COMP3234B Computer and Communication Networks

Lab 2: Go-back-N RDT

Introduction

In this lab, we will use Python socket programming to implement a simple application-layer Go-back-N (GBN) RDT protocol. We have introduced in lectures that the stop-and-wait RDT protocol is not efficient enough in sender utilization. In this lab exercise, we are going to implement the GBN protocol between a sender program and a receiver program, which communicates over UDP sockets and allows pipelined data transmission. The simple GBN RDT protocol uses sequence number, timer, ACK, retransmission and a sending window, but without checksum (i.e., we consider the channel may lose data/ACK packets but does not flip bits in the packets).

Lab Exercise: Simple Application-layer GBN RDT

We now implement a simple file transfer application with RDT over UDP, where a sender sends a file to a receiver.

Step 1: Download **lab2_materials.zip** from Moodle. Unzip it and you will find a few files provided: sender/RDTSend.py, sender/tcp.txt, receiver/RDTReceive.py and receiver/compare_files.py.

Step 2: Open **RDTReceive.py** using a text editor. **RDTReceive.py** contains the complete implementation of the receiver program. Study the receiver program carefully and you will learn from its code to complete the server program.

a. The receiver is to be started by command "python3 RDTReceiver.py". By default, you can run both the sender program and the receiver program on the same machine, i.e., using localhost "127.0.0.1"; data packets are sent from the sender's port 6666 to the receiver's port 7777, and ACK packets are sent from the receiver's port 7777 to the sender's port 6666.

b. We use the struct module to pack packet contents into binary data. Check out the functions and format strings in the struct module at: https://docs.python.org/3/library/struct.html. For example, "I" represents an unsigned integer number and "32s" means a 32byte-long string. With "packer = struct.Struct('I')", we create a Struct object which we will use to pack an unsigned integer number; with "unpacker = struct.Struct(' I 32s ')", we create a Struct object which we will use to unpack received binary data into an unsigned integer number and a 32-byte string together. For the usage of pack() and unpack(), you may check out more examples at: https://www.askpython.com/python-modules/python-struct-module.

c. After estabilishing the UDP socket and setting up the packer and unpacker, the receiver keeps receiving data in a While loop. The receiver uses unpack() function to unpack received binary data into a tuple whose entries correspond to the original packed items in the received packet. For data packet, the first entry is the sequence number and the second entry is the transferred data. When the receiver receives a packet with expected sequence number, it writes the content into file "recv.txt", sends corresponding ACK to the sender (if lostACK() function returns false), increases the value of expt_seq_num by 1, and updates latest_seq_num to the sequence number of the currently received packet (unpacked_data[0]). When the receiver receives a packet whose sequence number is not the expected sequence number, it resends the previous

ACK (latest_seq_num) to the sender. The lostACK() function is used to simulate the loss of ACK, i.e., by not sending the ACK packet out.

d. We provided the code to set a 5-second timeout (using socket settimeout() function) on the receiver's socket, i.e., if the receiver is not receiving any data within 5 seconds, the receiver's socket is closed and the receiver program exits. Check out the usage of socket settimeout() at https://docs.python.org/3/library/socket.html#socket.socket.settimeout and https://docs.python.org/3/library/socket.html#socket-timeouts. We also provided a number of print() statements, such that you can observe similar printouts at receiver and sender sides as in the sample screenshots in the appendix.

Step 3: Open RDTSend.py using a text editor, and you will find that it provides a sketch of the sender program. Complete the sender program following the hints given as "#TODO…." in the sendPacket(), recvAck() and funcTimeout() functions, to achieve the following service: The sender reads a file (e.g., tcp.txt) and sends the file contents to the receiver using a protocol similar to GBN RDT sender protocol (except for using checksum). In the sendPacket() function, the sender sends up to WINDOW_SIZE (default to 5) packets to receiver in pipeline, according to the base and next_seq_num pointers. In the recvAck() function, the sender receives ACK from the receiver, updates the base pointer accordingly, and stops the timer. If the value of base pointer is smaller than that of the next_seq_num pointer, it starts a new timer for remaining unacked packets within the sender window. The funcTimeout function handles the timeout event. When it is triggered, it starts a timer and retransmits all unacked packets. As we are running sender and receiver programs on the same machine, it is difficult to observe packet loss. In the sender program, we add a lostPacket() function (similar to the lostACK() function in receiver), which allows us to simulate "loss" of the current packet with a certain probability (default to 0.1), i.e., by not sending the packet out if lostPacket() function returns true.

