Protocol Software Engineering

Protocol Design Document

Student Number: 013338170
Student Name: Yi Ding

Table of contents

Abstract

This document proposes an experimental UDP based data transfer protocol called as Reliable User Datagram Protocol (RUDP). It aims at providing reliability and flow control to the data delivery operating on top of UDP. In RUDP, a separate protocol stack layer is introduced to provide a unified API for applications. The transmission control algorithms on both sending and receiving sides guarantee the RUDP capable of recovering from data loss, duplication, delay, and as well as reordering.

1. Introduction

The User Datagram Protocol (UDP) [RFC768] adopted in current Internet, is one of the core protocols within the Internet protocol suite. It is a minimal message-oriented transport layer protocol, providing a simple interface between network layer and application layer. In UDP, senders will not retain state information for the packets sending out. The delivery efficiency is achieved by avoiding the overheads from checking whether every packet actually arrives at the receiving side. Therefore, UDP provides no guarantees to the upper layer protocol of the reliability, flow control, or ordering control.

To help achieve reliability and flow control of data delivery, we design a UDP based data transfer protocol, the RUDP, which operates completely on top of UDP. The RUDP protocol stack architecture is as illustrated in Figure I.

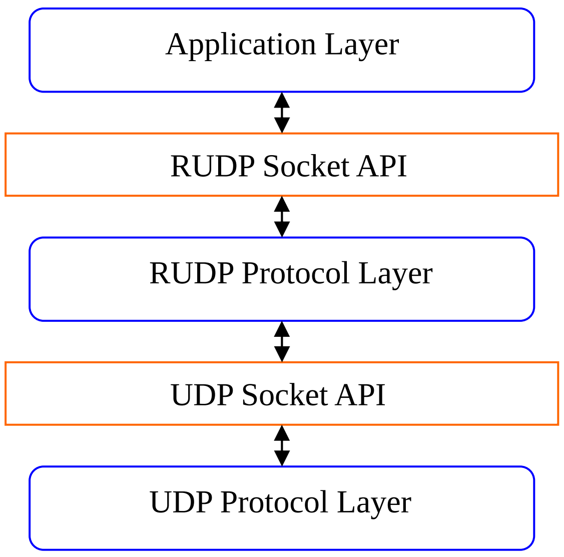

Figure I: RUDP protocol stack architecture.

The RUDP protocol layer makes use of UDP through the UDP socket API provided by the operating system. RUDP provides an interface as well to the application layer above. Applications use RUDP socket API normally in the same way as calling system socket API. When RUDP is adopted, applications will exchange data through

RUDP socket, which further uses the UDP socket to send and receive data.

RUDP is a connection-oriented and unicast protocol, in which both data and control packets are transferred on top of UDP. The connection-oriented feature of RUDP helps to maintain the flow control and reliability of data transfer. RUDP is a unicast protocol, while the multicast and broadcast features are not included for the simplicity reason.

This protocol design document defines the RUDP protocol specification and reflects the structure of final implementation. It covers the packet structure, timer, connection setup & shutdown, data sending & receiving, and security issues related to RUDP. If particular aspects of the specification described in the document are deemed as infeasible during the implementation phase, the reworking of relevant parts will be undertaken, so that the whole project can be completed in due time.

2. Packet structure

In RUDP, a RUDP header is introduced based on UDP. The newly defined RUDP header is inserted between the normal UDP header and application data payload. The RUDP header provides necessary transmission control features to achieve reliability and flow control of data delivery. Its datagram layout is presented in Figure II.

IP header	UDP header	RUDP header	Data payload

Figure II: RUDP datagram layout.

Two kinds of packets are used in RUDP: 1) data packet and 2) control packet. They are distinguished by the flag bit in the RUDP header of each packet, where 0 stands for the data packet and 1 stands for the control packet. Figure III described the general header structure for RUDP.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|                further defined according to type            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure III: RUDP general header structure, where ‘X’ is the flag bit.

2.1. Data packet

The data packet in RUDP is denoted by the first bit of 0 in its header. Hence, every data packet header starts with 0. The header structure of a data packet is illustrated in Figure IV.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|                  Packet Sequence Number                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PF |I|                  Message Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         *Time Stamp*                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure IV: RUDP data packet header structure.

With the starting bit 0, the following 31 bits after the flag bit are reserved for packet sequence number. In RUDP, packet-based sequencing is used that the sequence number will be increased by 1 for each data packet sent out. The sequence number will be wrapped once it reaches the maximum number (2^31 – 1).

