PA3 Report

Design

Utilities

create_packet: Create a packet with the assigned sequence number and message.

send_packet: Send a packet to the layer 3.

create_ack_packet: Create an acknowledgment packet with the provided sequence number.

send_ack_packet: Send the acknowledgment packet to the layer 3.

EntityA

Entity A is responsible for:

- Initiating data transmissions to EntityB.
- Keeping track of unacknowledged packets using a sliding window mechanism.
- Handling timeouts and retransmissions if acknowledgments from EntityB are not received in time.

output: If the window is not full, send a packet out.

Input: processes incoming packets. If a packet is valid, it prepares for sending the next packet.

timer_interrupt: Handles timeout events.

EntityB

Entity B is responsible for:

- Receiving packets from EntityA.
- Sending acknowledgments for correctly received packets.
- Ensuring packets are delivered to the upper layer in the correct sequence.

Input: Checks if an incoming packet is either corrupted or not in the expected sequence. Send an acknowledgment packet base on if the incoming packet is expected back to A for either recovering the wrong packet or moving on to the next packet.

Testing

Testing focused on verifying the correctness of the protocol under various conditions, including:

- Normal Conditions: No packet loss or corruption.

- High Loss Conditions: Up to 10% packet loss.
- High Corruption Conditions: Up to 10% packet corruption.
- Mixed Conditions: Combination of packet loss and corruption.

Known Issues

While the program handles basic packet corruption by recalculating and comparing checksums, the efficiency of the recovery mechanisms might not be sufficient, especially in scenarios where not many packets are corrupted.

Output

> **This section of your report must include output for a single run of the simulator, up to the point when at least 20 messages were successfully transferred from sender to receiver (i.e., the sender received ACKs for these messages), a loss probability of 0.05, a corruption probability of 0.05, and a trace level of 2. Annotate your output to show how your protocol correctly recovered from packet loss and corruption.**

% python rdtsim.py -n 20 -l 0.05 -c 0.05 -v 2

SIMULATION CONFIGURATION

(-n) # layer5 msgs to be provided: 20

(-d) avg layer5 msg interarrival time: 10.0

(-z) transport protocol seqnum limit: 16

(-l) layer3 packet loss prob: 0.05

(-c) layer3 packet corruption prob: 0.05

(-s) simulation random seed: 1713746406087082000

===== SIMULATION BEGINS

EntityA: sending packet Pkt(seqnum=0, acknum=0, checksum=1940, payload=b'aaaaaaaaaaaaaaaaaaaa')

EntityB: received packet Pkt(seqnum=0, acknum=0, checksum=1940, payload=b'aaaaaaaaaaaaaaaaaaaa') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=0, checksum=863, payload=b'ack 0
')

EntityA: received ack 0

EntityA: sending packet Pkt(seqnum=1, acknum=0, checksum=1961, payload=b'bbbbbbbbbbbbbbbbbbb')

EntityB: received packet Pkt(seqnum=1, acknum=0, checksum=1961, payload=b'bbbbbbbbbbbbbbbbbbb') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=1, checksum=865, payload=b'ack 1
')

EntityB received packet 'bbbbbb' but the acknowledge packet send to A acknowledging EntityB received packet 'bbbbbb' is lost. EntityA do not know Enitiy B has received the packet 'bbbbbb'.

TO_LAYER3: packet being lost

EntityA: sending packet Pkt(seqnum=2, acknum=0, checksum=1982, payload=b'ccccccccccccccccccc')

EntityB: received packet Pkt(seqnum=2, acknum=0, checksum=1982, payload=b'ccccccccccccccccccc') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2
')

EntityA: received ack 2

EntityA: sending packet Pkt(seqnum=3, acknum=0, checksum=2003, payload=b'ddddddddddddddddddddd')

The packet 'dddddd' sent to EntityB is corrupted.