- The sender reads contents from the file provided and stores them as 32-byte strings into array pkt_data. Learn more about the for… in… loop at https://docs.python.org/3.9/tutorial/controlflow.html and Python single-line for loop at https://blog.teamtreehouse.com/python-single-line-loops, range() function at https://www.w3schools.com/python/ref_func_range.asp, and len() function at https://www.w3schools.com/python/ref_func_len.asp.
- In sendPacket() function, the sender constructs a data packet by packing the sequence number and the 32-byte string, which is converted to bytes first using bytes() function. The Python built-in function bytes() is used to return an immutable bytes object initialized with the given data. Read the official description at https://docs.python.org/3/library/stdtypes.html#bytes. Also you can read more about the encoding methods, e.g., utf-8 at https://stackoverflow.com/questions/2241348/what-is-unicode-utf-8-utf-16.
- The timer is created as a Timer object in the threading module, and started using the start() function. Learn more about threading.Timer at https://docs.python.org/3/library/threading.html#timer-objects. The function funcTimeout is invoked when the timer times out.

Step 4: test your programs as follows:

- **Launch one terminal** and switch to the directory of Lab2/receiver. Run the receiver program as follows:

```
python3 RDTReceive.py
```

- **Launch the second terminal** and switch to the directory of Lab2/sender. Run the sender program as follows:

```
python3 RDTSend.py tcp.txt
```

The next two pages show sample printouts when the default setting of data/ACK loss probabilities are used. You can adjust the data and ACK loss probabilities to simulate the cases of no loss, packet loss only, ACK loss only, as well as different packet and ACK loss probabilities. You may also adjust the timeout times to allow the code to run better on your machine.

After successfully receiving the file from the sender, you should see the new file recv.txt created in the receiver folder with the same content as tcp.txt in the sender folder. Switch to directory Lab2/receiver and run the file comparison program as follows:

```
python3 compare_files.py
```

which will tell you if the received file is the same as the file at the sender side.

Submission:

You should submit the following files in the specified folder structure:
(1) sender/RDTSend.py
(2) sender/tcp.txt
(3) receiver/RDTReceive.py
(4) receiver/compare_files.py

Please compress the above files/folders in a lab2-yourUID.zip file and submit it on Moodle before 23:59 Wednesday Feb. 21, 2024:

(1) Login Moodle.
(2) Find “Labs” in the left column and click “Lab 2”.
(3) Click “Add submission”, browse your .zip file and save it. Done.
(4) You will receive an automatic confirmation email, if the submission was successful.
(5) You can “Edit submission” to your already submitted file, but ONLY before the deadline.

