A UDP-based Way to Improve Data Transmission Reliability

Xinhong HEI, Jia CHEN, Hongtao LU, Guo XIE, Haining MENG

Xi'an University of Technology, Xi'an 710048, China

E-mail: heixinhong@xaut.edu.cn

Abstract: In real-time control systems, a fast and reliable data transmission mechanism is necessary. This paper analyzes the advantages and disadvantages of the existing data transmission protocol and proposes a data transmission protocol called Deque-ERUDP (Deque Efficient and Reliable Protocol Based on UDP) which can guarantee data transmission reliability and efficiency. The proposed protocol uses double sub-queue data transmission and acknowledgment mechanism. This protocol controls TIQ (Timeout Interval of Queue) and TIP (Timeout Interval of Packet) dynamically to avoid network congestion. By comparing the results of simulation experiment, it really verifies the feasibility of Deque-ERUDP and improves the reliability of data transmission very well.

Key words: UDP, reliability, efficiency, Deque-ERUDP, congestion control

1 INTRODUCTION

With the development of network transmission technology, the efficient and reliable point-to-point communication systems are increasingly and widely used. The performance of network transmission in real-time control system becomes more and more important.

Now, there are three common transfer protocols for network data transmission, including TCP [1], SCTP [2] and UDP [3]. TCP and SCTP protocol are reliable and connection-oriented transport layer protocol, which has a special delivery assurance mechanism to ensure data transmission with high reliability characteristics. But their control mechanism are complicated, inefficient, wasting of resource and cannot support multicast delivery; UDP protocol has poor reliability. It's a connectionless-oriented transport layer protocol and support multicast delivery. It increases the efficiency of data transmission because it does not have complex delivery mechanism to guarantee the reliability of network data transmission [4] [5]. It's easy to see that TCP and SCTP protocol are difficult to meet the requirements of data transmission's high efficiency as UDP protocol cannot meet the requirements of its high reliability. In terms of network data transmission, it is significant to study data transmission reliability and efficiency of these two seemingly contradictory problems.

Today, there have been many reliable data transfer UDP-based protocols. Reference [5] proposed an improved protocol architecture called RUDP, which uses data transmission mechanism like TCP's timeout and retransmission mechanism. Compared to TCP, RUDP's mechanism is much simpler, but it's unsuitable for big data transmission because of its long waiting delay. Reference [6] [7] proposed RUDP protocol structure using a queue to store the sent packets temporarily. The sender sends a data packet, then the packet goes into the queue and the sender continues to send the next data packet rather than wait to receive the acknowledgment packet. So this protocol save much time by eliminating every data packet's confirmation time to improve the efficiency of data transmission. Reference [8] proposed a protocol which uses a batch acknowledgment mechanism. The receiver sends an acknowledgment for a group of packets after receiving a certain number of packets. To some extent, the protocol improve data transmission efficiency, but it still uses a stop-and-wait mechanism and waste a lot of time.

According to the protocol's deficiency, this paper proposes a protocol called Deque-ERUDP (Deque Efficient and Reliable Protocol Based on UDP) whose data packet's sending and acknowledgment is based on double queue. The proposed protocol uses a new control method to avoid network congestion. Finally, experiments show its feasibility.

This work is supported by National Natural Science Foundation of China (No.U1334211, No.U1534208, No .6160237), Key Laboratory Scientific Research Program of Shaanxi Provincial Department of Education (15JS078), Science and Technology Innovation Project of Shaanxi Province (2015KTZDGY0104).

2 PROTOCOL STRUCTURE

From the perspective of computer network system, the structure of Deque-ERUDP is shown in figure 1. In the TCP/IP four-layer model, a new layer is added between the application layer and the transport layer. So the traditional model becomes a five-layer network architecture. The function of new intermediate layer has the mechanism of data packet subcontracting, recognition, retransmission, recombinant and other congestion control mechanism to ensure the reliability of data transmission. The intermediate layer does not complete the transfer of data, the actual of data transmission is still completed by transport layer protocol of UDP. The newly formed five-layer network architecture not only ensure the reliability of data transmission, but also retain the UDP data transmission in a highly efficient status.