TO_LAYER3: packet being corrupted

EntityB: received packet Pkt(seqnum=3, acknum=1, checksum=2003, payload=b'ddddddddddddddddddddd') is corrupted

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2
')

EntityA: sending packet Pkt(seqnum=4, acknum=0, checksum=2024, payload=b'eeeeeeeeeeeeeeeeeee')

EntityB: received packet Pkt(seqnum=4, acknum=0, checksum=2024, payload=b'eeeeeeeeeeeeeeeeeee') is not the expected seq

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

EntityA: sending packet Pkt(seqnum=5, acknum=0, checksum=2045, payload=b'fffffffffffffffffff')

EntityB: received packet Pkt(seqnum=5, acknum=0, checksum=2045, payload=b'fffffffffffffffffff') is not the expected seq

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

EntityA: sending packet Pkt(seqnum=6, acknum=0, checksum=2066, payload=b'gggggggggggggggggggg')

The packet ‘gggggg’ sent to EntityB is corrupted.

TO_LAYER3: packet being corrupted

EntityB: received packet Pkt(seqnum=6, acknum=0, checksum=2066, payload=b'Zgggggggggggggggggggg') is corrupted

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

EntityA: sending packet Pkt(seqnum=7, acknum=0, checksum=2087, payload=b'hhhhhhhhhhhhhhhhhhhh')

EntityB: received packet Pkt(seqnum=7, acknum=0, checksum=2087, payload=b'hhhhhhhhhhhhhhhhhhhh') is not the expected seq

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

EntityA: sending packet Pkt(seqnum=8, acknum=0, checksum=2108, payload=b'iiiiiiiiiiiiiiiiii')

EntityB: received packet Pkt(seqnum=8, acknum=0, checksum=2108, payload=b'iiiiiiiiiiiiiiiiii') is not the expected seq

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

EntityA: sending packet Pkt(seqnum=9, acknum=0, checksum=2129, payload=b'jjjjjjjjjjjjjjjjjj')

The packet ‘jjjjjj sent to EntityB is corrupted.

TO_LAYER3: packet being lost

EntityA: sending packet Pkt(seqnum=10, acknum=0, checksum=2150, payload=b'kkkkkkkkkkkkkkkkkkkk')

EntityB: received packet Pkt(seqnum=10, acknum=0, checksum=2150, payload=b'kkkkkkkkkkkkkkkkkkkk') is not the expected seq

EntityB: sending ack Pkt(seqnum=0, acknum=2, checksum=867, payload=b'ack 2 ')

The timer at Entity A is timed out which means there is corrupted packets. Entity A resend all the packets store in its window.

EntityA: timer event

EntityA: resending packet Pkt(seqnum=3, acknum=0, checksum=2003, payload=b'ddddddddddddddddddddd')

EntityA: resending packet Pkt(seqnum=4, acknum=0, checksum=2024, payload=b'eeeeeeeeeeeeeeeeeeee')

EntityA: resending packet Pkt(seqnum=5, acknum=0, checksum=2045, payload=b'ffffffffffffffffffff')

EntityA: resending packet Pkt(seqnum=6, acknum=0, checksum=2066, payload=b'gggggggggggggggggggg')

EntityA: resending packet Pkt(seqnum=7, acknum=0, checksum=2087, payload=b'hhhhhhhhhhhhhhhhhhhh')

EntityA: resending packet Pkt(seqnum=8, acknum=0, checksum=2108, payload=b'iiiiiiiiiiiiiiiiii')

EntityA: resending packet Pkt(seqnum=9, acknum=0, checksum=2129, payload=b'jjjjjjjjjjjjjjjjjj')

EntityA: resending packet Pkt(seqnum=10, acknum=0, checksum=2150, payload=b'kkkkkkkkkkkkkkkkkkkk')

EntityB: received packet Pkt(seqnum=3, acknum=0, checksum=2003, payload=b'ddddddddddddddddddddd') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=3, checksum=869, payload=b'ack 3
')

EntityA: received ack 3

After EntityA receive the acknowledgement that EnityB received packet “dddddd”. So “dddddd” is removed from its current window which cause a space in the window. As there is a space in the window, it can continue to send out packets.