Appendix

Sender:

```
$ python3 RDTSend.py tcp.txt
Packet (seq num: 1) sent
Packet (seq num: 2) sent
Packet (seq num: 3) sent
Packet (seq num: 4) sent
Packet sent from Sender with seq_num 5 lost
Packet (seq num: 6) sent
Packet (seq num: 7) sent
Packet (seq num: 8) sent
Packet (seq num: 9) sent
Packet (seq num: 10) sent
Packet (seq num: 5) retransmitted
Packet (seq num: 7) retransmitted
Packet (seq num: 8) retransmitted
Packet (seq num: 9) retransmitted
Packet (seq num: 10) retransmitted
Packet (seq num: 6) retransmitted
Packet (seq num: 8) retransmitted
Packet (seq num: 11) sent
Packet (seq num: 9) retransmitted
Packet (seq num: 10) retransmitted
Packet (seq num: 11) retransmitted
Packet (seq num: 7) retransmitted
Packet (seq num: 12) sent
Packet (seq num: 9) retransmitted
Packet (seq num: 10) retransmitted
Packet (seq num: 11) retransmitted
Packet (seq num: 12) retransmitted
Packet (seq num: 8) retransmitted
Packet (seq num: 9) retransmitted
Packet (seq num: 13) sent
Packet (seq num: 11) retransmitted
Packet (seq num: 14) sent
Packet (seq num: 13) retransmitted
Packet (seq num: 14) retransmitted
Packet (seq num: 10) retransmitted
Packet (seq num: 11) retransmitted
Packet (seq num: 15) sent
Packet (seq num: 16) sent
Packet (seq num: 14) retransmitted
Packet (seq num: 15) retransmitted
Packet (seq num: 16) retransmitted
Packet (seq num: 12) retransmitted
Packet (seq num: 14) retransmitted
Packet (seq num: 17) sent
Packet (seq num: 15) retransmitted
Packet (seq num: 16) retransmitted
Packet (seq num: 17) retransmitted
Packet (seq num: 13) retransmitted
Packet (seq num: 14) retransmitted
Packet (seq num: 16) retransmitted
Packet (seq num: 17) retransmitted
Packet (seq num: 18) sent
Packet (seq num: 18) retransmitted
Packet sent from Sender with seq_num 19 lost
Packet (seq num: 15) retransmitted
Packet (seq num: 20) sent
Packet (seq num: 16) retransmitted
Packet (seq num: 21) sent
Packet (seq num: 19) retransmitted
Packet (seq num: 20) retransmitted
Packet (seq num: 21) retransmitted
Packet (seq num: 17) retransmitted
Packet (seq num: 19) retransmitted
Packet (seq num: 20) retransmitted
Packet (seq num: 21) retransmitted
Packet (seq num: 22) retransmitted
Packet (seq num: 22) sent
Packet (seq num: 18) retransmitted
```

```
Packet (seq num: 23) sent
Packet (seq num: 20) retransmitted
Packet (seq num: 21) retransmitted
Packet (seq num: 22) retransmitted
Packet (seq num: 23) retransmitted
Packet (seq num: 19) retransmitted
Packet (seq num: 20) retransmitted
Packet (seq num: 22) retransmitted
Packet (seq num: 23) retransmitted
Packet (seq num: 24) sent
Packet (seq num: 25) sent
Packet (seq num: 24) retransmitted
Packet (seq num: 21) retransmitted
Packet (seq num: 23) retransmitted
Packet (seq num: 26) sent
Packet (seq num: 24) retransmitted
Packet (seq num: 25) retransmitted
Packet (seq num: 26) retransmitted
Packet (seq num: 22) retransmitted
Packet (seq num: 23) retransmitted
Packet (seq num: 25) retransmitted
Packet (seq num: 27) sent
Packet (seq num: 26) retransmitted
Packet (seq num: 28) retransmitted
Packet (seq num: 28) sent
Packet (seq num: 24) retransmitted
Packet (seq num: 26) retransmitted
Packet (seq num: 27) retransmitted
Packet (seq num: 29) sent
Packet (seq num: 28) retransmitted
Packet (seq num: 29) retransmitted
Packet (seq num: 25) retransmitted
Packet (seq num: 30) sent
Packet (seq num: 27) retransmitted
Packet (seq num: 28) retransmitted
Packet (seq num: 29) retransmitted
Packet (seq num: 30) retransmitted
Packet (seq num: 26) retransmitted
Packet (seq num: 28) retransmitted
Packet (seq num: 29) retransmitted
Packet (seq num: 30) retransmitted
Packet (seq num: 31) retransmitted
Packet (seq num: 31) sent
Packet (seq num: 27) retransmitted
Packet (seq num: 29) retransmitted
Packet (seq num: 30) retransmitted
Packet (seq num: 32) sent
Packet (seq num: 31) retransmitted
Packet (seq num: 32) retransmitted
Packet (seq num: 28) retransmitted
Packet (seq num: 33) sent
Packet (seq num: 30) retransmitted
Packet (seq num: 31) retransmitted
Packet (seq num: 32) retransmitted
Packet (seq num: 33) retransmitted
Packet (seq num: 29) retransmitted
Packet (seq num: 30) retransmitted
Packet (seq num: 31) retransmitted
Packet (seq num: 34) retransmitted
Packet (seq num: 36) retransmitted
Packet (seq num: 34) sent
Packet (seq num: 35) sent
Packet (seq num: 36) sent
Packet (seq num: 32) retransmitted
Packet (seq num: 33) retransmitted
Packet (seq num: 37) sent
Packet (seq num: 35) retransmitted
Packet (seq num: 38) sent
Packet (seq num: 37) retransmitted
```

```
Packet (seq num: 38) retransmitted
Packet (seq num: 34) retransmitted
Packet (seq num: 39) sent
Packet (seq num: 36) retransmitted
Packet (seq num: 37) retransmitted
Packet (seq num: 38) retransmitted
Packet (seq num: 39) retransmitted
Packet (seq num: 35) retransmitted
Packet (seq num: 36) retransmitted
Packet (seq num: 37) retransmitted
Packet (seq num: 38) retransmitted
Packet (seq num: 39) retransmitted
Packet (seq num: 40) sent
Packet (seq num: 41) sent
Packet (seq num: 42) sent
```

