Design Document for Reliable UDP

SAFE UDP

Group 1

Members	*Student ID*
Radhesh Sarma	2017B4A70886H
Simran Sahni	2017B5A70856H
Kotikalapudi Venkat Karthik	2017B4A70927H
Ashi Sinha	2017B5A71149H
Chatrik Singh Mangat	2017B5A70822H

Group 2

Members	*Student ID*
Danish Mohammad	2018A7PS0103H
Anirudh Sood	2018A7PS0673H
Utkarsh Jha	2018A7PS0100H
Devendra Dheeraj Gupta Sanagapalli	2017B5A70670H
Vishnu Vardhan Reddy Ireddy	2017B3A70842H

Birla Institute of Technology and Science, Pilani
Hyderabad Campus

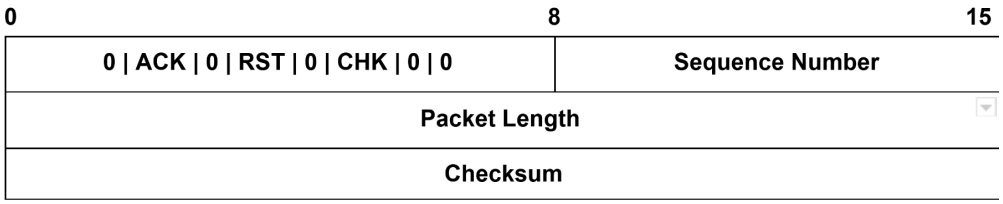
Contents

1. Introduction

Safe UDP (SUDP))is a reliable protocol that uses UDP at the transport layer. We make our protocol reliable by adding metadata as an application-layer header. It uses sequence numbers to number packets to keep track of the packets and acknowledgement to confirm that packet has been received. An octet of single bits is used to control the configuration of data transmission.

2. Overview of the Protocol

2.1 Data Structure Format



A minimum of six-byte header is required for data transmission though SUDP. The first byte consists of a series of single-bit flags. Then we have three byte-sized fields- Header length, sequence number, acknowledgment number, and finally a 2-byte checksum.

2.1.1 Control Bits

Different bits indicate the contents of the packet. These form the heart of the RUDP and signify the further actions in the connection. We list the control bits below:

1. **ACK bit:** Indicates whether or not the packet is an acknowledgment.
2. **RST bit:** Indicates whether or not a packet is a reset segment.
3. **CHK bit:** Indicates whether checksum field contains the checksum of just the header or header and the field.

2.1.2 Packet Length

It is a 16 bit number that indicates the length of the packet. User data will exist in the packet if the length of the packet is greater than the length of the header field. In the absence of user data, the length will be 6 bytes. A UDP packet containing user data is called a data segment. The ACK bit will

always be set for such a packet whereas packets with RST bits set, cannot store user data. With 16 bits, the total length of the packet can be at most 2^{16} bits = 8 KB.

2.1.3 Sequence Number

It is an 8 bit number that is used to order the packets correctly. The sequence number for both the sender and receiver starts at 0 and since it is an 8 bit number, it ends at $2^8 - 1$. Every packet number till $2^8 - 1$ has a sequence number one higher than the last. The $(2^8 - 1)^{th}$ packet's successor will have packet number 0.

2.1.4 Checksum

The algorithm used is the same as mentioned in RFC 793.

2.2 Segment Descriptions

2.2.1 ACK Segment

A packet is considered to ba a ACK segment when the ACK bit is set to 1. This is sent exclusively from the receiver to the sender in acknowledgement of a packet denoted by the sequence number contained in the header. The packet does not have a body, so the length of the packet is 6 bytes.

2.2.2 RST Segment

A packet is considered to be a RST segment when the RST bit is set to 1. It can be sent by either the sender or the receiver. It is used to close or reset a connection and is a data-less segment. Upon receipt of an RST segment, the sender must stop sending new packets but must continue to attempt delivery of packets already accepted from the API.

2.2.3 Data Segment

A packet is a data segment when both the ACK and the RST bits are 0. These packets are sent exclusively from the sender to the receiver. They contain data that the sender wants to transfer to the receiver.

2.3 Handling Special Cases

1. **Retransmission:** They are taken care of by a timer and a counter present in every end node, with the counter value, being negotiated upon for every connection initialized. The retransmission timer is restarted every time a NUL, RST, or Data segment is received.

2. **Packet loss/ ARQ:** In case a packet is lost in the network, it is retransmitted using the selective repeat mechanism.

3. **Out-of-order transmission:** The receiver maintains a buffer of all the segments that have arrived out-of-sequence. This buffer has a maximum size given by window size, N.

4. **Flow control:** The size of the receiver's input queue is a configurable parameter. The recommended value of the receiver input queue size is 32 packets.

3. Selective Repeat

3.1 Sender

W is the sequence segment whose maximum size is MSS. The window size is N (N $<$ MSS). Each cell in W has 3 states:

1. *"Unsent":* The message represented by the sequence number of the cell has not yet been sent.
2. *"Sent":* The message represented by the sequence number of the cell has been sent by the sender but an acknowledgment has not yet been received.
3. *"Ack":* The message represented by the sequence number of the cell has been sent and acknowledged by the receiver.

The window behaves as if the sequence segment is infinite by looping to the beginning of the sequence segment when it reaches the end. The initial state of all cells is "Unsent".

3.1.1 Sender Scenarios

Application gives a message to be sent

- Sender sends the packet with the next available sequence number and changes the state of that cell to "Sent" as long as its window is not full, in which case it informs the receiver of the same.
- Every time a packet is sent, a timer is started.

Acknowledgement is received and the packet is not corrupt

- If the sequence number is within the valid window the timer of the corresponding packet is stopped and the state of the cell is changed to "Ack".

- If the sequence number matches that of an acknowledgment previously received, it is ignored.
- If the packet with the given sequence number has not yet been sent, then an ERROR is raised since the situation should not occur.

Timer goes off

- This means that an acknowledgment of the packet whose timer went off has not been received. In this case, the packet is sent out again and the timer is restarted.

Received packet is corrupt

- The packet is discarded and no further action is taken.

3.1.2 Movement of the Window

- The first cell of the window is the designated `base`.
- The first unsent cell of the window is designated `next_seq_num`.
- Whenever the base cell receives an acknowledgment, the window moves ahead to a "Sent" cell with the smallest next sequence number.

3.2 Receiver

W' is the sequence segment whose maximum size is MSS. The window size is N (N $<$ MSS). Each cell in W has 2 states:

1. *"Not Received":* The message represented by the sequence number of the cell has not yet been received.
2. *"Ack":* The message represented by the sequence number of the cell has been received and an acknowledgment has been sent out for the same.

The window behaves as if the sequence segment is infinite by looping to the beginning of the sequence segment when it reaches the end. The initial state of all cells is "Not Received".

3.2.1 Receiver Scenarios

Packet received and the packet is not corrupt

- If the packet is within the window it is buffered and an acknowledgment is sent out for it. The state of the cell is changed to "Ack"
- Else it is discarded and no further action is taken.

Packet received is corrupt

- It is discarded and no further action is taken.

3.2.2 Movement of the Window

- The first cell of the window is designated the `recv_base`.
- Whenever `recv_base` is acknowledged, the window moves to a "Not Received" cell with the next smallest sequence number.