Reliable Data Transfer Using User Datagram Transport Based on UDP

Moh Moh Nyunt Aung; Khin Ei Ei Chan
Computer University, Sittway
mohmoh.na@gmail.com, khineieichan@gmail.com

Abstract

As the network bandwidth and delay increase, TCP becomes inefficient. The protocol, named User Datagram Transport (UDT), or UDP (User Datagram Protocol) based data transfer protocol, works above UDP with reliability. UDT uses both positive acknowledgement and negative acknowledgement to guarantee data reliability. In this paper, the UDT high performance data transport protocol is used to provide data transfer service that can utilize the bandwidth efficiently and fairly and to obtain rapid development of UDP-based transport protocol and application specific data transfer. Moreover, UDT can also be used to transfer data from one single byte to hundreds of gigabits. It is built on top of UDP with reliability control. Meanwhile, a set of Application Programming Interface (APIs) is used to support easy application implementation, including both reliable data streaming and partial reliable messaging. This system can transfer any unactive files between the two machines. This paper presents the details of File Transfer using UDT protocol with simulation, implementation results and analysis.

1. Introduction

Rapid increase of network bandwidth has enabled data transfer at speed of gigabits per second. Meanwhile, new optical technologies together with modern switching and routing technologies are expected to provide much faster network links in the near future. The widely used TCP on the Internet has exposed its inefficiencies as the network bandwidth and delay product (BDP) increases [1]. To reach high speed over high BDP networks, TCP requires an extremely low link loss error rate, which is impossible for the UDT technology. An example of applications is streaming join whose performance is limited by the slowest data stream [6]. This is motivated to design and develop a lightweight high performance application-level data transfer protocol. This new protocol is named UDT or UDP-based Data Transfer Protocol.

User Datagram Protocol (UDP) is a connectionless protocol. So, the transferred data is not reliable [2]. Lacking reliability, UDP applications must generally be willing to accept some loss. Because it cannot be retransmit the packet loss in real time. Some application such as File Transfer Protocol (FTP) [4] is needed to be reliable. UDP most often uses in Streaming media, real-time multiplayer games and voice over IP [5]. Although UDP is connectionless, UDP-based Data Transfer Protocol called User Datagram Transport (UDT) is connection-oriented, end-to-end, unicast, reliability. This system can be transferred the data with reliability over the UDP Protocol in the transport layer [1].

This paper organizes with the following sections. Section 2 describes related work, section 3 describes background theory, section 4 presents flow diagram of the system, section 5 explains system implementation, section 6 shows experimental result, section 7 describes conclusion, the second last section describes limitation and the last section is references.

2. Related Work

Earlier idea of protocols using rate based congestion control can be found data transfer protocols that transfer data block by block. It updates data sending rate after each block according the packet loss of the last data block transfer. However, most recent rate based schemes are motivated by Internet streaming multimedia applications [6]. Simple Available Bandwidth Utilization Library is an application level protocol that uses UDP to transfer data and TCP to transfer control information. And the UDT protocol can support data transfer at very high speeds with a single flow while sharing the network resources fairly with concurrent flows. This is not flexible by using existing Internet data transport protocols (e.g., TCP). One of the most protocols called UDT (for UDP-based Data Transfer) can be used to support teraflows for distributed and grid computing applications. UDT teraflows are sent between Amsterdam (NetherLight) and Chicago (StarLight). The terabytes data is being delivered to the Asia-Pacific region, including Australia, Japan, South Korea, and China [7]. Of course, UDT can also be deployed more efficiently by modifying the operating system kernel [6] [7].

3. Background Theory

Within the last few decades, networking has evolved from dumb terminals connected to mainframes via 110 bps serial ports to packet-switched networks covering the entire world at speeds of hundreds of megabytes per second. Protocols are usually designed to be able to utilize the network as efficiently as possible, but little attention is paid to how efficiently the host system resources can be used. However, when the network is nearly as fast as the endpoints, as is the case on gigabit networks and currently available PCs, the endpoints are the limiting factor, not the network. The traditional way of describing network protocols is to divide the protocols into multiple layers. The OSI Reference Model is the most commonly used model for describing network architectures [1]. In the OSI model, network protocols are split into seven layers with each layer communicating only with the layers below and above it. While the OSI model is a good way of designing and describing network protocols, it is not a very good way of implementing them. In the Transport Layer of the OSI model, TCP and UDP protocols are used for data transfer [1].

3.1. User Datagram Transport Protocol

UDT adapts itself into the layered network protocol architecture as shown in Figure 1. UDT uses UDP through the socket interface provided by operating systems. Meanwhile, it provides a UDT socket interface to applications. Applications can call the UDT socket API in the same way they call the system socket API. UDT is the application layer above UDP. The application exchanges its data through UDT socket, which then uses UDP socket to send or receive the data [6] [7].