Receiver:

```
python3 RDTReceive.py
Receive expected packet with seq num: 1
Cumulative ACK 1 sent to the Sender
Receive expected packet with seq num: 2
Cumulative ACK 2 sent to the Sender
Receive expected packet with seq num: 3
Cumulative ACK 3 sent to the Sender
Receive expected packet with seq num: 4
Cumulative ACK 4 sent to the Sender
Receive unexpected packet with wrong seq_num 6, resending latest ACK ...
ACK 4 sent from Receiver lost
Receive unexpected packet with wrong seq_num 7, resending latest ACK ...
Latest ACK 4 resent to the Sender
Receive unexpected packet with wrong seq_num 8, resending latest ACK ...
Latest ACK 4 resent to the Sender
Receive unexpected packet with wrong seq_num 9, resending latest ACK ...
Latest ACK 4 resent to the Sender
Receive expected packet with seq num: 5
Cumulative ACK 5 sent to the Sender
Receive unexpected packet with wrong seq_num 10, resending latest ACK ...
ACK 5 sent from Receiver lost
Receive unexpected packet with wrong seq_num 7, resending latest ACK ...
Latest ACK 5 resent to the Sender
Receive unexpected packet with wrong seq_num 8, resending latest ACK ...
Latest ACK 5 resent to the Sender
Receive unexpected packet with wrong seq_num 9, resending latest ACK ...
Latest ACK 5 resent to the Sender
Receive unexpected packet with wrong seq_num 10, resending latest ACK ...
Latest ACK 5 resent to the Sender
Receive expected packet with seq num: 6
Cumulative ACK 6 sent to the Sender
Receive unexpected packet with wrong seq_num 11, resending latest ACK ...
Latest ACK 6 resent to the Sender
Receive unexpected packet with wrong seq_num 8, resending latest ACK ...
Latest ACK 6 resent to the Sender
Receive unexpected packet with wrong seq_num 9, resending latest ACK ...
ACK 6 sent from Receiver lost
Receive unexpected packet with wrong seq_num 10, resending latest ACK ...
Latest ACK 6 resent to the Sender
Receive unexpected packet with wrong seq_num 11, resending latest ACK ...
Latest ACK 6 resent to the Sender
Receive expected packet with seq num: 7
Cumulative ACK 7 sent to the Sender
Receive unexpected packet with wrong seq_num 12, resending latest ACK ...
Latest ACK 7 resent to the Sender
Receive unexpected packet with wrong seq_num 9, resending latest ACK ...
Latest ACK 7 resent to the Sender
Receive unexpected packet with wrong seq_num 10, resending latest ACK ...
Latest ACK 7 resent to the Sender
Receive unexpected packet with wrong seq_num 11, resending latest ACK ...
Latest ACK 7 resent to the Sender
Receive unexpected packet with wrong seq_num 12, resending latest ACK ...
Latest ACK 7 resent to the Sender
Receive expected packet with seq num: 8
Cumulative ACK 8 sent to the Sender
Receive unexpected packet with wrong seq_num 13, resending latest ACK ...
ACK 8 sent from Receiver lost
Receive expected packet with seq num: 9
Cumulative ACK 9 sent to the Sender
Receive unexpected packet with wrong seq_num 11, resending latest ACK ...
Latest ACK 9 resent to the Sender
Receive unexpected packet with wrong seq_num 14, resending latest ACK ...
Latest ACK 9 resent to the Sender
Receive unexpected packet with wrong seq_num 13, resending latest ACK ...
Latest ACK 9 resent to the Sender
Receive unexpected packet with wrong seq_num 14, resending latest ACK ...
Latest ACK 9 resent to the Sender
Receive expected packet with seq num: 10
ACK 10 sent from Receiver lost
```

```
Receive expected packet with seq num: 11
Cumulative ACK 11 sent to the Sender
Receive unexpected packet with wrong seq_num 15, resending latest ACK ...
Latest ACK 11 resent to the Sender
Receive unexpected packet with wrong seq_num 16, resending latest ACK ...
Latest ACK 11 resent to the Sender
Receive unexpected packet with wrong seq_num 14, resending latest ACK ...