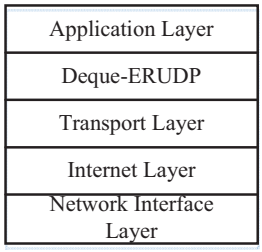

Fig 1. The protocol structure.

3 Deque-ERUDP MECHANISM

Deque-ERUDP uses a combination of multiple mechanisms to ensure the reliability of data transmission. We use a simple three-way handshake mechanism to establish the connection between the sender and receiver. There are two identical data buffer-queue in both sender and receiver. After the connection is prepared, the sender sends the packets to the two data buffer-queue. In the data buffer-queue, Deque-ERUDP adopts the queue-partition recognition and the queue-partition reorganization mechanism. In both sides, the data in each buffer-queue include the buffer-queue name, data number flag (the packet start number and end number), over-time variability, time-out flag, packets loss flag, packets retransmission data area and packets send data area. The retransmission data area stores the packet number to be retransmitted, and the sending data area stores the packet number of the current queue to be sent. Similarity, the receiver also has the same buffer-queue used to store the received data packets, and then extract the received data packets and reorder the unordered packets.

3.1 Connection Establish

The middle-layer Deque-ERUDP is a connection-oriented protocol to ensure the reliability. As shown in figure 2, it uses like TCP's three-way handshake to establish a connection before the data start to transfer between the sender and receiver. The receiver is treated as a server when a connection is established. The sender sends the handshake packet when it needs to establish a connection with the receiver, and then the timer starts to count. Before the sender receives the receiver's handshake response packet successfully or the timer doesn't arrive at the limit time, the sender will send a handshake packet to the server at regular intervals. In order to prevent the response packet losing between the receiver and the sender, the receiver will send a handshake response packet to the sender once it receives a handshake packet. The sender will enter the data sending or receiving state when it receives a successful handshake response packet and it will ignore any other handshake response packets once it's in sending or receiving state.

When the connection needs to be closed, one side will send a closed packet to the other side, then the other side will close the connection and release the resource. Due to the closed packet is sent only once through the UDP, so we cannot ensure that the closed packet can be received successfully by the other end. To solve this problem, it uses the category of time overflow to close the connection. It sends a heartbeat packet to the other side regularly. Then no packet received by the other side in a short time can be the condition to detect whether the connection is closed or not.

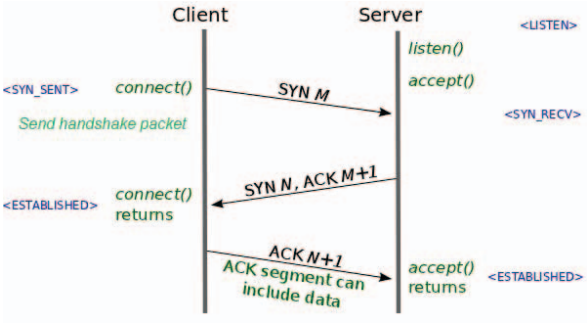

Fig 2. Connection establish.

3.2 The Process of Queue Data Distribution

Load the sent data packets into two queues after the connection is established and set the flag of the two queues, the data distribution of each queue is shown as figure 3.

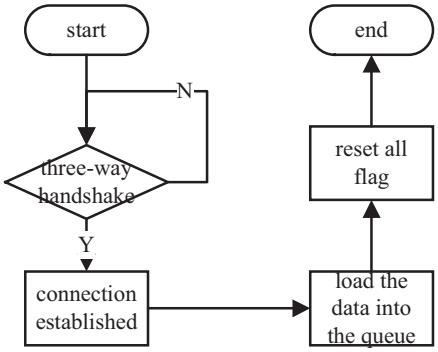

Fig 3. Flow chart of the process of the queue data distribution.

