Volume 4, Issue 5, May 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Hybrid TCP/IP and UDP: A Review Article

Sangeeta Yadav
Student
Department of Computer Science Engg.
G.I.T.M, Gurgaon , India

Vivek Bansal
Associate Professor
Department of Computer Science Engg.
G.I.T.M, Gurgaon , India

Abstract: - Today all the application on the Internet usesthe World Wide Web and HTTP protocol is used by all the transportation of the Web, which is built upon the TCP model. But as we know TCP is poorly suited for the short conversationsthat comprise a significant component of web traffic. The use of the TCP protocol insures the application that runs above it the ordered transparence of data without any package loss, but causes long delays and inefficiency in short transmissions. In this paper we studied both UDP and TCP models and perform a comparative study of both. And conclude that if a proxy server is build which uses hybrid scheme of TCP/UDP as the underlying transport protocol for carrying web traffic.

Keywords:-TCP, UDP, HTTP, Proxy Server, IP Address.

I. INTRODUCTION

- ***Network***

A network is a collection of computers and other devices that can send data to and receive data from each other, more or less in real time. Wires generally connect a network, and the bits of data are turned into electromagnetic waves that move through wires. Each machine on a network is called node. Most nodes are computers, but printers, routers, bridges, gateways, dumb terminals, and Coca-Cola machines can also be nodes. The machines which are fully functional computers are called hosts'.

Every network node has an **address***:* a series of bytes that uniquely identify it. We can think of this group of bytes as a number. The more bytes in each address, the more addresses are there available and the more devices that can be connected to the network simultaneously. Addresses are assigned differently on different networks. AppleTalk addresses are chosen randomly at startup of each host. The host then checks to see whether any other machine is using the same address and if so then the hosts randomly choose another and the process goes on until finally we get address that is not being used. Ethernet addresses are attached to the physical Ethernet hardware.

Internet addresses are normally assigned to a computer by the organization which is responsible to it. However, addresses that an organization is allowed to choose for its computers are assigned to by Organization Internet Service Provider (ISP). ISPs get there, Internet Protocols(1P) addresses from one of the three regional Registries(the registry for America 'and Africa is' ARIN, the American Registry for Internet Numbers, http://www.arfn.net/).which in tern assigned IP addresses by the Internet Assigned Number Authority(IANA, http://wrVW.fana.org/)

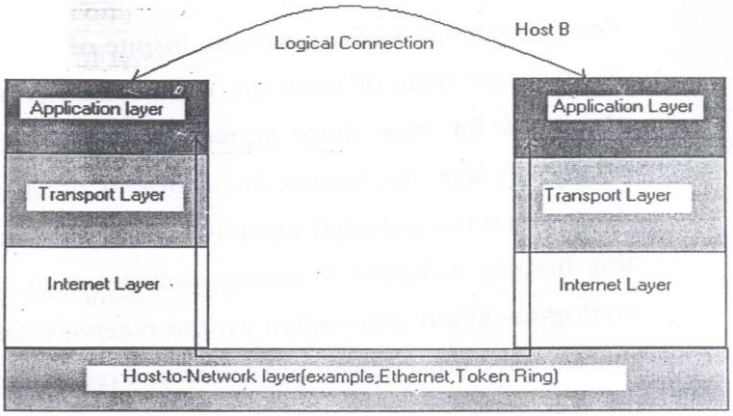
On some kinds of networks, nodes also have some name that helps human being to identify them. At a particular moment of time a particular name normally refers to exactly one address. But it does not mean names are locked to addresses. So two conditions can occur either name can change while address stays the same, or address can change while the name stays the same. It is not uncommon for address to have several names and it is possible, though somewhat less common, for one name to refer to several different addresses. All modem networks are *packet-switched* networks. This means the data travel on network is broken into chunks called packets, and each packet is handled separately. The most important advantage of breaking data into individually address packets is that packet from many ongoing exchanges can travel on one wire, which makes it much cheaper to build a network: many computer can share the same wire without interfering. Another advantage of packet is that checksums can be used to detect whether apacket was damaged in transit. Computers needs to say to pass data back and forth are called **protocol**. A protocol is a precise set of rules defining how the computer will communicate; the format of addresses, how data is splited into packets etc. There are many protocols like:- HTTP, TCP, UDP, IP, Ethernet Card ranging from Application Layer to Host-to-Network Layer

- ***The Layers of Network***

Sending data across network is a complex operation that must be carefully tuned to the physical characteristics of the network as well as the logical character of the data being sent. Software that send data across a network must understand how to avoid collision between packets, how to convert digital data to analog signals, how to detect and correct the errors, how to route the packets from one host to another, and more.

