See below a print-out from 10 messages being simulated, commentary is in red.

Enter the number of messages to simulate: 10

Enter packet loss probability [enter 0.0 for no loss]:0.1

Enter packet corruption probability [0.0 for no corruption]:0.15

Enter average time between messages from sender's layer5 [ > 0.0]:1000

Enter TRACE:2

EVENT time: 46.784164, type: 1, fromlayer5 entity: 0

Packet 1 being written

EVENT time: 50.030354, type: 2, fromlayer3 entity: 1

Packet 1 received by Host B

EVENT time: 51.779907, type: 2, fromlayer3 entity: 0

Packet 1 ACK received

Successful packet transfer with ACK received.

EVENT time: 803.845276, type: 1, fromlayer5 entity: 0

Packet 2 being written

EVENT time: 805.045898, type: 2, fromlayer3 entity: 1

Packet 2 received by Host B

EVENT time: 808.264221, type: 2, fromlayer3 entity: 0

Packet 2 ACK received

Successful packet transfer with ACK received.

EVENT time: 1160.997925, type: 1, fromlayer5 entity: 0

Packet 3 being written

EVENT time: 1163.245972, type: 2, fromlayer3 entity: 1

Packet 3 received by Host B

EVENT time: 1165.188477, type: 2, fromlayer3 entity: 0

Packet 3 ACK received

Successful packet transfer with ACK received.

EVENT time: 1666.376709, type: 1, fromlayer5 entity: 0

Packet 4 being written

TOLAYER3: packet being corrupted

First instance of packet being corrupted, given by TOLAYER3 message.

EVENT time: 1669.164917, type: 2, fromlayer3 entity: 1

Packet 4 corrupted at Host B, responding with NACK

Checksum successfully caught error, respond with NACK.

EVENT time: 1671.603027, type: 2, fromlayer3 entity: 0

Packet 4 NACK received, retransmitting

NACK received, retransmitting packet.

EVENT time: 1674.883667, type: 2, fromlayer3 entity: 1

Packet 4 received by Host B

TOLAYER3: packet being lost

First instance of packet being lost as given by TOLAYER3 message.

EVENT time: 1711.603027, type: 0, timerinterrupt entity: 0

Packet 4 timed-out, retransmitting

Time-out successful in catching error, retransmitting packet.

EVENT time: 1715.076782, type: 2, fromlayer3 entity: 1

Packet 4 received by Host B

EVENT time: 1716.675659, type: 2, fromlayer3 entity: 0

Packet 4 ACK received

Successfully delivered packet 4 and ACK received.

EVENT time: 2528.908203, type: 1, fromlayer5 entity: 0

Packet 5 being written

TOLAYER3: packet being corrupted

Packet corruption, expecting NACK.

EVENT time: 2529.427002, type: 1, fromlayer5 entity: 0

Packet 6 dropped in transfer to layer 3

Between transmission, next packet is sent from application layer. This packet was dropped.

EVENT time: 2531.697510, type: 2, fromlayer3 entity: 1

Packet 5 corrupted at Host B, responding with NACK

NACK expected -- checksum successful again.

EVENT time: 2532.961182, type: 2, fromlayer3 entity: 0

Packet 5 NACK received, retransmitting

Retransmit after NACK receipt.

EVENT time: 2534.421143, type: 2, fromlayer3 entity: 1

Packet 5 received by Host B

TOLAYER3: packet being corrupted

Packet corruption, expecting NACK.

EVENT time: 2539.738525, type: 2, fromlayer3 entity: 0

Packet 5 NACK received, retransmitting

NACK expected, checksum successful.

EVENT time: 2543.279053, type: 2, fromlayer3 entity: 1

Packet 5 received by Host B

EVENT time: 2546.054932, type: 2, fromlayer3 entity: 0

Packet 5 ACK received

Successfully delivered packet 5 and ACK received.

