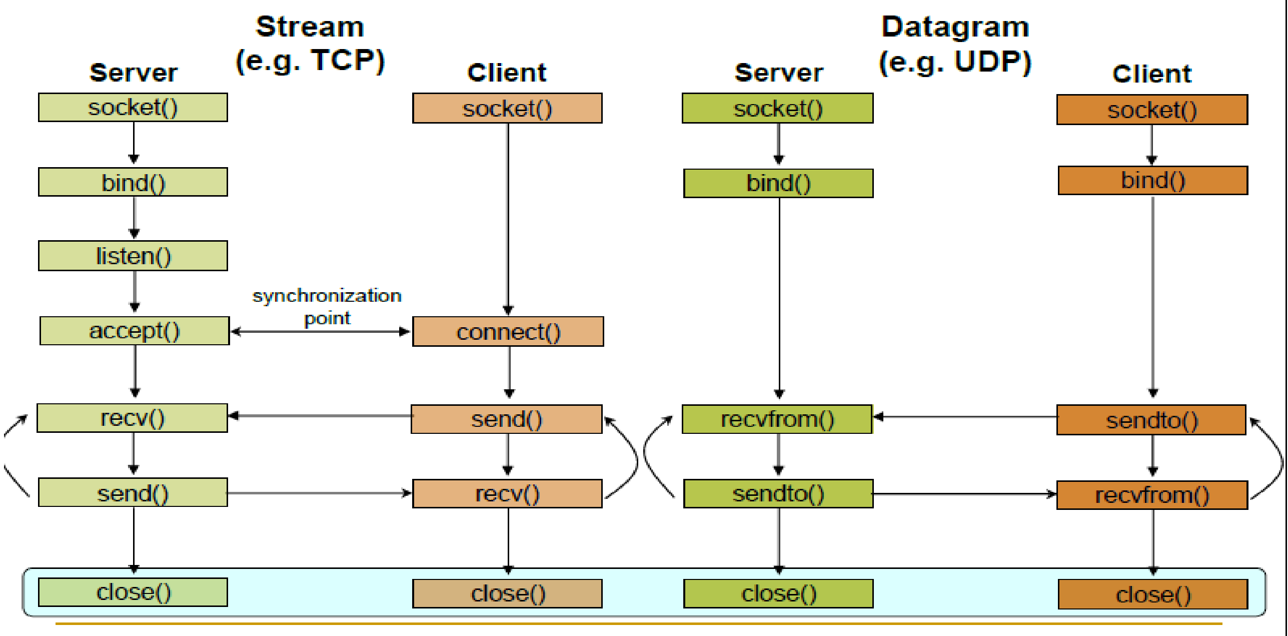

Fig: TCP and UDP Flow.

Conclusion: Thus we have successfully implemented the socket programming for TCP

Signature with Date