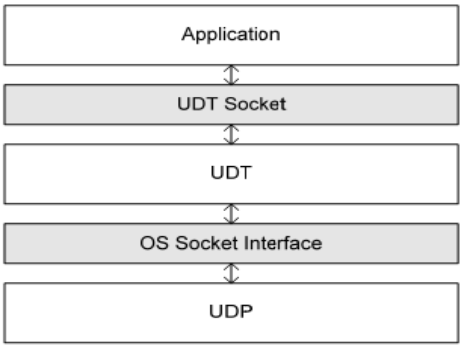

Figure 1. UDT in the Layer Architecture.

It supports both reliable data streaming and partial reliable messaging. Figure 3 describes the protocol architecture between the UDT sender and the receiver. In Figure 2, the UDT entity A sends application data to the UDT entity B. The data is sent from A's sender to B's receiver, whereas the control flow is exchanged between the two receiver. All UDT entities have the same architectures, each having both a sender and a receiver. This figure demonstrates the situation when a UDT entity A sends data to another UDT entity B. Data is transferred from A's sender to B's receiver, whereas control information is exchanged between the two receivers.

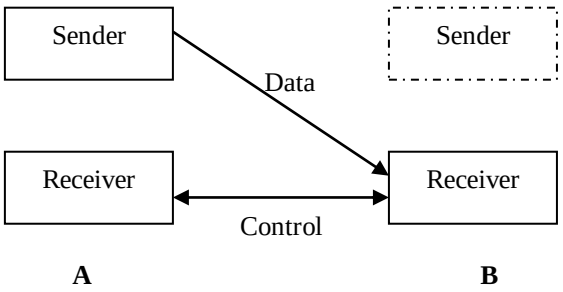

Figure 2. UDT Protocol Architecture.

The receiver is also responsible for triggering and processing all control events, including congestion control and reliability control, and their related mechanisms as well. UDT uses rate-based congestion control (rate control) and window-based flow control to regulate the outgoing data traffic. Rate control updates the packet-sending period every constant interval, whereas flow control updates the flow window size each time and acknowledgment packet is received.

4. Flow Diagram of the System

The sender transfers a file by using UDT based on UDP. If the receiver does not receive the file, the error message will be shown at the sender. If the receiver receives the file, the file size and file transfer time will be shown at the sender. This system flow diagram of the sender is shown in Figure 3.

If the connection has been created, the user will enter IP address and file name. Then the system processes this file and checks the packet lost. If the packet is lost, the error message will be shown. Otherwise, the successful message will be shown. The system flow diagram of the receiver is shown in Figure 4.