EVENT time: 3278.583984, type: 1, fromlayer5 entity: 0

Packet 6 being written

TOLAYER3: packet being corrupted

Packet corrupted, expecting NACK.

EVENT time: 3282.939697, type: 2, fromlayer3 entity: 1

Packet 6 corrupted at Host B, responding with NACK

NACK expected, checksum successful.

EVENT time: 3286.287598, type: 2, fromlayer3 entity: 0

Packet 6 NACK received, retransmitting

TOLAYER3: packet being corrupted

NACK received, retransmitting and packet corrupted again.

EVENT time: 3290.714355, type: 2, fromlayer3 entity: 1

Packet 6 corrupted at Host B, responding with NACK

TOLAYER3: packet being lost

Packet being lost, expect time-out.

EVENT time: 3326.287598, type: 0, timerinterrupt entity: 0

Packet 6 timed-out, retransmitting

TOLAYER3: packet being corrupted

Time-out expected, time-out feature successful and retransmitting.

EVENT time: 3331.549316, type: 2, fromlayer3 entity: 1

Packet 6 corrupted at Host B, responding with NACK

TOLAYER3: packet being lost

Packet being lost, expect time-out.

EVENT time: 3366.287598, type: 0, timerinterrupt entity: 0

Packet 6 timed-out, retransmitting

TOLAYER3: packet being lost

Time-out expected, time-out feature successful and retransmitting; packet lost again, expect another time-out.

EVENT time: 3406.287598, type: 0, timerinterrupt entity: 0

Packet 6 timed-out, retransmitting

Time-out expected, time-out feature successful and retransmitting.

EVENT time: 3407.367920, type: 2, fromlayer3 entity: 1

Packet 6 received by Host B

TOLAYER3: packet being corrupted

Packet corrupted, expect NACK.

EVENT time: 3409.966064, type: 2, fromlayer3 entity: 0

Packet 6 NACK received, retransmitting

TOLAYER3: packet being lost

NACK received, another time-out expected.

EVENT time: 3426.993164, type: 1, fromlayer5 entity: 0

Packet 8 dropped in transfer to layer 3

Next packet tried to enter from layer 5, due to link being busy it is dropped.

EVENT time: 3449.966064, type: 0, timerinterrupt entity: 0

Packet 6 timed-out, retransmitting

TOLAYER3: packet being lost

Time-out was expected, time-out feature successful and retransmitting; another time-out expected.

EVENT time: 3489.966064, type: 0, timerinterrupt entity: 0

Packet 6 timed-out, retransmitting

Time-out was expected, retransmit.

EVENT time: 3491.742188, type: 2, fromlayer3 entity: 1

Packet 6 received by Host B

Correctly received, expect ACK.

EVENT time: 3496.598145, type: 2, fromlayer3 entity: 0

Packet 6 ACK received

ACK expected, packet was finally delivered after many bumps.

EVENT time: 4404.577637, type: 1, fromlayer5 entity: 0

Packet 7 being written

TOLAYER3: packet being lost

Packet being lost, expect time-out.

EVENT time: 4444.577637, type: 0, timerinterrupt entity: 0

Packet 7 timed-out, retransmitting

Time-out was expected, retransmit.

EVENT time: 4446.010254, type: 2, fromlayer3 entity: 1

Packet 7 received by Host B

EVENT time: 4450.613770, type: 2, fromlayer3 entity: 0

Packet 7 ACK received

Successfully delivered packet 7 and ACK received.

EVENT time: 5115.098633, type: 1, fromlayer5 entity: 0

Packet 8 being written

Since two packets were lost in entry, this is expected to be the last packet entering transmission.

EVENT time: 5117.860840, type: 2, fromlayer3 entity: 1

Simulator terminated at time 5117.860840

after sending 10 msgs from layer5

Code below, [Github link](https://github.com/LucasDionisopoulos/Lab_3) to the full code and compiled file.