According to the figure, the main idea of data distribution is the process of data packets loaded into the two queues. The sender side need to reset the queue all flags and clear over-timer variability after the each queue has been full of data packets. The sender has the priority order as queue D1, D2 to load the data. Data distribution and data sending is a synchronous process. Data sending process will start only after the connection is established and both of queues must be loaded data packets firstly. Before each queue starts to send the data packets, first, the buffer-queue name type will be set 1; second, data number flag will be marked on packet start number and end number, third, timer variable start to count; then, timeout flag is set to 0 and packet loss flag is set to 0; finally, the retransmission data area is filled with 0 and the send data area is filled with data packets between start number and end number. When the link is in a high rate, D1 will receive confirmation packet from receiver quickly during the data distribution process. When loading the data packets for the second time, if D1 has received the confirmation packet, the queue D1 will be still distributed data only and D2 will be in an idle state. When the link is in a low rate, D1 will not be able to get confirmed timely so that it cannot be distributed data. At this time we can distribute data to D2 and it can reduce the cost of D1.

3.3 The Process of Queue Data Send

The data is sent in accordance with the order of the queue. Every time we must scan the double queue data so that both of the queue data packets can be sent completely. The specific process shown in figure 4.

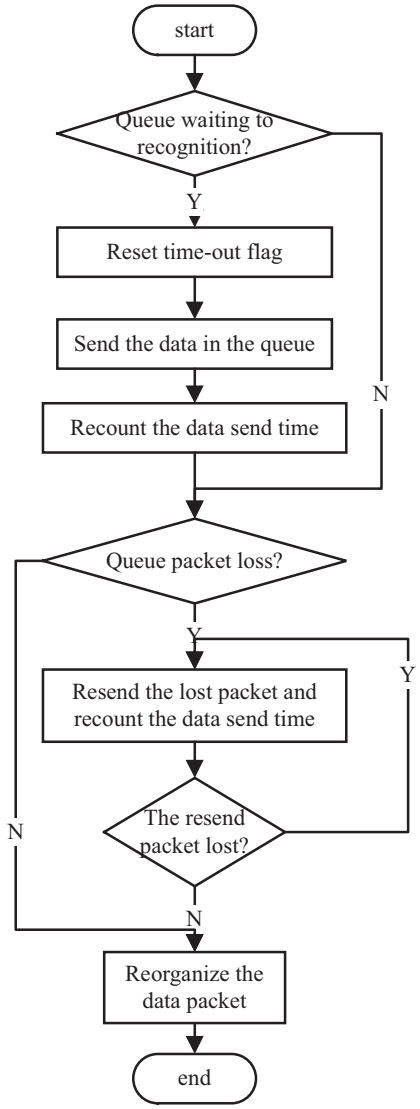

Fig 4. Flow chart of the process of the queue data send.

On the data sender side, if the queue over-time variability reach the limit, the time-out flag will be set to 0 and this queue continue to send the data. The queue over-time variability restarts to count time. On the data receiver side, the receiver will check queue data packets number. If the queue data packet number is missing or duplicate or disordered, the receiver will response as follows: first, the duplication of data packets will be discarded; second, the lost data packets will be extracted and put into retransmission area; last, the receiver will inform the sender resend lost data packets. Therefore, the receiver informs the sender to resend the missing data

packets according to the each queue. When the retransmission data packets arrive the receiver again, the receiver uses data reorganization algorithm to reorganize the data packets. Compared with TCP, this can greatly reduce times of retransmission and confirmation, save the data link source and improve the efficiency of the data transmission.