To make this complexity manageable and to hide most of it from the application developers and end user, the different aspect of network communication are separated into multiple layers. Each layer represent a different level of abstraction between the physical hardware (e.g.,wires and electricity) and the information being transmitted. In theory each layer

talks only to the layers immediately above and immediately below it. There are four layer in TCP/IP based network-moded and these are shown

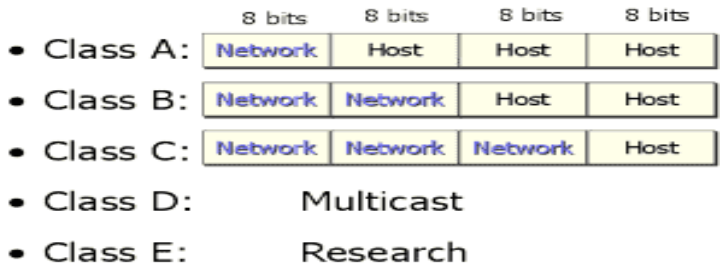


- ***IP Addresses and Domain Names***

Every computer on an IP network is identified by an 4-byte number. This is normally written in format like 192.168.1.1 where each of the four numbers is one unsigned byte ranging in value from 0 to 255. Ever computer attached to' an IP network has an unique 4-byte address. This address is known as destination address in IP datagram header. Routers along the way choose the best route to send the packet along by inspecting the destination address.

Although computers are comfortable with the numbers, human beings are not good at remembering. them. Therefore, the Domain Name System (DNS) was developed to translate host names that humans can remember (like http://www.yahoo.com) into numeric Internet address (like 196.122.144.122).When Java programs access the network, they need to process both these numeric addresses and their corresponding host names. There are our classes of addressing

- Class A - Range 1.0.0.0 to 126.0.0.0
- Class B -Range 128.0.0.0 to 191.255.0.0
- Class C -Range 192.0.0.0 to 223.255.255.0
- Class D - Range 224.0.0.0 to 239.255.255.255



- ***Transmission Control Protocol***

Abbreviation of Transmission Control Protocol is TCP, pronounced as separate letters. TCP is one of the main **protocols** in **TCP/IP** networks. Whereas the **IP** protocol deals only with **packets**, TCP enables two **hosts** to establish a connection and exchange streams of data. TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent. It is described in STD-7/RFC-793. TCP is a connection-oriented protocol that is responsible for reliable communication between two end processes. The unit of data transferred is called a stream, which is simply a sequence of bytes. Being connection-oriented means that before actually transmitting data, you must open the connection between the two end points. The data can be transferred in full duplex (send and receive on a single connection). When the transfer is done, you have to close the connection to free system resources. Both ends know when the session is opened (begin) and is closed (end). The data transfer cannot take place before both ends have agreed upon the connection. The connection can be closed by either side. Provision is made to close gracefully or just abort the connection.The Transmission Control Protocol (TCP) provides a reliable, connection-oriented data stream delivery service. TCP provides this sequenced, in-order, error-free packet delivery by using a sliding window flow control protocol with adaptive retransmission. This reliability comes at a cost, though: TCP requires a three-way handshake for connection setup as well as a connection tear down (except for abrupt through TCP' reset). Datagram must be acknowledged by the receivers, which generate additional traffic. While the setup and tear down costs amortize well over long connections, they are expensive for short connection. The **Transmission Control Protocol** (**TCP**) is one of the core protocols of the Internet protocol suite. TCP provides reliable, in-order delivery of a stream of bytes, making it suitable for applications like file transfer and e-mail. It is so important in the Internet protocol suite that sometimes the entire suite is referred to as "the TCP/IP protocol suite." TCP is the transport protocol that manages the individual conversations between web servers and web clients. TCP divides the HTTP messages into smaller pieces, called segments, to be sent to the destination client. It is also responsible for controlling the size and rate at which messages are exchanged between the server and the client.

- ***User Datagram Protocol***

User Datagram Protocol, a **connectionless protocol** that is similar to TCP that also runs on top of IP networks. Unlike **TCP/IP**, UDP/IP provides very few error recovery services, offering instead a direct way to send and receive datagrams over an IP network. It's used primarily for **broadcasting** messages over a network.

