



# **iJRASET**

International Journal For Research in  
Applied Science and Engineering Technology



---

# **INTERNATIONAL JOURNAL FOR RESEARCH**

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

---

**Volume: 8      Issue: V      Month of publication: May 2020**

**DOI: <http://doi.org/10.22214/ijraset.2020.5210>**

**[www.ijraset.com](http://www.ijraset.com)**

**Call: ☎ 08813907089**

**E-mail ID: [ijraset@gmail.com](mailto:ijraset@gmail.com)**

# Chat Application via UDP Protocol in C

A. Krishna Kumar<sup>1</sup>, G. Ashritha<sup>2</sup>

<sup>1</sup>Assistant Professor, CBIT, Hyderabad

<sup>2</sup>M.E (ES & VLSI Design), CBIT, Hyderabad

**Abstract:** User Datagram Protocol (UDP) is a connectionless and light-weight protocol. It is fast and efficient. It is also easy to be used in a network. Socket network programming is one of the most popular technologies used to build a chat application and establishing network communication between systems. Socket programming helps to implement the bottom level of network communication, using Application Programming Interface (API). In this paper, we present a chat based application implemented in 'C' via UDP protocol by Windows Socket Programming based on Client-Server model. Visual Studio Ultimate 2010 software is used. Wireshark software is also used in the background.

**Index Terms:** UDP, Windows Socket Programming, Client-Server model.

## I. INTRODUCTION

THE User Datagram Protocol (UDP) belongs to the Transport Layer of OSI model. In computer networking, UDP is one of the core members of the Internet protocol suite. The protocol was designed by David P. Reed in 1980 and formally defined in RFC 768. With UDP, computer applications can send messages, in this case referred to as 'datagrams,' to other hosts on an Internet Protocol (IP) network. In order to set up communication channels or datapaths, prior communications are not required.

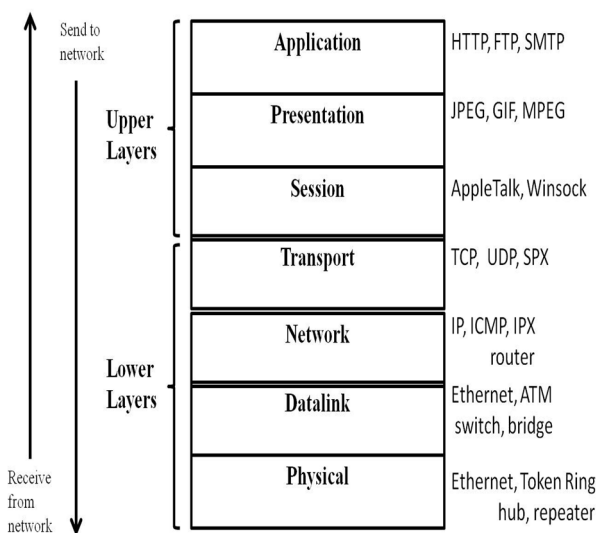


Fig 1 OSI Model

The characteristic features of UDP are it is connectionless, fast and efficient, low overhead, header length of 8 bytes, error-checking capacity and light-weight. It is also used in other protocols, namely, DNS (Domain Name Service), DHCP (Dynamic Host Configuration Protocol), SNMP (Simple Network Management Protocol) and VOIP (Voice Over Internet Protocol). The UDP header format consists of the fields - source port, destination port, length and checksum. Source and destination port numbers are communication end points for sending and receiving devices. The length field indicates the total size of the datagram including both header and data. The checksum used for data integrity allows receivers to cross-check incoming data.

Compared to other protocols, UDP is unique in that it does not establish end-to-end connections between communicating end systems. UDP communication consequently does not incur connection establishments and teardown overheads. Because of these characteristics, UDP can offer very efficient communication transport to some applications. On many platforms, applications can send UDP datagrams at the line rate of link interface, which is often much greater than the available path capacity.

## II. RELATED WORK

A lot of research work has been carried out in the area of Socket Programming via Ethernet. A few of the substantial works are discussed below.

In the paper, “UDP Based Chat Application<sup>[1]</sup>,” the authors proposed a method to make a chat room using socket based on User Datagram Protocol (UDP) which enables the feature of acknowledgments after every message sent. It is analogous to a dedicated chat server having a Server and n number of Clients. One can achieve many machines to communicate through peer to peer communication, multicasting and File sending, after client and server set up to connect. There might be different system and network failures occurring during communication taking place, which they have discussed and proposed a convenient solution for that. This system designed accomplishes a satisfying and efficient communication between the users by experimental verification.