Latest ACK 11 resent to the Sender
Receive unexpected packet with wrong seq_num 15, resending latest ACK ...
Latest ACK 11 resent to the Sender
Receive unexpected packet with wrong seq_num 16, resending latest ACK ...
Latest ACK 11 resent to the Sender
Receive expected packet with seq num: 12
Cumulative ACK 12 sent to the Sender
Receive unexpected packet with wrong seq_num 14, resending latest ACK ...
Latest ACK 12 resent to the Sender
Receive unexpected packet with wrong seq_num 17, resending latest ACK ...
Latest ACK 12 resent to the Sender
Receive unexpected packet with wrong seq_num 15, resending latest ACK ...
Latest ACK 12 resent to the Sender
Receive unexpected packet with wrong seq_num 16, resending latest ACK ...
Latest ACK 12 resent to the Sender
Receive unexpected packet with wrong seq_num 17, resending latest ACK ...
Latest ACK 12 resent to the Sender
Receive expected packet with seq num: 13
Cumulative ACK 13 sent to the Sender
Receive expected packet with seq num: 14
Cumulative ACK 14 sent to the Sender
Receive unexpected packet with wrong seq_num 16, resending latest ACK ...
Latest ACK 14 resent to the Sender
Receive unexpected packet with wrong seq_num 17, resending latest ACK ...
Latest ACK 14 resent to the Sender
Receive unexpected packet with wrong seq_num 18, resending latest ACK ...
Latest ACK 14 resent to the Sender
Receive unexpected packet with wrong seq_num 18, resending latest ACK ...
Latest ACK 14 resent to the Sender
Receive expected packet with seq num: 15
Cumulative ACK 15 sent to the Sender
Receive expected packet with seq num: 16
Cumulative ACK 16 sent to the Sender
Receive unexpected packet with wrong seq_num 20, resending latest ACK ...
Latest ACK 16 resent to the Sender
Receive unexpected packet with wrong seq_num 21, resending latest ACK ...
Latest ACK 16 resent to the Sender
Receive unexpected packet with wrong seq_num 19, resending latest ACK ...
Latest ACK 16 resent to the Sender
Receive unexpected packet with wrong seq_num 20, resending latest ACK ...
Latest ACK 16 resent to the Sender
Receive unexpected packet with wrong seq_num 21, resending latest ACK ...
Latest ACK 16 resent to the Sender
Receive expected packet with seq num: 17
Cumulative ACK 17 sent to the Sender
Receive unexpected packet with wrong seq_num 19, resending latest ACK ...
Latest ACK 17 resent to the Sender
Receive unexpected packet with wrong seq_num 20, resending latest ACK ...
ACK 17 sent from Receiver lost
Receive unexpected packet with wrong seq_num 21, resending latest ACK ...
Latest ACK 17 resent to the Sender
Receive unexpected packet with wrong seq_num 22, resending latest ACK ...
Latest ACK 17 resent to the Sender
Receive unexpected packet with wrong seq_num 22, resending latest ACK ...
Latest ACK 17 resent to the Sender
Receive expected packet with seq num: 18
Cumulative ACK 18 sent to the Sender
```

```
Receive unexpected packet with wrong seq_num 23, resending latest ACK ...
Latest ACK 18 resent to the Sender
Receive unexpected packet with wrong seq_num 20, resending latest ACK ...
Latest ACK 18 resent to the Sender
Receive unexpected packet with wrong seq_num 21, resending latest ACK ...
Latest ACK 18 resent to the Sender
Receive unexpected packet with wrong seq_num 22, resending latest ACK ...
Latest ACK 18 resent to the Sender
Receive unexpected packet with wrong seq_num 23, resending latest ACK ...
ACK 18 sent from Receiver lost
Receive expected packet with seq num: 19
Cumulative ACK 19 sent to the Sender
Receive expected packet with seq num: 20
Cumulative ACK 20 sent to the Sender
Receive unexpected packet with wrong seq_num 22, resending latest ACK ...
ACK 20 sent from Receiver lost