UDP stands for User Datagram Protocol. It is described in STD-6/RFC-768 and provides a connectionless host-to-host communication path. UDP has minimal overhead:; each packet on the network is composed of a small header and user data. It is called a UDP datagram.

UDP preserves datagram boundaries between the sender and the receiver. It means that the receiver socket will receive an OnDataAvailable event for each datagram sent and the Receive method will return a complete datagram for each call. If the buffer is too small, the datagram will be truncated. If the buffer is too large, only one datagram is returned, the remaining buffer space is not touched. As UDP is connectionless it means that a datagram can be sent at any moment without prior advertising, negotiation or preparation. Just send the datagram and hope the receiver is able to handle it.UDP is an unreliable protocol. There is absolutely no guarantee that the datagram will be delivered to the destination host. But to be honest, the failure rate is very low on the Internet and nearly null on a LAN unless the bandwidth is full.

Not only the datagram can be undelivered, but it can be delivered in an incorrect order. It means you can receive a packet before another one, even if the second has been sent before the first you just received. You can also receive the same packet twice.Your application must be prepared to handle all those situations: missing datagram, duplicate datagram or datagram in the incorrect order. You must program error detection and correction. For example, if you need to transfer some file, you'd better set up a kind of zmodemprotocol.The main advantages for UDP are that datagram boundaries are respected, you can broadcast, and it is fast.The main disadvantage is unreliability and therefore complicated to program at the application level.

The User Datagram Protocol (UDP) provides a best-effort datagram service and therefore can operate in a more efficient manner than TCP. For example, UDP require no connection setup or tear down, no acknowledgment, and little protocol state machine processing. On the flip side, UDP does not offer any of TCP's reliable delivery or good congestion control behavior. Unlike TCP, the User Datagram Protocol (UDP) does not present data as a stream of bytes, nor does it require that you establish a connection with another program in order to exchange information. Data is exchanged in discrete units called datagrams, which are similar to IP datagrams. In fact, the only features that UDP offers over raw IP datagrams are port numbers and an optional checksum. UDP is sometimes referred to as an unreliable protocol because when a program sends a UDP datagram over the network, there is no way for it to know that it actually arrived at it's destination. This means that the sender and receiver must typically implement their own application protocol on top of UDP

II. WHERE TCP/IP AND UDP USE

TCP and UDP use IP to transport them from one network to another. For example IP as a sort of high-way that allows other protocols to get on and find their way to other computers. TCP and UDP are the "trucks" on the highway, and the "load" they are carrying are protocols such as HTTP, File Transfer Protocol and more. As TCP and UDP are transport protocols used by protocols such as FTP, HTTP, and SMTP. While both TCP and UDP are used to transport other protocols

HTTP is a request/reply protocol, where client applications can request data from servers by providing a *Universal Resource Locator* (URL). HTTP is used to address many different types of resources, including text, image, audio, video, executable files, index search results, and database query result