In the paper, “TCP & UDP Packets Analysis using Wireshark<sup>[2]</sup>,” the authors proposed to analyze packets of TCP and UDP while sending a e-mail using a tool called

Wireshark. It is a free and open-source packet analyzer. They used different parameters like frame no, on wire frame length, IP source, IP destination, header length of the packets and also window size value to inspect the packets of TCP and UDP. They also analyzed the packet flow scenario i.e. how packet is flowing from source to destination while sending a e-mail. as we know that TCP is connection oriented protocol and connection is established before packets flow from source to destination and acknowledgement is also sent by receiver and UDP is connectionless protocol so according to frame length, frame no. and bytes captured during sending a mail, all the values of these are more of TCP than UDP.

The paper, “The Socket Programming and Software Design for Communication Based on Client/Server<sup>[3]</sup>,” introduces the application of the client/server(C/S) mode, the concept and the programming principle of the socket based on C/S. The method of software design for the communication between the client, server-process using the socket method is mainly analyzed, and gives examples of connection-oriented service program. The transmission layer can provide connection-oriented to use TCP protocol, connectionless-oriented to use UDP protocol. There are two different types of services with different kinds of sockets. Only by understanding the characters of TCP and UDP protocols, the service provided by the two protocols for application, can one know what is dealt in these protocols, what needs to be dealt with in application, how can we easily compile vigorously, highly efficient client service program.

The paper, “A Design and Implementation of Active Network Socket Programming<sup>[4]</sup>,” introduces a concept that is similar to the network socket programming. It establishes a set of simple interfaces to program active applications. This set of interfaces, known as Active Network Socket Programming (ANSP), works on top of all other execution environments in future. Thus, ANSP offers a concept that is similar to “write once, run everywhere.” It is an open programming model which active applications can work on all execution environments. It solves the heterogeneity within active networks. This is specifically useful when active applications require to access all regions within a heterogeneous network to deploy special service at critical points or to monitor the performance of the entire networks. Instead of introducing a new platform, this approach provides a thin, transparent layer on top of existing environments that can be easily installed for all active applications.

In the paper, “Transport Layer Proxy for Stateful UDP Packet Filtering<sup>[5]</sup>,” the authors propose a Transport Layer Proxy (TLP) to provide a secure UDP firewall traversal service on the transport layer. The TLP supports TCP as well. For each UDP association with endpoints separated by a TLP server, the TLP server performs user level or host-level authentication, packet filtering, packet relaying, optional network address translation, session logging, timing-out of idle association, and other security related functions. The core of the TLP is a two-step TLP binding procedure that makes a UDP association stateful between a TLP client and a TLP server. This binding procedure supports Active UDP Open, Passive UDP Open, and Source-Specific UDP Open, which a local program may perform on a UDP socket.