Receive unexpected packet with wrong seq_num 23, resending latest ACK ...
Latest ACK 20 resent to the Sender
Receive unexpected packet with wrong seq_num 24, resending latest ACK ...
Latest ACK 20 resent to the Sender
Receive unexpected packet with wrong seq_num 24, resending latest ACK ...
Latest ACK 20 resent to the Sender
Receive unexpected packet with wrong seq_num 25, resending latest ACK ...
Latest ACK 20 resent to the Sender
Receive expected packet with seq num: 21
Cumulative ACK 21 sent to the Sender
Receive unexpected packet with wrong seq_num 26, resending latest ACK ...
Latest ACK 21 resent to the Sender
Receive unexpected packet with wrong seq_num 23, resending latest ACK ...
Latest ACK 21 resent to the Sender
Receive unexpected packet with wrong seq_num 24, resending latest ACK ...
ACK 21 sent from Receiver lost
Receive unexpected packet with wrong seq_num 25, resending latest ACK ...
Latest ACK 21 resent to the Sender
Receive unexpected packet with wrong seq_num 26, resending latest ACK ...
ACK 21 sent from Receiver lost
Receive expected packet with seq num: 22
Cumulative ACK 22 sent to the Sender
Receive expected packet with seq num: 23
ACK 23 sent from Receiver lost
Receive unexpected packet with wrong seq_num 25, resending latest ACK ...
Latest ACK 23 resent to the Sender
Receive unexpected packet with wrong seq_num 27, resending latest ACK ...
Latest ACK 23 resent to the Sender
Receive unexpected packet with wrong seq_num 26, resending latest ACK ...
Latest ACK 23 resent to the Sender
Receive unexpected packet with wrong seq_num 28, resending latest ACK ...
Latest ACK 23 resent to the Sender
Receive unexpected packet with wrong seq_num 28, resending latest ACK ...
Latest ACK 23 resent to the Sender
Receive expected packet with seq num: 24
Cumulative ACK 24 sent to the Sender
Receive unexpected packet with wrong seq_num 26, resending latest ACK ...
Latest ACK 24 resent to the Sender
Receive unexpected packet with wrong seq_num 27, resending latest ACK ...
Latest ACK 24 resent to the Sender
Receive unexpected packet with wrong seq_num 29, resending latest ACK ...
Latest ACK 24 resent to the Sender
Receive unexpected packet with wrong seq_num 28, resending latest ACK ...
Latest ACK 24 resent to the Sender
Receive unexpected packet with wrong seq_num 29, resending latest ACK ...
Latest ACK 24 resent to the Sender
Receive expected packet with seq num: 25
Cumulative ACK 25 sent to the Sender
Receive unexpected packet with wrong seq_num 30, resending latest ACK ...
ACK 25 sent from Receiver lost
Receive unexpected packet with wrong seq_num 27, resending latest ACK ...
```

```
Latest ACK 25 resent to the Sender
Receive unexpected packet with wrong seq_num 28, resending latest ACK ...
Latest ACK 25 resent to the Sender
Receive unexpected packet with wrong seq_num 29, resending latest ACK ...
ACK 25 sent from Receiver lost
Receive unexpected packet with wrong seq_num 30, resending latest ACK ...
Latest ACK 25 resent to the Sender
Receive expected packet with seq num: 26
Cumulative ACK 26 sent to the Sender
Receive unexpected packet with wrong seq_num 28, resending latest ACK ...
Latest ACK 26 resent to the Sender
Receive unexpected packet with wrong seq_num 29, resending latest ACK ...
Latest ACK 26 resent to the Sender
Receive unexpected packet with wrong seq_num 30, resending latest ACK ...
Latest ACK 26 resent to the Sender
Receive unexpected packet with wrong seq_num 31, resending latest ACK ...
Latest ACK 26 resent to the Sender
Receive unexpected packet with wrong seq_num 31, resending latest ACK ...
Latest ACK 26 resent to the Sender
Receive expected packet with seq num: 27
Cumulative ACK 27 sent to the Sender