3.4 The Process of Queue Data Confirmation

The sender judges the response packet type whether the received data packet is ACK or NCK. In this strategy, 0 represents the ACK packet and 1 represents the NCK packet. When the queue data packets are all received by receiver, the receiver will process the received packets and then send an ACK packet to the sender side. At the same time, the sender clears the queue and resets all flags; if the queue data is fully received, but the queue time-out of receiver reaches a certain threshold, the receiver will send a NCK packet to the sender. The NCK packet contains the queue name and the lost packets number. The NCK packet tells the sender to resend the lost packet. By judging whether the received acknowledgment packet is ACK or NCK, the sender side decides whether to clear the queue data or resend it. When the received packet is ACK, the sender clears the queue and waits for the next time to allocate data for the queue. If it receives the NCK acknowledgment packet, the sender extracts the packet from the queue and gets the lost packet number, then resend the lost data. The queue data confirmation of the process is shown in figure 5.

4 THE CONGESTION CONTROL MECHANISM

Deque-ERUDP adopts to controls TIQ (Timeout Interval of Queue) and TIP (Timeout Interval of Packet) dynamically to avoid link congestion. After the queue is allocated data (for the first time), when the data in the queue will be sent to the receiver is determined by the value of TIQ. If the value of TIP is 0, the data will be sent immediately. Therefore, the value of different TIP can control the queue data transmission rate, the greater the value, the smaller the transmission rate.

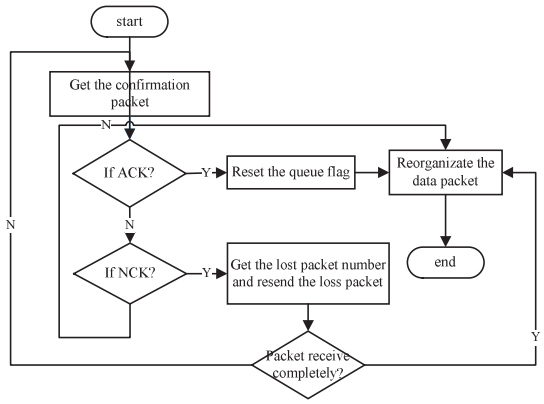

Fig 5. Flow chart of the process of the queue data confirmation.

The TIQ is adjusted by the number of lost packets. We can set a threshold value according to the network load. As shown in formula (1), if the number of lost packets is less than or equal to the threshold, the value of TIQ will be increased in value of 1.5 times RTT; as shown in formula (2), if the number of lost packets is greater than the set threshold, the new value of TIQ will be assigned weighted mean of TIQ. TIQ_{old} represents the last value of TIQ and TIQ_{new} represents the new value of TIQ. According to the reference [9], the value of α equals to 1/8.

$$\mathrm{TIQ} = 1.5 \times 2 \times \left(T_{receive} - T_{send}\right) \tag{1}$$

$$\mathrm{TIQ} = 0.5 \times \left(\left(1-\alpha\right) \times \mathrm{TIQ}_{old} + \alpha \times \mathrm{TIQ}_{new}\right) \tag{2}$$

The traditional TCP protocol mostly uses a sliding window mechanism algorithm called Additive Increase Multiplicative Decrease (AIMD) [10] to avoid link congestion. In this paper, a mechanism based on queue congestion control is adopted, that is, MIMD (Multiplicative Increase Multiplicative Decrease). The basic idea of MIMD is that the rate of data transmission in the queue is increasing while the increment of the data transmission rate is decreasing during the congestion avoidance. That is, the data transmission rate in the queue is still increasing, but the increase rate will be slower and slower. With the data in the network link increasing, the network tends to be congested, so reduce the incensement in network can avoid link congestion in a certain level. The comparison chart between MIMD and AIMD is shown in figure 6. It can be seen from the figure that the data transmission rate of the MIMD algorithm increases

rapidly at the beginning of the congestion avoidance phase. With the data in network link continues to increase, In order to avoid the congestion, the growth trend of the rate tends to be gradual and at the same time the network can be unblocked.

