K.G.D.S.RANASINGHE
230522H

01.02.2026

LAB 1 - Socket Programming in C

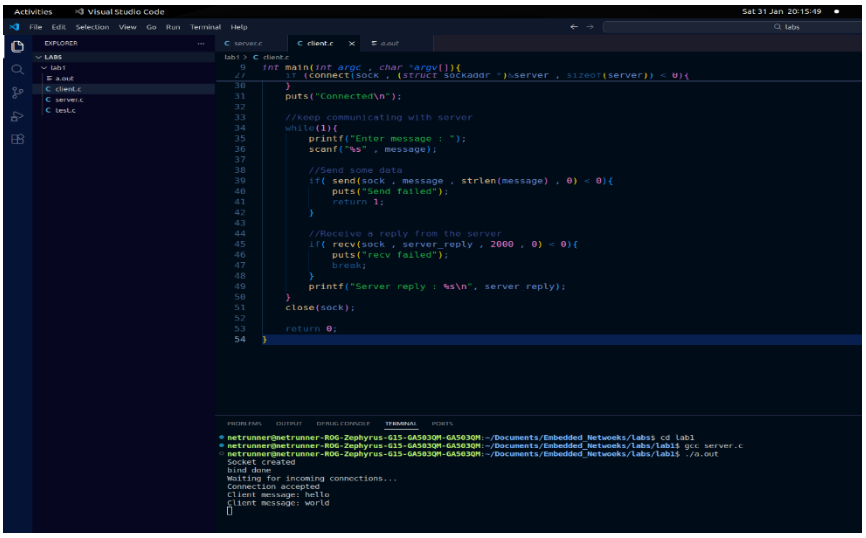
Embedded Computer Networks
In23-S4-CS3263

- **Upload your server.c and client.c files as txt files in Moodle.** ✓

- **Upload a pdf with screenshots of terminals when server and client are running. Please number the figures and give brief explanation about the figures by referring to the figure number**

The following figures were taken when the client.c and the server.c files were running in 2 terminals

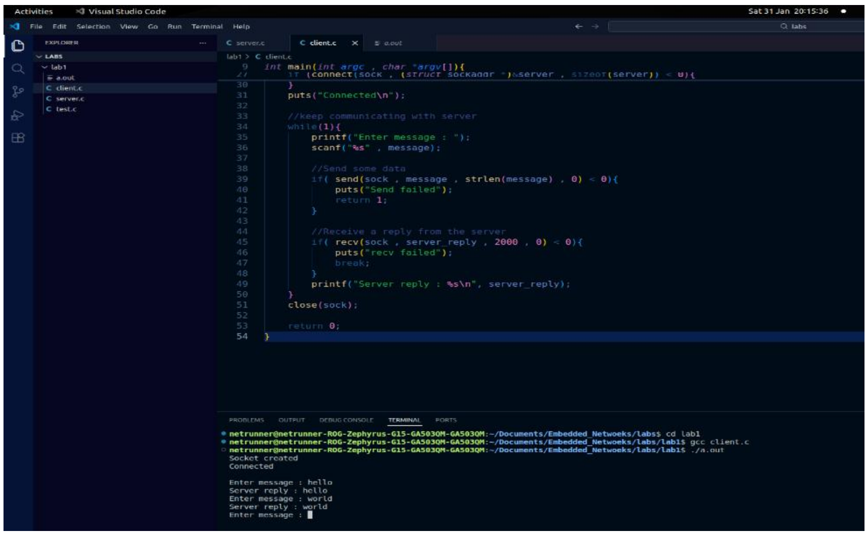
.



```c
int main(int argc , char *argv[]){
    if (connect(sock , (struct sockaddr *)&server , sizeof(server)) < 0){
    }
    puts("Connected\n");

    //keep communicating with server
    while(1){
        printf("Enter message : ");
        scanf("%s" , message);

        //Send some data
        if( send(sock , message , strlen(message) , 0) < 0){
            puts("Send failed");
            return 1;
        }

        //Receive a reply from the server
        if( recv(sock , server_reply , 2000 , 0) < 0){
            puts("recv failed");
            break;
        }
        printf("Server reply : %s\n", server_reply);
    }
    close(sock);

    return 0;
}
```

```
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs$ cd lab1
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ gcc server.c
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ ./a.out
Socket created
bind done
Waiting for incoming connections...
Connection accepted
Client message: hello
Client message: world
▯
```