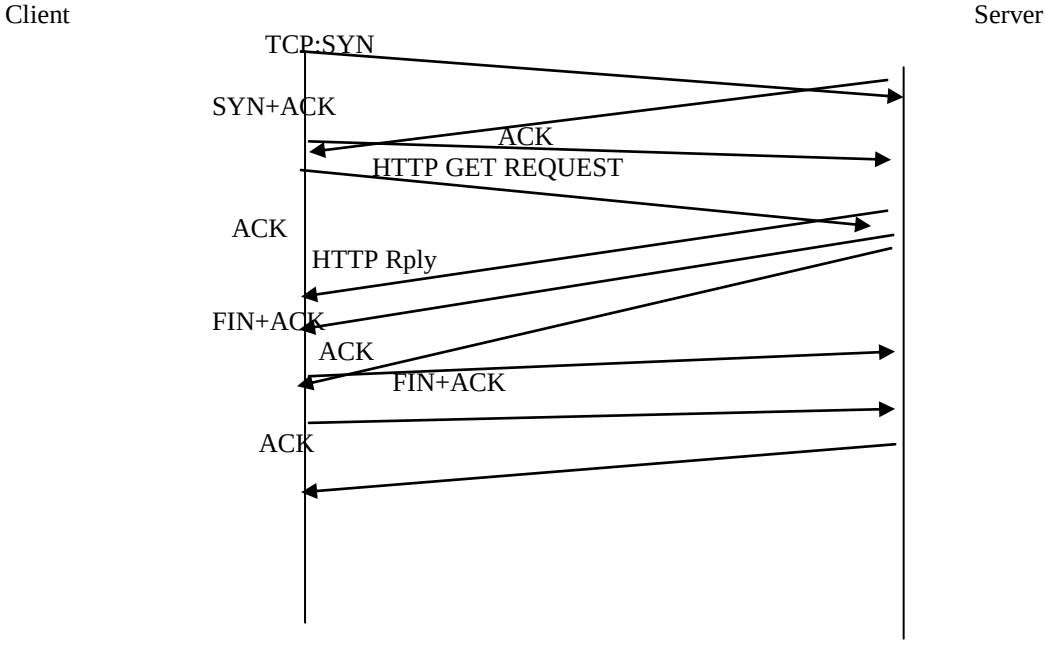


Figure illustrates a typical packet exchange for an HTTP GET request between a client and a server. In Figure only two packets seem to be *useful* (i.e., carrying data): one is the HTTP GET request, and the other one the HTTP reply. All other packets represent TCP overhead. Under the HTTP protocol Version 1.0, each transfer requires a separate TCP connection. HTTP Version 1.1 introduced *persistent HTTP connections* to address this problem.
One may expect that all pages for a web site (i.e., with a common server identifier) reside on the same server.
The HTTP REDIRECT mechanism is used to support heterogeneous content. HTTP server A hands off HTTP requests from clients to server B by sending a REDIRECT as a response to an HTTP GET request. Note that the HTTP REDIRECT is part of a short HTTP transfer. Using TCP for an HTTP request–HTTP REDIRECT pair requires at least 7 packets (usually 9-10 packets), while two packets suffice if UDP is used as the underlying transport protocols.

III. COMPARATIVE ANALYSIS - TCP - UDP

UDP is not a reliable protocol as TCP, UDP has its own properties to fit best for these applications.
Few reasons to choose UDP than TCP are:-

- No connection establishmentTCP uses a three-way handshake to establish a connection before it starts to transfer data UDP just blasts away sending data without any formal preliminaries to setup any connection. Thus UDP does not introduce any —delay to establish a connection, which is very key factor for interactive real-time applications.
- No connection state TCP maintains connection state in the end systems during the transmission. This connection state includes receive and send buffers, congestion control parameters, and sequence and acknowledgment number parameters. On the other hand, TCP does not maintain connection state and does not track any of these parameters. For this reason, a server devoted to a particular application can typically support many more active clients when the application runs OVER UDP rather than TCP.
- Small segment header overheadThe TCP segment has 20 bytes of header overhead in its every segment, whereas UDP only has 8 bytes of overhead.
- Unregulated send rate
- UDP preferred over TCP would be where the following characteristics hold true:

- Messages that require no acknowledgement
- Messages between hosts are sporadic or irregular
- Reliability is implemented at the process level.

In real terms this would be something like an online video broadcast where frame losses are pretty much tolerable. In other words this is pretty much what Multicast needs. A Multicast server would not have to be concerned about re-transmitting packets once they are sent out using UDP.

IV. HOW UDP IS BETTER THAN TCP

TCP can be a tiny bit less efficient than UDP. TCP has things like sequence numbers and acknowledgements to detect lost and out-of-order transmission. If you don't mind lost data, out of order data, and data that are duplicated, UDP can be a tiny bit more efficient On the other hand, TCP has things like congestion avoidance. If you go spraying UDP packets to the network, you can overflow your routers and firewalls. This will cause lots of packet loss and can make your entire network slow.
TCP is widely used. Because of that, TCP stacks have been optimized in all "serious" operating systems. Don't go around thinking that TCP is somehow abysmally slow. It has a fraction of a percent overhead compared to UDP (YMMV: percentage-wise, tiny data packets have more overhead than max size packets because TCP has a couple of data fields more in the packet headers). The overhead isn't there just to annoy you; it's there because it is required for reliable data transfer
UDP can be appropriate for applications where you don't want packet retransmission. If a packet is lost on the way, it's ok for it to get lost rather than have it retransmitted a while later. Consider a IP telephone application: a TCP retransmit causes a pause in the voice stream (= bad), while a lost UDP packet causes a small fraction of a second's glitch in the sound, after which the voice simply continues (= better). Some game applications can be the same: if you transmit your guy"'s coordinates on the playing field, a TCP retransmit can make you freeze for a few seconds until a retransmit timer fires. Whereas a lost UDP packet could make your guy jump from one position to a third one without the intervening position, which can be better. Be careful though: all packets are lost, so if you send a packet that informs everyone "my guy died" and it gets lost, well, what happens then. If you are making a choice between TCP and UDP based on performance, you are asking the wrong question. TCP vs UDP is not a performance issue, it is a question of whether you need a reliable stream protocol or an unreliable packet protocol.