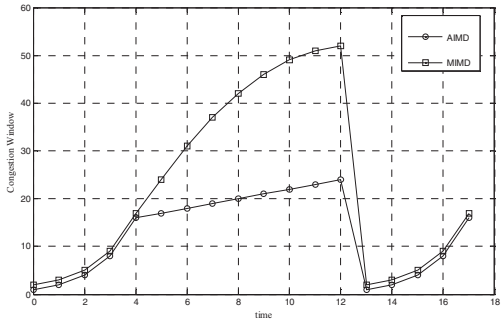

Fig 6. The comparison between AIMD and MIMD.

5 PERFORMANCE EVALUATION AND ANALYSIS

This experiment uses NS2 [11] as the experimental simulation platform. The network topology is shown in figure 7. The bottleneck link between Router 1 and Router 2 is Drop Tail. The link bandwidth is set to 2 Mbps and the delay is 50 MS. The link bandwidth on both sides is set to 10 Mbps, and the delay is 20 MS. The router cache is 100. The network nodes adopts TCP, UDP and Deque-ERUDP to be the data streams. The three data streams must be guaranteed to be in the same environment.

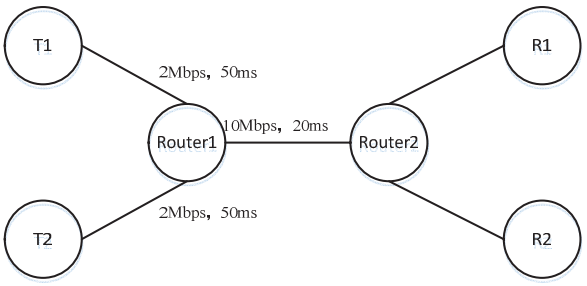

Fig 7. The Network Topology.

As shown in figure 8, both the throughput of TCP and Deque-ERUDP increase exponentially during the initial phase of data transmission. With the data increasing in the link, owing to Deque-ERUDP using a double-queue transmission confirmation mechanism, Deque-ERUDP obtains a significant better throughput than TCP. Its throughput keep increasing for a long time and tends to a stable state finally.

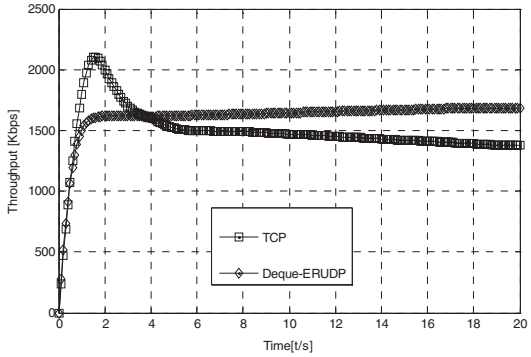

Fig 8. The comparison throughput between TCP and Deque-ERUDP.

Through measuring and comparing the RTT values of the two protocols. As shown in figure 9, it can be seen that Deque-ERUDP adopting complicated mechanism to data transmission, data confirmation and data retransmission will not increase the whole data round trip time.

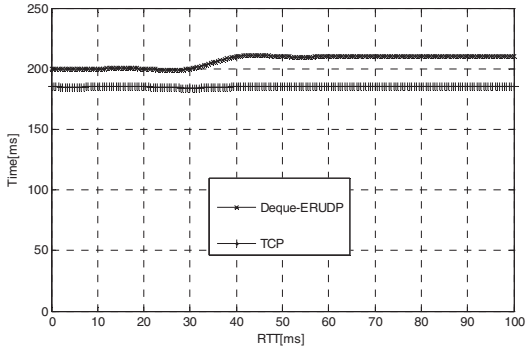

Fig 9. The comparison RTT between TCP and Deque-ERUDP.