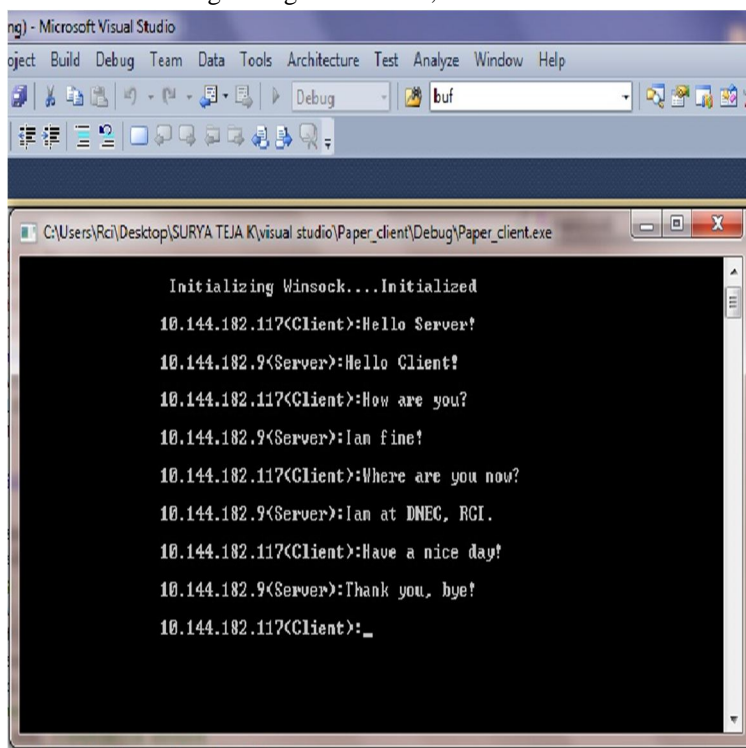
## III. METHODOLOGY

In Socket Programming world, all communication is based on Client-Server model. One PC acts as Server and the other acts as Client. A Server is an entity that provides information whereas a Client is an entity that seeks information. The Server starts up first and waits for the Client to connect to it. When a Client successfully connects, it requests some info. The Server serves this info to the Client. The Client then disconnects and the Server waits for more Clients. In Windows Socket Programming, the steps involved in establishing communication from one PC to the other on the Server side are Initialize WSA (Windows Socket Application), Create a Socket, Bind the Socket, Listen on the Socket, Accept a Connection, Send and Receive data, Disconnect. Similarly, steps involved on Client side are Initialize WSA (Windows Socket Application), Create a Socket, Connect to the Server, Send and Receive data, Disconnect.

The chat conversation is initiated by the Client. In the client code, a buffer of maximum size of 512 bytes is declared. The message manually entered by the user is sent into the buffer by gets() function. It is then sent to the Server through send() function. On the Server side, message is received through receive() function into a buffer and it is displayed on console by puts() function. On both sides, the IP address of the device sending the message is displayed via the network parameter inet\_ntoa(si\_other.sin\_addr).

#### IV. RESULTS & DISCUSSION

The results in the form of chat conversation between client and server as obtained in Visual Studio Ultimate 2010 are presented below. The IP addresses of the devices are also displayed on the console. The verification reports obtained in Wireshark software are also presented which show various fields including timing information, source and destination IP addresses, port numbers and data.



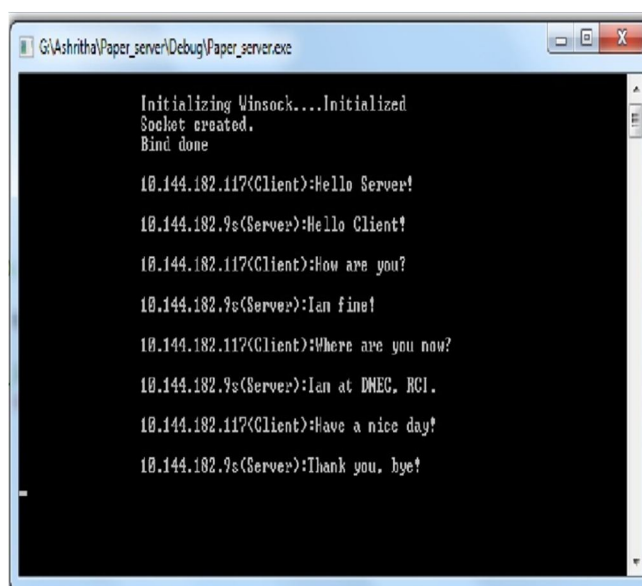
```

ng) - Microsoft Visual Studio
Project Build Debug Team Data Tools Architecture Test Analyze Window Help
Debug buf

C:\Users\Rci\Desktop\SURYA TEJA K\visual studio\Paper_client\Debug\Paper_client.exe

Initializing Winsock...Initialized
10.144.182.117(Client):Hello Server!
10.144.182.9(Server):Hello Client!
10.144.182.117(Client):How are you?
10.144.182.9(Server):I am fine!
10.144.182.117(Client):Where are you now?
10.144.182.9(Server):I am at DNEG, RCI.
10.144.182.117(Client):Have a nice day!
10.144.182.9(Server):Thank you, bye!
10.144.182.117(Client):_
  
```

Fig 2 Visual Studio Ultimate 2010 – Client



```

G:\Ashritha\Paper_server\Debug\Paper_server.exe

Initializing Winsock...Initialized
Socket created.
Bind done

10.144.182.117(Client):Hello Server!
10.144.182.9s(Server):Hello Client!
10.144.182.117(Client):How are you?
10.144.182.9s(Server):I am fine!
10.144.182.117(Client):Where are you now?
10.144.182.9s(Server):I am at DNEG, RCI.
10.144.182.117(Client):Have a nice day!
10.144.182.9s(Server):Thank you, bye!
  