Figure 1

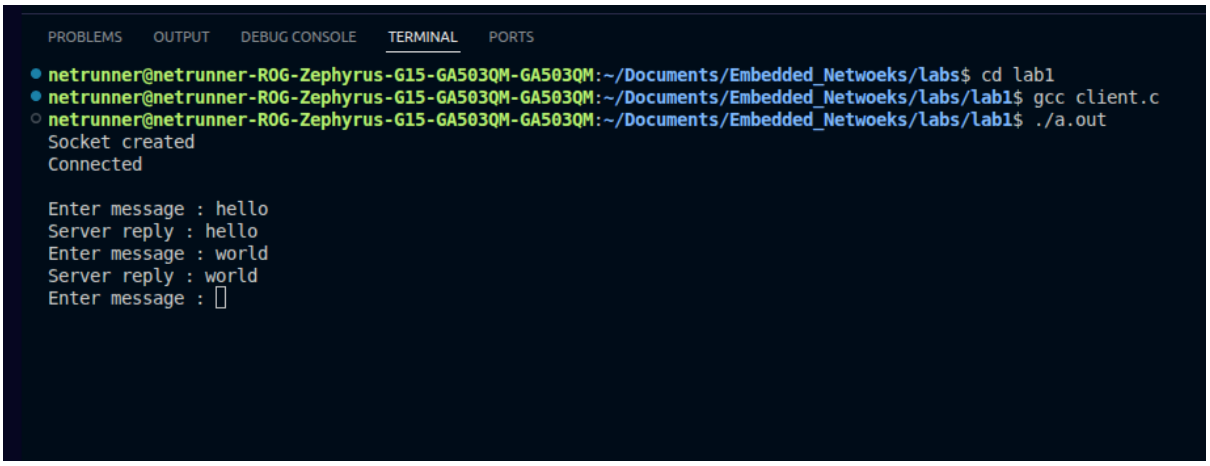


```c
int main(int argc , char *argv[]){
    if (connect(sock , (struct sockaddr *)&server , sizeof(server)) < 0){
    }
    puts("Connected\n");

    //keep communicating with server
    while(1){
        printf("Enter message : ");
        scanf("%s" , message);

        //Send some data
        if( send(sock , message , strlen(message) , 0) < 0){
            puts("Send failed");
            return 1;
        }

        //Receive a reply from the server
        if( recv(sock , server_reply , 2000 , 0) < 0){
            puts("recv failed");
            break;
        }
        printf("Server reply : %s\n", server_reply);
    }
    close(sock);

    return 0;
}
```

```
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs$ cd lab1
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ gcc client.c
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ ./a.out
Socket created
Connected

Enter message : hello
Server reply : hello
Enter message : world
Server reply : world
Enter message : █
```