Keep the simulation environment unchanged and TCP will be replaced by UDP. The throughput and packet loss rate are shown in figure 10 and figure 11. From the comparison of UDP and Deque-ERUDP, we can see that Deque-ERUDP protocol, which is similar to TCP's guarantee of transmission reliability mechanism, has higher throughput obviously. However, UDP with no guarantee reliability mechanism has high packet loss rate but the throughput is very low. This proves that Deque-ERUDP protocol greatly improves the reliability and efficiency of data transmission.

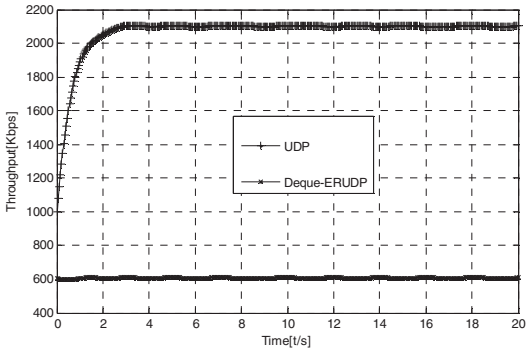

Fig 10. The comparison RTT between UDP and Deque-ERUDP.

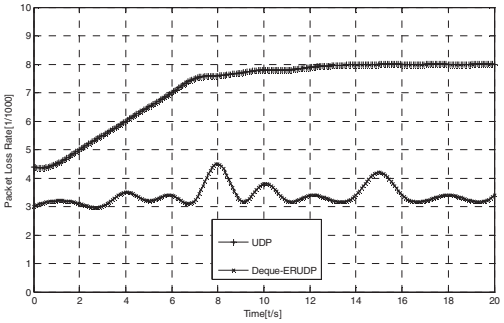

Fig 11. The comparison packet loss rate between UDP and Deque-ERUDP.

6 SUMMARY

In this paper, we propose an improved protocol Deque-ERUDP based on the traditional UDP protocol which adopts double-queue mechanism to finish data confirming and data retransmission. Deque-ERUDP effectively solves the problem that the packet transmission wastes too much bandwidth and low reliability of data transmission. This experiment is based on NS2 platform simulation. In terms of avoiding congestion, we achieve the goal of congestion avoidance by controlling TIQ and TIP dynamically. The experiment shows that Deque-ERUDP really saves the resources of network bandwidth and improves the reliability and efficiency of data transmission.

REFERENCES

[1] J. Postel. Transmission Control Protocol. RFC793. September.

[2] Stewart R. Stream Control Transmission Protocol[S]. RFC 4960, Internet Engineering Task Force, 2007.

[3] J. Postel. User Datagram Protocol. RFC768. August 1980.

[4] Wang Jigang, Gu Guochang, Xu Lifeng, Wang Chen. Research and Design of Reliable Data Transfer Based on UDP [J]. Computer Engineering and Applications, 2006, (15):113-116.

[5] YAN Huan, GAO De-yun, SONG Fei. Performance Evaluation of SCTP-Based Concurrent Multipath Transfer [J] .Computer Technology and Development, 2010, (11):29-32+41.

[6] ZHOU Jin-cai. The designing thinking and achieving method of Reliable UDP protocol [J] .Journal of zhoukou normal university, 2006, (02):104-108.

[7] L. Guo and L. Chengtong, "Research and Achievement of a Way to Improve the Data Transmission Reliability of UDP," 2012 International Conference on Computer Science and Service System, Nanjing, 2012, pp. 627-630.

[8] WANG Hai-jun, LIU Cai-xia, CHENG Dong-nian. Analysis and Research of a Reliable Transmission Protocol Based on UDP [J] .Application Research of Computers, 2005, (11):181-183.

[9] V. Paxson, M. Allman. Computing TCP's Retransmission Timer. RFC2988. November 2000.

[10] LUO Wan-Ming, LIN Chuang, YAN Bao-Ping. A Survey of Congestion Control in the Internet [J] .Chinese Journal of Computers, 2001, (01):1-18.

[11] NS2, Network Simulator Version 2, http://www.isi.edu/nsnam/.