```

Fig 3 Visual Studio Ultimate 2010 - Server



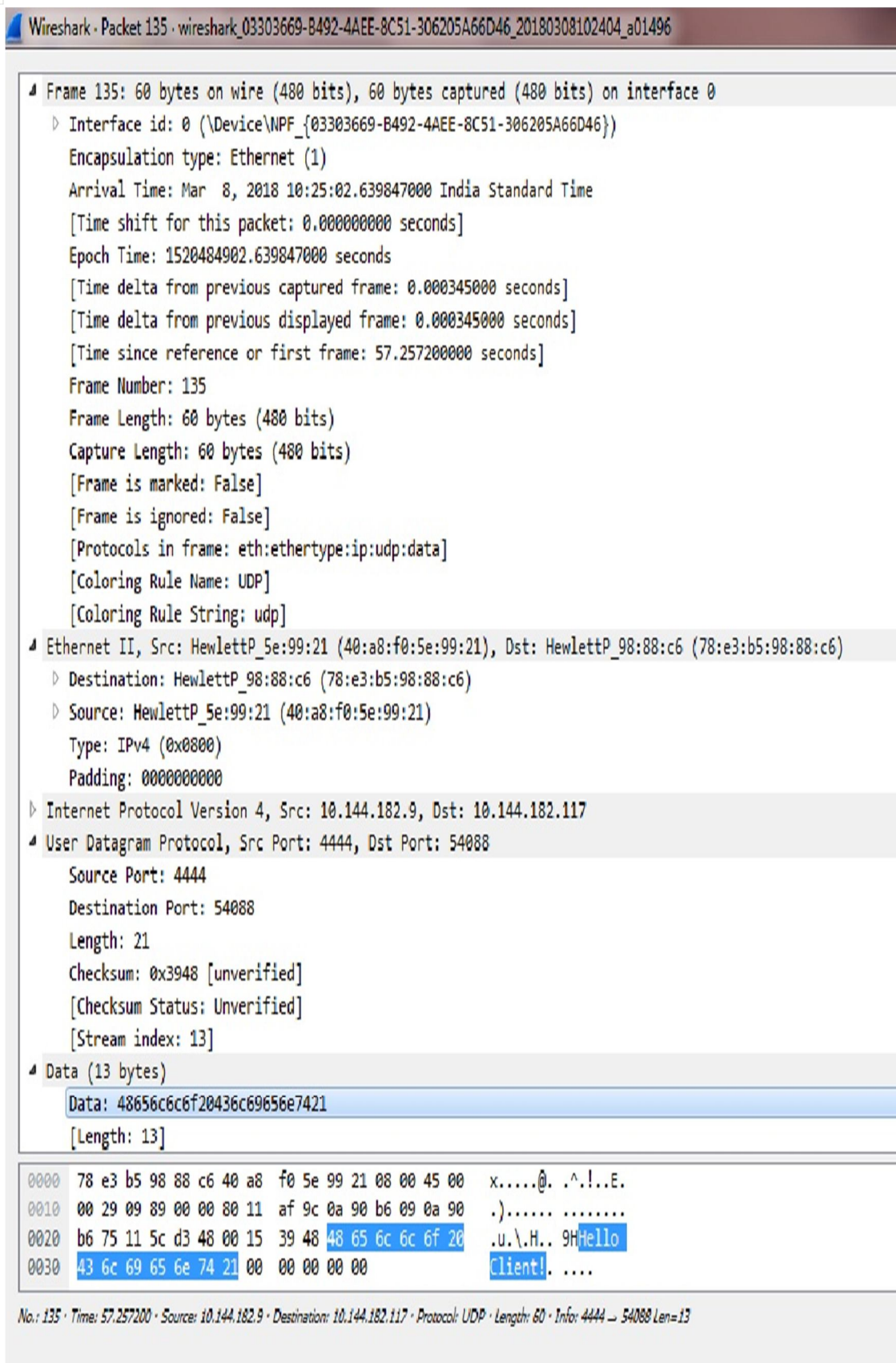


Fig 4 Wireshark Report of Client

```

112.49.831773000 10.144.182.117 10.144.182.9 UDP 60 Source port: 54088 Destination port: krb524
Frame 112: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
  Interface id: 0
  Encapsulation type: Ethernet (1)
  Arrival Time: Mar  8, 2018 10:29:43.346237000 India Standard Time
  [Time shift for this packet: 0.000000000 seconds]
  Epoch Time: 1520485183.346237000 seconds
  [Time delta from previous captured frame: 0.000392000 seconds]
  [Time delta from previous displayed frame: 0.000392000 seconds]
  [Time since reference or first frame: 49.831773000 seconds]
  Frame Number: 112
  Frame Length: 60 bytes (480 bits)
  Capture Length: 60 bytes (480 bits)
  [Frame is marked: False]
  [Frame is ignored: False]
  [Protocols in frame: eth:ip:udp:data]
  [Coloring Rule Name: UDP]
  [Coloring Rule String: udp]
Ethernet II, Src: Hewlett_98:88:c6 (78:e3:b5:98:88:c6), Dst: HewlettP_5e:99:21 (40:a8:f0:5e:99:21)
  Destination: HewlettP_5e:99:21 (40:a8:f0:5e:99:21)
  Source: Hewlett_98:88:c6 (78:e3:b5:98:88:c6)
  Type: IP (0x0800)
  Padding: 0000000000
Internet Protocol Version 4, Src: 10.144.182.117 (10.144.182.117), Dst: 10.144.182.9 (10.144.182.9)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
  Total Length: 41
  Identification: 0x0884 (2180)
  Flags: 0x00
  Fragment offset: 0
  Time to live: 128
  Protocol: UDP (17)
  Header checksum: 0xb0a1 [validation disabled]
  Source: 10.144.182.117 (10.144.182.117)
  Destination: 10.144.182.9 (10.144.182.9)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
User Datagram Protocol, Src Port: 54088 (54088), Dst Port: krb524 (4444)
  Source port: 54088 (54088)
  Destination port: krb524 (4444)
  Length: 21
  Checksum: 0x2940 [validation disabled]
Data (13 bytes)
  Data: 48656c6c6f2053657276657221
  [Length: 13]

0000  40 a8 f0 5e 99 21 78 e3  b5 98 88 c6 08 00 45 00  @..A.!X. ....E.
0010  00 29 08 84 00 00 80 11  b0 a1 0a 90 b6 75 0a 90  .).....u...
0020  b6 09 d3 48 11 5c 00 15  29 40 48 65 6c 6c 6f 20  ...H... )Hello
0030  53 65 72 76 65 72 21 00  00 00 00 00                server]. ....

```

Fig 5 Wireshark Report of Server