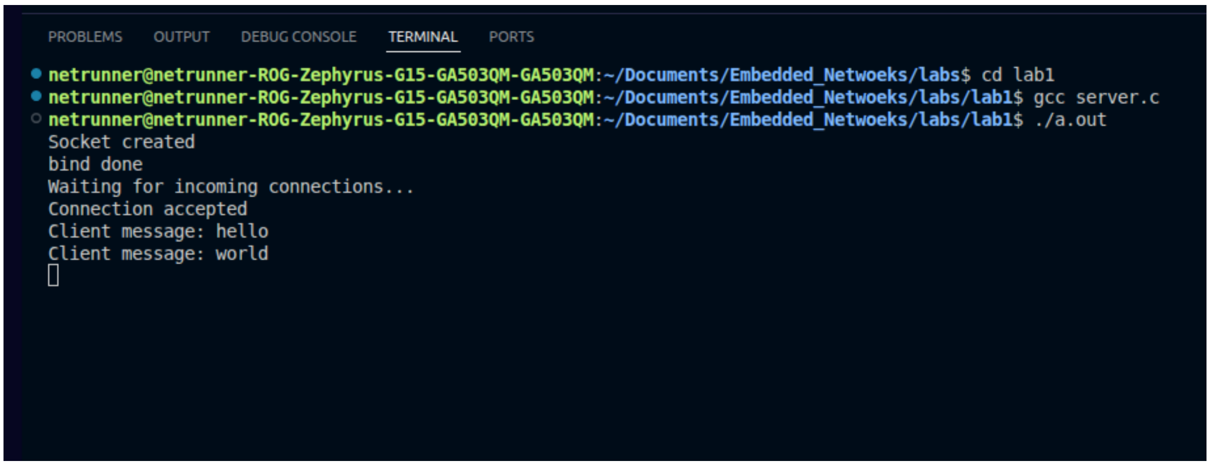
Figure 2

The figure 1 shows the server.c file in terminal. As you can see the server creates a socket and waits for the a successful bind. Then we initialize the client.c file in another terminal and after that client terminal shows that the connection is successful by "connected" message and the server terminal shows that the binding is successful and listens to incoming messages from the client. The following screenshots shows a closed up image of the 2 terminals



```
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs$ cd lab1
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ gcc client.c
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ ./a.out
Socket created
Connected

Enter message : hello
Server reply : hello
Enter message : world
Server reply : world
Enter message :
```

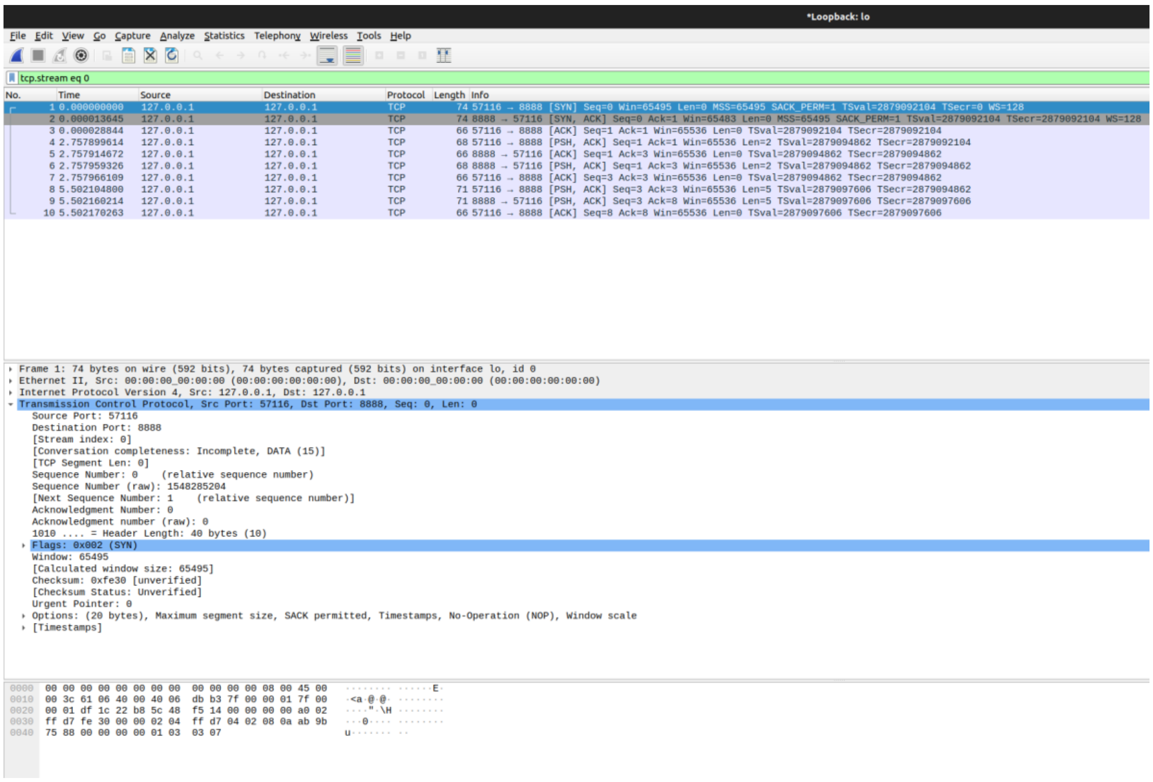
In here the client entered and message as "hello" and the server accept it and replied with the same message as "hello". Then another message was sent to the server as "world" and the server replied with the same message.



```
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs$ cd lab1
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ gcc server.c
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ ./a.out
Socket created
bind done
Waiting for incoming connections...
Connection accepted
Client message: hello
Client message: world
```