Receive unexpected packet with wrong seq_num 29, resending latest ACK ...
Latest ACK 27 resent to the Sender
Receive unexpected packet with wrong seq_num 30, resending latest ACK ...
Latest ACK 27 resent to the Sender
Receive unexpected packet with wrong seq_num 32, resending latest ACK ...
Latest ACK 27 resent to the Sender
Receive unexpected packet with wrong seq_num 31, resending latest ACK ...
Latest ACK 27 resent to the Sender
Receive unexpected packet with wrong seq_num 32, resending latest ACK ...
Latest ACK 27 resent to the Sender
Receive expected packet with seq num: 28
Cumulative ACK 28 sent to the Sender
Receive unexpected packet with wrong seq_num 33, resending latest ACK ...
Latest ACK 28 resent to the Sender
Receive unexpected packet with wrong seq_num 30, resending latest ACK ...
Latest ACK 28 resent to the Sender
Receive unexpected packet with wrong seq_num 31, resending latest ACK ...
Latest ACK 28 resent to the Sender
Receive unexpected packet with wrong seq_num 32, resending latest ACK ...
Latest ACK 28 resent to the Sender
Receive unexpected packet with wrong seq_num 33, resending latest ACK ...
Latest ACK 28 resent to the Sender
Receive expected packet with seq num: 29
ACK 29 sent from Receiver lost
Receive expected packet with seq num: 30
Cumulative ACK 30 sent to the Sender
Receive expected packet with seq num: 31
Cumulative ACK 31 sent to the Sender
Receive unexpected packet with wrong seq_num 34, resending latest ACK ...
Latest ACK 31 resent to the Sender
Receive unexpected packet with wrong seq_num 36, resending latest ACK ...
Latest ACK 31 resent to the Sender
Receive unexpected packet with wrong seq_num 34, resending latest ACK ...
Latest ACK 31 resent to the Sender
Receive unexpected packet with wrong seq_num 35, resending latest ACK ...
Latest ACK 31 resent to the Sender
Receive unexpected packet with wrong seq_num 36, resending latest ACK ...
Latest ACK 31 resent to the Sender
Receive expected packet with seq num: 32
Cumulative ACK 32 sent to the Sender
Receive expected packet with seq num: 33
Cumulative ACK 33 sent to the Sender
Receive unexpected packet with wrong seq_num 37, resending latest ACK ...
Latest ACK 33 resent to the Sender
Receive unexpected packet with wrong seq_num 35, resending latest ACK ...
Latest ACK 33 resent to the Sender
Receive unexpected packet with wrong seq_num 38, resending latest ACK ...
```

```
Latest ACK 33 resent to the Sender
Receive expected packet with seq num: 34
Cumulative ACK 34 sent to the Sender
Receive unexpected packet with wrong seq_num 39, resending latest ACK ...
Latest ACK 34 resent to the Sender
Receive unexpected packet with wrong seq_num 36, resending latest ACK ...
Latest ACK 34 resent to the Sender
Receive unexpected packet with wrong seq_num 37, resending latest ACK ...
Latest ACK 34 resent to the Sender
Receive unexpected packet with wrong seq_num 38, resending latest ACK ...
Latest ACK 34 resent to the Sender
Receive unexpected packet with wrong seq_num 39, resending latest ACK ...
Latest ACK 34 resent to the Sender
Receive expected packet with seq num: 35
Cumulative ACK 35 sent to the Sender
Receive expected packet with seq num: 36
Cumulative ACK 36 sent to the Sender
Receive expected packet with seq num: 37
Cumulative ACK 37 sent to the Sender
Receive expected packet with seq num: 38
Cumulative ACK 38 sent to the Sender
Receive expected packet with seq num: 39
Cumulative ACK 39 sent to the Sender
Receive expected packet with seq num: 40
Cumulative ACK 40 sent to the Sender
Receive expected packet with seq num: 41
Cumulative ACK 41 sent to the Sender
Receive expected packet with seq num: 42
Cumulative ACK 42 sent to the Sender
Socket timeout, terminate the receiver program.
```