The following 32-bit field is for the messaging purpose. The two bits "PF" flag is to indicate the position of a packet to the message, to which this packet belongs. When PF=10, the packet is the first one in the message while PF=01 means the last one. PF=11 indicates that there is only one packet for the message, and PF=00 means any packet in the middle. The flag "I" conveys whether the message should be delivered in order or not, with 1 standing for in order and 0 for not in order. The in order for a message delivery requires that if in order, all previous messages must be either delivered or dropped. The rest 29 bits of message number is similar to packet sequence number but independent with each other – a RUDP message may contain multiple RUDP packets.

The next 32 bits time stamp field is an option part in data packet. It is used only when user-defined control algorithms will require the timing information. The time stamp contains a relative value which starts as the connection is set up.

2.2. Control packet

The RUDP control packet starts with the bit 1. It contains ACK sequence number field, type field, message number field, control information field, and also time stamp field. When a control packet arrives, it will be parsed according to the header structure in Figure V.

The 31-bit ACK sequence number in control packet is a matching for the sequence number field in data packet.

The following 3-bit type field will determine the contents appearing in the control information field. The types of control packets include: 1) 000, connection handshake;

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1|                  ACK Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| type|                  Message Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                 Control Information Field                     ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        *Time Stamp*                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure V: RUDP control packet header structure.

2) 111, connection shutdown; 3) 001, acknowledgement (ACK); 4) 100, keep alive; and 5) 010, acknowledgement for acknowledgement (ACK 2). The rest type options are reserved for further usage.

The next 29-bit message number field is also a counterpart for message number in data packet. The variable length control information field is used to support reliability and flow control for the data transmission. Finally, the optional time stamp field exists in the control packet in accordance with the data packet.

The RUDP Type-To-Control Information mapping is listed as the following:
1) Type 000: connection handshake
   Control information field: initial sequence number; packet size; and maximum flow window size (for flow control).
2) Type 111: shutdown
   Control information field: None
3) Type 001: ACK
   Control information field: packet sequence number to which all the previous packets have been received; available buffer size (for flow control).
4) Type 100: keep alive
   Control information field: None
5) Type 010: ACK 2
   Control information field: ACK sequence number.

3. Timers

In RUDP, two types of timers are adopted to trigger different periodical events. Each RUDP timer has its specific period and operates independently with microseconds as the granularity.

On the sender side, a RESEND timer is used to trigger the retransmission of a lost data packet if the loss is detected. The RESEND timer plays a key role to guarantee the data transfer reliability. On the receiver side, an EXP timer is used to maintain connection status as well as to trigger the sending out of duplicated ACK. The EXP timer is also used by senders to avoid infinite retransmissions or reconnections.

4. Connection setup and shutdown

RUDP adopts the typical client/server mode to setup and shutdown each connection session.

4.1. Connection setup

To setup an RUDP connection, one entity starts at first as a server to listen to a predefined port. The server will accept and process incoming connection request and create new socket for each new session.

To connect to the server, client side should send a handshake packet first. The contents of a handshake packet are defined in Section 2.2 as type 000. The client should keep on sending the handshake request until it receives a response handshake or an EXP timer expires.

On the arrival of a handshake packet, the server will compare the packet size and maximum window size with its own values. It then sets the corresponding parts to the smaller values. The result values are sent back to the client by a response handshake packet. After receiving a response handshake packet, the server is ready for data transmission. Nevertheless, the server should send back response packet as long as it receives any further handshakes from the same client. The purpose of this action is to guarantee connection reliability.

The client can start data transmission once it receives a response handshake packet from the server to which the client want to connect. The initial sequence number information is exchanged during the handshake phase. Further response packet should be omitted after the first response packet.

4.2. Connection shutdown

If one side of connected entities wants to terminate the connection, it will send a shutdown packet to the peer side. After receiving the shutdown packet, the peer side will be closed. The shutdown message is sent out only once and will not guarantee to be received. If this message is lost, the peer side will be closed after a certain number of EXP timer timeout. The default number for the shutdown EXP is 5 and it could be modified by applications to adjust to various environment requirements.

5. Data sending and receiving

Two logical parts of the RUDP session include the sender and the receiver: the sender sends out application data and carries out retransmission according to the RUDP transmission control mechanism; one the other hand, the receiver receives both data and control packets, and replies with control packets.

5.1. Sending algorithm

In RUDP, the sending side uses the following data structures and variables:

1) Sender buffer: a ring buffer used to store packets that are sent out but unacknowledged yet. The buffer size determines how many packets can be sent from the sending side at one time.
2) RESEND timer: a timer controlling the retransmission. Only one RESEND timer is attached to the first packet in the sender buffer. When timer timeout, it triggers the retransmission of the first unacknowledged packet in the sender buffer.
3) EXP timer: a timer to maintain connection status. A keep alive packet is periodically sent when the EXP timer timeouts. If there is no response after a certain number of EXP timer timeout, the connection will be closed.