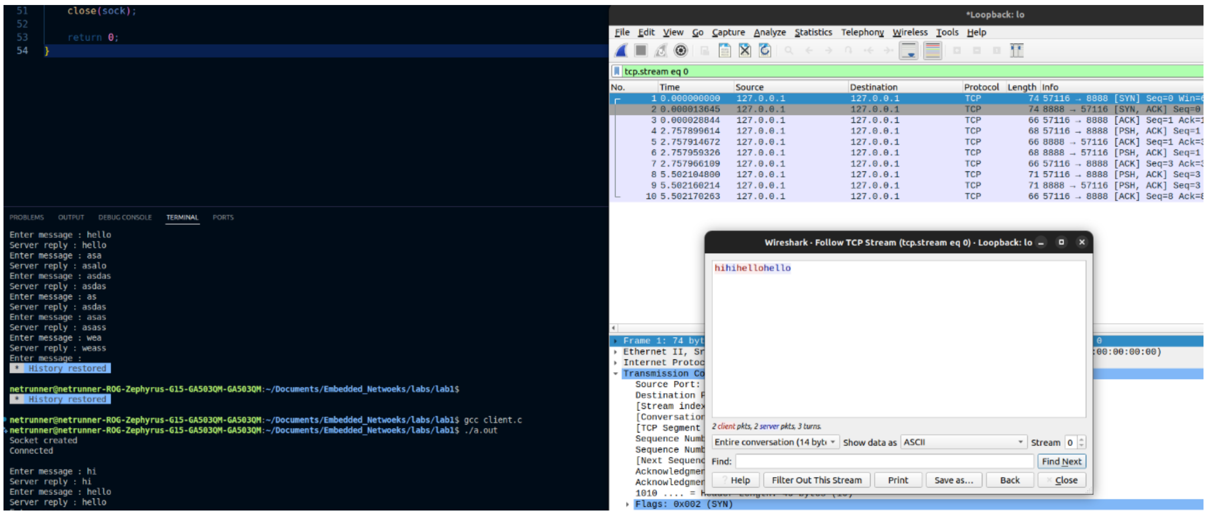
This is a closed up screenshot of the server's terminal with the received messages printed one by one