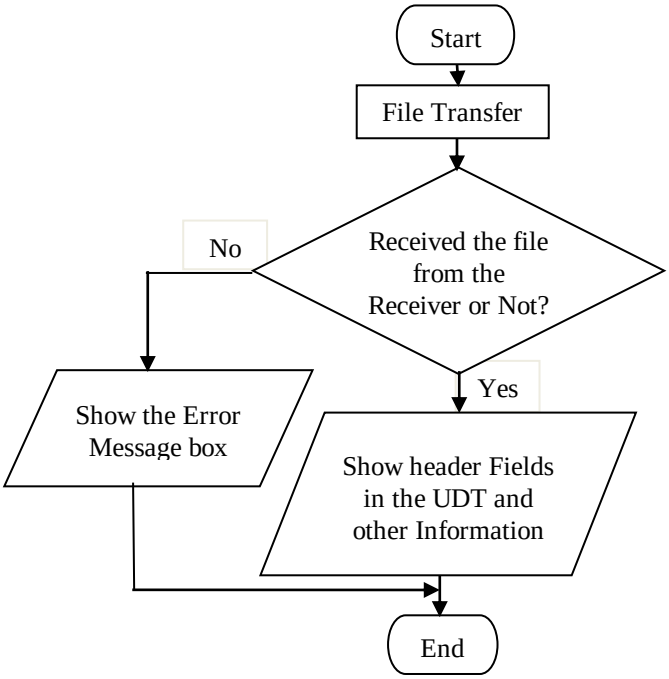

Figure 3. System Flow Diagram of the Sender

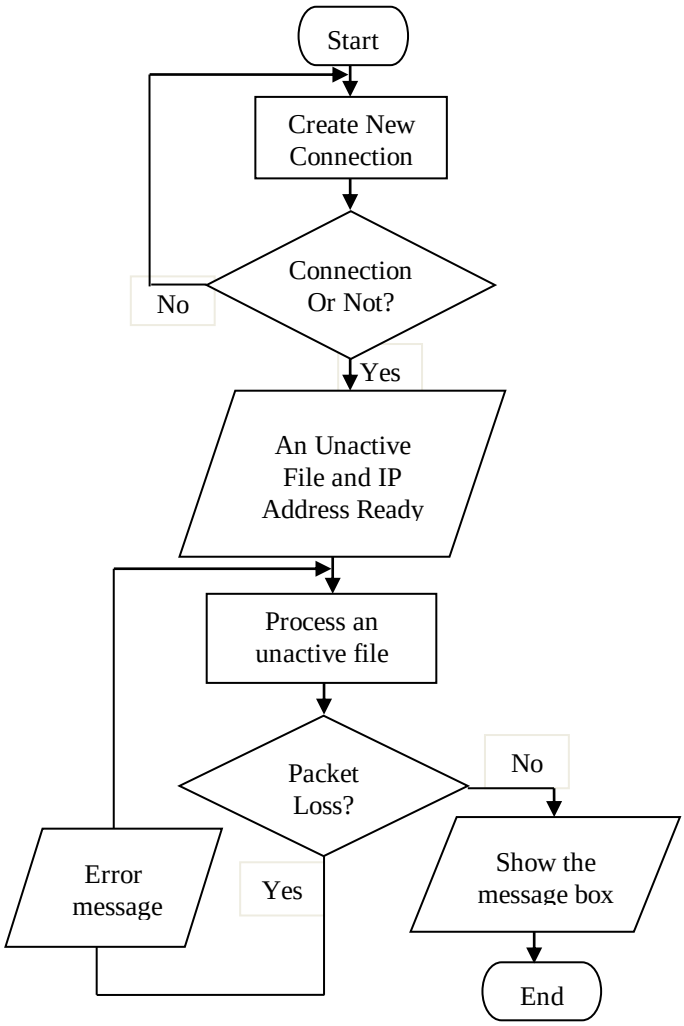

Figure 4. System Flow Diagram of the Receiver

5. System Implementation

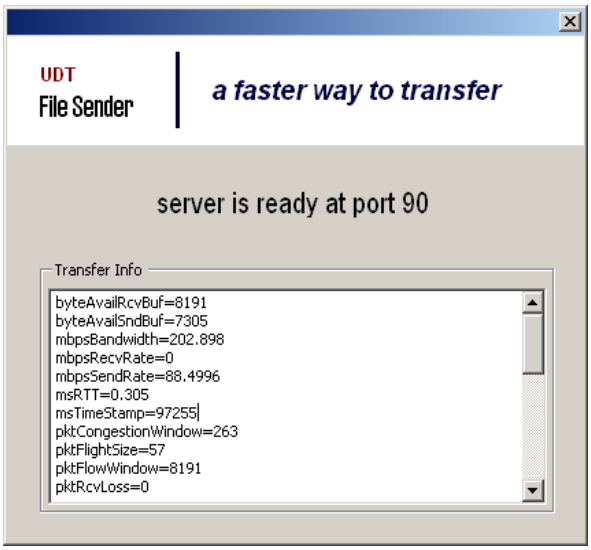

Figure 5. Results for the UDT Sender

The system is implemented for the UDT Sender and the UDT Receiver to transfer any unactive file types. Figure 5 shows the resultant information for the UDT Sender. By seeing, this information, UDT data transfer with differences in among other protocol is significant. The user can see the processed information by the system about the transferring of an unactive file on the sender side. The system will describe the byte available buffer size of Receiver and Sender. And the bandwidth describes in megabit per second (mbps). Round Trip Time (RTT), Time Stamp, mbps Send Rate is shown too significant the using of UDT protocol. Unlike UDP, UDT uses both positive acknowledgement and negative acknowledgement. So, UDP- based transport protocol is reliable because of using both acknowledgements. Therefore, it may also prevent the packet loss. This information is useful for the user. UDT Receiver information for the user is shown in Figure 6.