V. HYBRID APPROACH OF USING TCP/IP AND UDP

If we built a proxy server, which can be connected by any known browser, and through it the browser will be able to connect to a UDP/TCP based server. While the connection between the proxy. and the browser is one by HTTP, which uses only the TCP protocol, the connection between the proxy server and the other server which server holds the desired files that the client seeks, can be either by HTTP- TCP protocol or HTTP-TCP/UDP protocol. A server that sits between a client application, such as a Web browser, and a real server. It intercepts all requests to the real server to see if it can fulfill the requests itself. If not, it forwards the request to the real server.
Proxy servers have two main purposes:

- **Improve Performance:** Proxy servers can dramatically improve performance for groups of users. This is because it saves the results of all requests for a certain amount of time. Consider the case where both user X and user Y access the World Wide Web through a proxy server. First user X requests a certain Web page, which we'll call Page 1. Sometime later, user Y requests the same page. Instead of forwarding the request to the Web serverwhere Page 1 resides, which can be a time-consuming operation, the proxy server simply returns the Page 1 that it already fetched for user X. Since the proxy server is often on the same network as the user, this is a much faster operation. Real proxy servers support hundreds or thousands of users. The major online services such as America Online, MSN and Yahoo, for example, employ an array of proxy servers.
- **Filter Requests:** Proxy servers can also be used to filter requests. For example, a company might use a proxy server to prevent its employees from accessing a specific set of Web sites.

VI. CONCLUSION

In this paper we have discus the basics of networking, its terminologies, and about working of TCP/IP and UDP models. From the comparison of TCP/IP and UDP we conclude that UDP model is more efficient and over TCP model. But we can't make choice between both of them one benifits from the low overhead for short transmission and other provides large transfer of data with relible delivery acknowlwdgement and good congestion behavior. It is better to use hybrid scheme of combining both TCP/UDP model with HTTP for taking advantage of the low cost of using UDP with the high reliability and congestion control features of TCP.

REFERENCES

[1] Hybrid TCP-UDP Transport forWeb Traffic by Israel Cidon, Amit Gupta, Raphael Rom, and ChristophSchuba Sun Microsystems Laboratories

[2] Bob Braden. RFC-1644 T/TCP - TCP Extensions for Transactions. Network Working Group, July 1994.

[3] CERT. IP Spoofing Attacks and Hijacked Terminal Connections, CA-95:01. Computer Emergency Response Team, Carnegie Mellon University, Pittsburgh, Pennsylvania, January 1995.

[4] Roy T. Fielding, Jim Gettys, Jeffrey C. Mogul, HenrikFrystyk Nielsen, and Tim Berners-Lee.RFC-2068

[5] Hypertext Transfer Protocol – HTTP/1.1. Network Working Group, January 1997.

[6] "HYBRID TCP-UDP Transport for Web Traffic" article by. Geoffrey Raehr, Amit

[7] Steven D. Gribble. UC Berkeley Home IP HTTP traces. July 1997. Available at http://www.acm.org/sigcomm/ITA/

[8] Je_rey Mogul and Steve Deering. Path MTU Discovery. Arpanet Working Group Requests for Comment, DDN Network Information Center, SRI International, Menlo Park, CA, November 1990.RFC-1191.

[9] B. Cli_ord Neuman. The Virtual System Model: A Scalable Approach to Organizing Large Systems. PhD thesis, University of Seattle, Washington, 1992.

[10] Venkata N. Padmanabhan. Private communication, 1997.

[11] K. Poduri and K. Nichols. Simulation studies of increased initial tcp window size, February 1999. Internet Draft expires 8/98.

[12] Kedarnath Poduri and Kathleen Nichols. RFC-2415 Simulation studies of increased initial TCPwindow size. Network Working Group, September 1998.