- **Wireshark packet capture**


*Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp.stream eq 0
No.  Time         Source     Destination  Protocol Length Info
   1 0.000000000  127.0.0.1  127.0.0.1    TCP      74 57116 → 8888 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2879092104 TSecr=0 WS=128
   2 0.000013645  127.0.0.1  127.0.0.1    TCP      74 8888 → 57116 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2879092104 TSecr=2879092104 WS=128
   3 0.000028844  127.0.0.1  127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2879092104 TSecr=2879092104
   4 2.757899614  127.0.0.1  127.0.0.1    TCP      68 57116 → 8888 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=2 TSval=2879094862 TSecr=2879092104
   5 2.757914672  127.0.0.1  127.0.0.1    TCP      66 8888 → 57116 [ACK] Seq=1 Ack=3 Win=65536 Len=0 TSval=2879094862 TSecr=2879094862
   6 2.757959326  127.0.0.1  127.0.0.1    TCP      68 8888 → 57116 [PSH, ACK] Seq=1 Ack=3 Win=65536 Len=2 TSval=2879094862 TSecr=2879094862
   7 2.757966109  127.0.0.1  127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=3 Ack=3 Win=65536 Len=0 TSval=2879094862 TSecr=2879094862
   8 5.502104800  127.0.0.1  127.0.0.1    TCP      71 57116 → 8888 [PSH, ACK] Seq=3 Ack=3 Win=65536 Len=5 TSval=2879097606 TSecr=2879094862
   9 5.502160214  127.0.0.1  127.0.0.1    TCP      71 8888 → 57116 [PSH, ACK] Seq=3 Ack=8 Win=65536 Len=5 TSval=2879097606 TSecr=2879097606
  10 5.502170263  127.0.0.1  127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=8 Ack=8 Win=65536 Len=0 TSval=2879097606 TSecr=2879097606

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 57116, Dst Port: 8888, Seq: 0, Len: 0
   Source Port: 57116
   Destination Port: 8888
   [Stream index: 0]
   [Conversation completeness: Incomplete, DATA (15)]
   [TCP Segment Len: 0]
   Sequence Number: 0    (relative sequence number)
   Sequence Number (raw): 1548285204
   [Next Sequence Number: 1    (relative sequence number)]
   Acknowledgment Number: 0
   Acknowledgment number (raw): 0
   1010 .... = Header Length: 40 bytes (10)
 Flags: 0x002 (SYN)
   Window: 65495
   [Calculated window size: 65495]
   Checksum: 0xfe30 [unverified]
   [Checksum Status: Unverified]
   Urgent Pointer: 0
 Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
 [Timestamps]

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ········ ······E·
0010  00 3c 61 06 40 00 40 06  db b3 7f 00 00 01 7f 00   ·<a·@·@· ········
0020  00 01 df 1c 22 b8 5c 48  f5 14 00 00 00 00 a0 02   ····"·\H ········
0030  ff d7 fe 30 00 00 02 04  ff d7 04 02 08 0a ab 9b   ···0···· ········
0040  75 88 00 00 00 00 01 03  03 07                     u······· ··
```

This image shows the TCP packet exchanged happened during this task. When we follow the TCP stream we can see that these exchanges resemble the message exchange we done while running the wireshark. The following image shows the TCP stream (RED- CLIENT AND BLUE-SERVER)

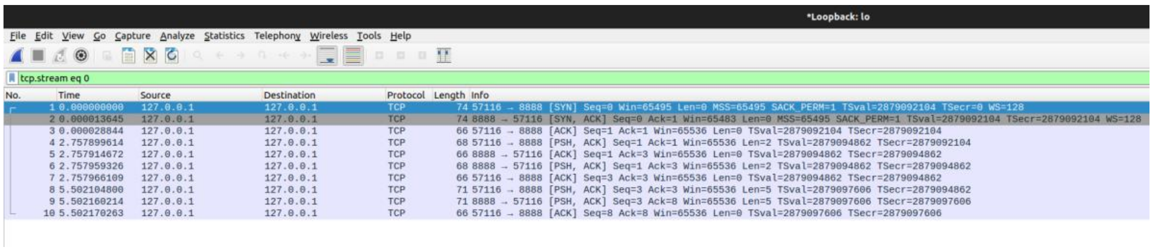