EntityA: sending packet Pkt(seqnum=11, acknum=0, checksum=2171, payload=b'lllllllllllllllllll')

EntityB: received packet Pkt(seqnum=4, acknum=0, checksum=2024, payload=b'eeeeeeeeeeeeeeeeeee') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=4, checksum=871, payload=b'ack 4
')

EntityB: received packet Pkt(seqnum=5, acknum=0, checksum=2045, payload=b'fffffffffffffffffff') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=5, checksum=873, payload=b'ack 5
')

EntityA: received ack 4

EntityA: sending packet Pkt(seqnum=12, acknum=0, checksum=2192, payload=b'mmmmmmmmmmmmmmmmmmmm')

EntityB: received packet Pkt(seqnum=6, acknum=0, checksum=2066, payload=b'gggggggggggggggggggg') is correct

EntityB: sending ack Pkt(seqnum=0, acknum=6, checksum=875, payload=b'ack 6
')

EntityA: received ack 5

EntityA: sending packet Pkt(seqnum=13, acknum=0, checksum=2213, payload=b'nnnnnnnnnnnnnnnnnnn')

===== SIMULATION ENDS

SIMULATION SUMMARY

---

layer5 msgs provided to A: 20

elapsed time units: 178.8655215350471

layer3 packets sent by A: 22

layer3 packets sent by B: 14

layer3 packets lost: 2

layer3 packets corrupted: 2

layer5 msgs delivered by A: 0

layer5 msgs delivered by B: 7

layer5 msgs by B/elapsed time: 0.039135546861827214

This section of your report must also show the statistics for your program under various loss and corruption probabilities, for runs of the simulator in which at least 1,000 layer5 messages are simulated at the sender. (Short runs lead to wildly varying statistics.)

	Loss 1%, Corruption 1%	Loss 10%, Corruption 1%	Loss 1%, Corruption 10%	Loss 10%, Corruption 10%
layer5 msgs provided to A:	1000	1000	1000	1000
elapsed time units:	10051.8181230254	9855.298929782	9870.305076508	10122.246496072
layer3 packets sent by A:	1206	1445	1351	1388
layer3 packets sent by B:	1192	1302	1333	1247
layer3 packets lost:	25	252	25	285
layer3 packets corrupted:	25	31	244	235
layer5 msgs delivered by A:	0	0	0	0
layer5 msgs delivered by B:	999	696	697	434
layer5 msgs by B/elapsed time:	0.0993850055555	0.0706219065458	0.0706158517489	0.042875857663

Loss 1%, Corruption 1%

The protocol successfully delivered almost all messages (999 out of 1000) with a fast delivery rate at 0.0993850055555.

The simulator appears to be functioning correctly and the protocol is working well.

Loss 10%, Corruption 1%
The number of messages successfully delivered (697out of 1000) drops due to high loss rate.
The number of packets A sent increase due to packet loss.
Due to high loss rate, B need to ask A to retransmit lost packets before message successfully delivered, which takes more time.
The simulator and protocol work as expected.

Loss 1%, Corruption 10%
The number of messages successfully delivered (696 out of 1000) drops due to high corruption rate.
The number of packets A sent increase due to packet corruption.
Due to high corruption rate, B need to ask A to retransmit corrupted packets before message successfully delivered, which takes more time.
The simulator and protocol work as expected.

Loss 10%, Corruption 10%
The number of messages successfully delivered (434out of 1000) drops due to high loss and corruption rate.
The number of packets A sent increase due to packet loss and corruption.
Due to high loss and corruption rate, B need to ask A to retransmit loss and corrupted packets before message successfully delivered, which takes more time.
The simulator and protocol work as expected.