## V. CONCLUSIONS

A chat application similar to that of messenger app was implemented successfully via UDP protocol by Windows Socket Programming based on Client-Server model.



## REFERENCES

- [1] Akshit Malhotra, Vaibhav Sharma, Prateek Gandhi, Neetesh Purohit, "UDP based chat application," 2<sup>nd</sup> International Conference on Computer Engineering and Technology, Vol.6, pp: V6-374 – V6-377, 2010.
- [2] Dr. Mahesh Kumar, Rakhi Yadav, "TCP & UDP Packets Analysis using Wireshark," International Journal on Science, Engineering, Technology and Research, Vol.7, Issue:4, 2015.
- [3] Ming Xue, Changjun Zhu, "The Socket Programming and Software Design for Communication based on Client/Server," Pacific-Asia Conference on Circuits, Communications and Systems, pp: 775-777, 2009.
- [4] K.L.E. Law, R. Leung, "A Design and Implementation of Active Network Socket Programming," Eleventh International Conference on Computer Communications and Networks, pp:78-83, 2002.
- [5] R.K.C Chang, K.P. Fung, "Transport Layer Proxy for Stateful UDP Packet Filtering," Seventh International Symposium on Computers and Communication, pp:595-600, 2002.
- [6] Manas Pratim Sarma, "Performance Measurement of TCP and UDP using Different Queuing Algorithm in High Speed Local Area Network," International Journal of Future Computer and Communication, Vol.2, No.6, December, 2013.
- [7] Ming-Hung Wang, Lung-Wen Chen, Po-Wen Chi, Ching Luang Lei, "SDUDP: A Reliable UDP based Transmission Protocol over SDN," IEEE Access, Vol.5, pp.5904-3916, 2017.
- [8] G. Xylomenos, G.C. Polyzos, "TCP and UDP Performance over a Wireless LAN," Eighth Annual Joint Conference of the IEEE Computer and Communications Societies, Vol.2, pp.439-446, 1999.
- [9] Santhosh Kumar, Sonam Rai, "Survey on Transport Layer Protocols: TCP and UDP," International Journal of Computer Applications, Vol.46, No.7, May 2012.
- [10] Wasan Ali Hussein, Song Feng Lu, "Performance Comparison of Transport Layer Protocols for Multimedia Application in Wired Networks," IOSR Journal of Computer Engineering, Vol.18, Issue-6, pp.33-39, 2016.





10.22214/IJRASET



45.98



IMPACT FACTOR:  
7.129



IMPACT FACTOR:  
7.429



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24\*7 Support on Whatsapp)