```
    close(sock);

    return 0;
}
```

```
Enter message : hello
Server reply : hello
Enter message : asa
Server reply : asalo
Enter message : asdas
Server reply : asdas
Enter message : as
Server reply : asdas
Enter message : asas
Server reply : asass
Enter message : wea
Server reply : weass
Enter message :
 *  History restored

netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$
 *  History restored

netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ gcc client.c
netrunner@netrunner-ROG-Zephyrus-G15-GA503QM-GA503QM:~/Documents/Embedded_Netwoeks/labs/lab1$ ./a.out
Socket created
Connected

Enter message : hi
Server reply : hi
Enter message : hello
Server reply : hello
```

```
hihihellohello
```

2 client pkts, 2 server pkts, 3 turns.


*Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp.stream eq 0

No.   Time         Source      Destination  Protocol Length Info
   1 0.000000000  127.0.0.1   127.0.0.1    TCP      74 57116 → 8888 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2879092104 TSecr=0 WS=128
   2 0.000013645  127.0.0.1   127.0.0.1    TCP      74 8888 → 57116 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2879092104 TSecr=2879092104 WS=128
   3 0.000028844  127.0.0.1   127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2879092104 TSecr=2879092104
   4 2.757899614  127.0.0.1   127.0.0.1    TCP      68 57116 → 8888 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=2 TSval=2879094862 TSecr=2879092104
   5 2.757914672  127.0.0.1   127.0.0.1    TCP      66 8888 → 57116 [ACK] Seq=1 Ack=3 Win=65536 Len=0 TSval=2879094862 TSecr=2879094862
   6 2.757959326  127.0.0.1   127.0.0.1    TCP      68 8888 → 57116 [PSH, ACK] Seq=1 Ack=3 Win=65536 Len=2 TSval=2879094862 TSecr=2879094862
   7 2.757966109  127.0.0.1   127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=3 Ack=3 Win=65536 Len=0 TSval=2879094862 TSecr=2879094862
   8 5.502104800  127.0.0.1   127.0.0.1    TCP      71 57116 → 8888 [PSH, ACK] Seq=3 Ack=3 Win=65536 Len=5 TSval=2879097606 TSecr=2879094862
   9 5.502160214  127.0.0.1   127.0.0.1    TCP      71 8888 → 57116 [PSH, ACK] Seq=3 Ack=8 Win=65536 Len=5 TSval=2879097606 TSecr=2879097606
  10 5.502170263  127.0.0.1   127.0.0.1    TCP      66 57116 → 8888 [ACK] Seq=8 Ack=8 Win=65536 Len=0 TSval=2879097606 TSecr=2879097606
```

When we pay attention to the captured 10 packets,

- **The first 3 packets with len=0**

  • Packet 1 (SYN): Client (57116) → Server (8888).
  • Packet 2 (SYN, ACK): Server (8888) → Client (57116)
  • Packet 3 (ACK): Client (57116) → Server (8888)
  Shows the 3 way handshake to establish the connection between the server and the client

- **The next 4 packets**

  • Packet 4 : Client → Server : **Len=2**, PSH, ACK
  This shows the "hi" message going to the server. The PSH (Push) flag show that the receiver should process it next.
  • Packet 5 : Server → Client : Len=0, ACK
  This shows the server's acknowledgement about the received hi msg
  • Packet 6 : Server → Client : **Len=2**, PSH, ACK
  This shows the server's reply as "hi".
  • Packet 7 : Client → Server : Len=0, ACK
  This shows the client's acknowledges about the received msg.

- **The last 3 packets**

  • Packet 8 : Client → Server **: Len=5**, PSH, ACK
  The client sends "hello" to the server.
  • Packet 9 : Server → Client : **Len=5**, PSH, ACK
  Instead of sending a separate empty ACK packet (like Packet 5) and then the data (like Packet 6), here it combined them as a single packet.
  • Packet 10 : Client → Server : Len=0, ACK
  The client acknowledges about the received "hello" reply.