Algorithm description:

The sending process starts when there is data coming from the application layer. RUDP will pack the data and send it out, and at the same time, the packet will be stored in the sender buffer waiting for acknowledgement. The RESEND timer starts to count once there is one packet in the sender buffer. After the buffer is fully occupied, any further data coming from applications would be rejected, and an error message will be sent to the upper layer to notice the buffer situation. When ACK packet arrives to acknowledge a data packet, the sequence number is compared. The corresponding data packet will be removed from the sender buffer after receiving ACK. If there is space in the sender buffer, data from application layer could be sent out and also stored in the buffer.

The retransmission is triggered by two events: 1) when RESEND timer expires; 2) after receiving continuously triple duplicated ACK on the same sequence number. The first packet in the sender buffer will be resent again when the retransmission event occurs. In RUDP, cumulative acknowledgement is used to avoid unnecessary retransmission of packets. It indicates that the sender will recognize the latest ACK sequence number it receives. In the scenario that if there are two ACK packets from the receiving side, the first ACK is lost, and only the second one with larger sequence number arrives, the sender will accept the latest ACK knowing that the receiver has received everything up to the sequence number contained in the latest ACK.

5.2. Receiving algorithm

In RUDP, the receiving side uses the following data structures and variables:

1) Receiver buffer: a ring buffer holding the disordered the packet. The disordered packet indicates the sequence number is larger than the expected value, which implies the happening of transmission reordering. The buffer size determines how many disordered packets can be held by the receiver. Once the buffer is full, any arriving disordered packet will trigger the sending of ACK packet.
2) EXP timer: a timer to maintain connection status and trigger ACK retransmission. An ACK packet is periodically sent when the EXP timer timeouts. If there is no response after a certain number of EXP timer timeout, the connection will be closed.

Algorithm description:

Receiving side will reply ACK packet if the coming data packet matches the expected sequence number. If the sequence number is smaller than expected, the data packet is dropped. At the same time a corresponding ACK will be sent out immediately indicating the latest received data packet sequence number. If the sequence number is greater than expected, the data packet is stored in the receiver buffer and as well an ACK packet will be sent. When the retransmitted data packet finally arrives, the receiver will check the receiver buffer to clear all the consecutive packets that can be accepted according to the cumulative sequence order. If the receiver buffer is full, any arriving disordered packet will be dropped and an ACK will be sent.

An EXP timer timeout is an active triggering event for the receiver to send out ACK packet. It guarantees the data transmission efficiency and avoids the sender to enter RESEND timer timeout phase too quickly. The EXP timer will stop once the shutdown packet arrives at the receiver.

5.3. Flow control

Flow control in RUDP is to eliminate the possibility of letting the sender to overflow the receiver’s buffer. For every ACK packet received, synchronization operation will occur that the sender will update the window size to match with the receiver’s available buffer size. This operation limits the sender’s capacity to send out a large number of packets at one time.

5.4. Considerations of data loss, duplication, delay, and reordering

Our RUDP design targets at providing reliability and flow control to the data transfer on top of UDP. It can help to recover from the following error events: 1) data loss; 2) duplication; 3) transmission delay; 4) reordering.

The data loss and duplication errors are prevented by using the sequence number and retransmission algorithms on both the sending and receiving sides. The sequence number helps in determining the position of every packet. Combined with ACK sequence number, the loss and duplication can be detected, and retransmission will take place to recover from such errors. Timer is used to guarantee the triggering of retransmission hence the data packet that is lost will be resent.

The transmission delay is prevented by the retransmission algorithms. When a packet happens to be routed in the network for too long, the sender will simply retransmit the packet again. The receiver only accepts the first arrived packet according to the expected sequence number hence eliminating the happening of delay error.

The reordering is handled by introducing the receiver buffer at the receiving side. The disordered packets are buffered, and when the missing packet gaps are filled by the correct packets, the buffer is emptied. The sequence number is used to location the position of each packet to check the data validity.

In summary, the cooperation of transmission algorithms at both sending and receiving side provides the reliability of data transfer. The RUDP is therefore capable of overcoming the error events including data loss, duplication, delay, and as well as reordering.

6. Security considerations

The RUDP is an experimental protocol and does not provide specific security mechanism. However, it could depend on the application layer or lower layers to obtain authentication and security supports. Since UDP is a connectionless protocol, RUDP implementation will check that all the packets arrived should come from the expected source.

Reference

Nominative:
[UDP] J. Postel, RFC 768: User Datagram Protocol, Aug. 1980.

Information:
[UDT] Y. Gu, UDP-based data transfer for high-speed wide area network, Dec. 2006.

Acknowledgement

Dedicated to Professor Kimmo Raatikainen who was the teacher of this course last year. Special thanks go to Mrs. Tiina Niklander for her sincere supports, and to Mr. Tan from TKK for the help and useful tips on the protocol design.