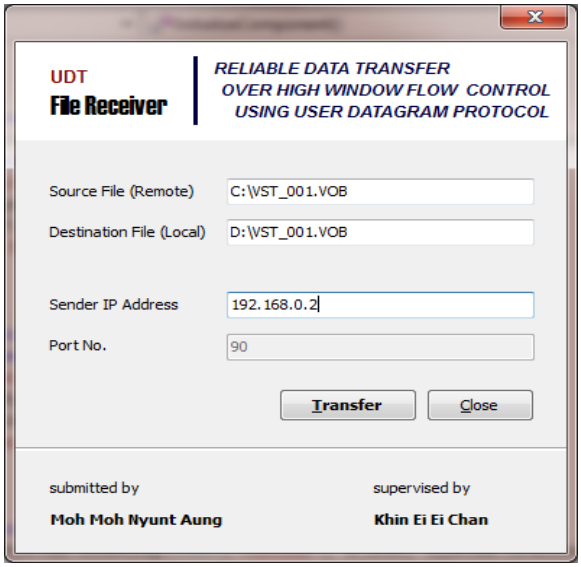

Figure 6. Results for the UDT Receiver

If the connection is established, this system can perform the copying of any unactive files type from remote machine to the local between the two machines. If there is a terminal who want to copying any unactive files that terminal must know one of the terminal's IP address and port number. The UDT Receiver side also needs to know the location where the user wants to copy an unactive file name from remote machine. Continuously, the user also needs to save the file from the remote machine. Therefore, the user gives the destination location of this file. Port number 90 is used as the default port in the system. When the user has been selected to copy an unactive file from the remote machine, it may transfer to the local machine. Besides, the user can give the same as the source file name of the user can give the other file name. The user can see the wanted or renamed file in the location giving the user.

6. Experimental Result

This system is reliable because of using connection oriented UDT. Besides it can transfer large file size within seconds. The experimental result is shown by using table and figure. Table 1 shows the relationship between data transfer bit and transmission time in seconds. By using the data from this table, the plot of time versus transfer bit can be got. By using this system, the data can transfer up to gigabits of file size. It is shown in Figure 7. The more the transmission data, the larger transmission time.

Table 1. Relationship between Data Transfer Bits and Time

Data(Byte)	Time(sec)
45M	8
100M	16
145M	23
200M	28
261M	35
380M	44
437M	53
581M	71
1G	120
1.5G	180

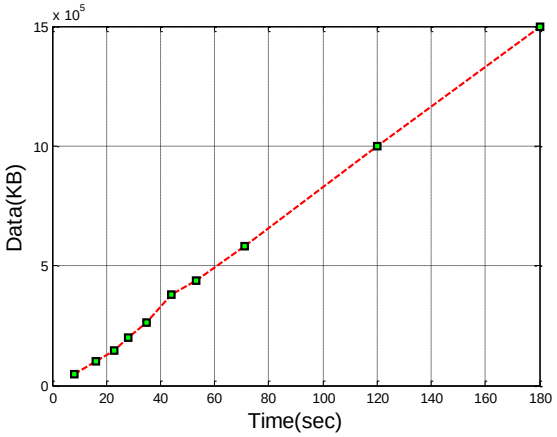

Figure 7. Data Transfer Bits versus Time

7. Conclusion

High-speed data transfer creates many challenges for the design and implementation of different transport protocols. Any additional action on per packet processing can lead to a significant increase in CPU usage, whereas a bursting of CPU usage can further lead to packet loss. Moreover, on long distance links, the number of on flight packets is also huge and requires large data storage to temporally record their information. Access to such data storages is also critical.

This system presented the details of the UDT protocol and in particular those related to high performance data transfer. This proposed system is expected to be useful for future work on transport protocols. The efficiency of using UDT protocol in media file transfer is applicable for the media business. The UDT method 1) provides the use of bandwidth estimation in transport protocols, 2) make useful for other experimental high performance network transport protocols, 3) alternate loss list processing, and bandwidth estimation techniques can be tested. User Datagram

Transport Protocol (UDT) takes good use of available bandwidth. It is intra-protocol fairness and TCP friendliness. And it can be used in open source. This technology used in this paper is computer network protocol based (UDP/IP) technology. Any files can be transferred between the two users.

8. Limitations

The system is designed for Local Area Network (LAN), therefore it will be slow on WAN or Internet. It is intended for any unactive files transfer. It does not calculate the efficiency because this system is major for any unactive files transferring with the less packet loss. This system cannot be transferred folder. This system will use only the port 90. So, other ports numbers are not used in this system. The Visual Studio 2005 must have for this system.

9. References

[1] Briscoe, N."Understanding the OSI 7-Layer Model"

[2] Comer, D. E."Internetworking with TCP/IP"Prientice Hall, Inc.2000. ISBN: 0-13-01935

[3] Fairhurst, G."Unicast, Broadcast and Multicast" Online document.
http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/ip.html

[4] Postel, J.,"RFC 959-File Transfer Protocol, chapter 8" Online document.
http://tools.ietf.org/html/rfc959. 2009-01-04.

[5] "User Datagram Protocol"
Online document.
http://en.wikipedia.org/wiki/User_Datagram_Protocol

[6] Yunhong Gu and Robert L. Grossman,
"UDT: UDP-based Data Transfer for High-Speed Wide Area Networks" Computer Networks (Elsevier). Volume 51, Issue 7.May 2007.

[7] Yunhong Gu, B.E.,
"UDT: A high performance data transfer protocol"