*/\*\*\*\*\*\*\*\*\* STUDENTS WRITE THE NEXT SEVEN ROUTINES \*\*\*\*\*\*\*\*\*/*

*bool* pktintransit;              */\* Boolean to test whether link is open    \*/*

struct pkt pktsend;             */\* Packet structure to be sent             \*/*

struct pkt response;            */\* NACK or ACK to be sent back             \*/*

*int* globalseqnum;               */\* Global var to track seqnum              \*/*

*int* totalpackets;               */\* Global var to track packets sent        \*/*

*/\* Simple wraparound function for summing two 8 bit integers \*/*

*int* wraparound(wrapsum) {

*int* carry **=** (wrapsum **>** 255) **?** 1 **:** 0;

**return** ((wrapsum **%** 256) **+** carry);

}

*/\* helper function that returns a checksum for a provided packet \*/*

*int* calc\_checksum(packet)

  struct pkt packet;

{

*int* sum **=** 0;

  sum **+=** packet.*seqnum* **+** packet.*acknum*;

  sum **=** wraparound(sum);

**for** (*int* i**=**0; i**<**20; i**++**) {

    sum **+=** packet.*payload*[i];

    sum **=** wraparound(sum);

  }

  sum **=** **~**sum;     */\* convert to one's compliment \*/*

**return** sum;

}

*/\* helper function that tests whether checksum passes \*/*

*int* test\_checksum(packet)

  struct pkt packet;

{

*int* sum **=** 0;

  sum **+=** packet.*seqnum* **+** packet.*acknum*;

  sum **=** wraparound(sum);

  sum **+=** packet.*checksum*;

  sum **=** wraparound(sum);

**for** (*int* i**=**0; i**<**20; i**++**) {

    sum **+=** packet.*payload*[i];

    sum **=** wraparound(sum);

  }

**if** (sum **==** 254) {  */\* Checksum passes, would return -1 but set-up so it should return 255 - 1; return true \*/*

**return** true;

  } **else** {

**return** false;

  }

}

*/\* called from layer 5, passed the data to be sent to other side \*/*

A\_output(message)

  struct msg message;

{

  totalpackets **++**;

**if** (pktintransit) {      */\* packet is outstanding, drop message \*/*

    printf("Packet *%d* dropped in transfer to layer 3 \n", totalpackets);

  }

**else** {                   */\* link is open, prepare to send packet \*/*

    printf("Packet *%d* being written \n", globalseqnum);

**for** (*int* i**=**0; i**<**20; i**++**) {     */\* fill sending packet with message data \*/*

      pktsend.*payload*[i] **=** message.*data*[i];

    }

    pktsend.*seqnum* **=** globalseqnum;   */\* increment seqnum \*/*

    pktsend.*checksum* **=** calc\_checksum(pktsend);

    pktintransit **=** true;

    tolayer3(0, pktsend);

    starttimer(0, 40.0);    */\* Start time to watch for packet loss, timeout at 4x RTT \*/*

  }

}

B\_output(message)  */\* need be completed only for extra credit \*/*

  struct msg message;

{

}

*/\* called from layer 3, when a packet arrives for layer 4 \*/*

A\_input(packet)

  struct pkt packet;

{

**if** (packet.*acknum* **==** 0 **||** **!**test\_checksum(packet)) {    */\* NACK received or response is corrupted --> retransmit message \*/*

    stoptimer(0); */\* Stop timer, will restart if retransmitting or on next packet \*/*

    printf("Packet *%d* NACK received, retransmitting \n", packet.*seqnum*);

**for** (*int* i**=**0; i**<**20; i**++**) {                   */\* Rewrite the packet with the right data \*/*

      packet.*payload*[i] **=** pktsend.*payload*[i];

    }

    packet.*checksum* **=** calc\_checksum(packet);

    tolayer3(0, pktsend);

    starttimer(0, 40.0); */\* Start time to watch for packet loss, timeout at 4x RTT \*/*

  }

**else** {       */\* ACK received, increment seqnum and enable new messages \*/*

    stoptimer(0); */\* Stop timer, will restart if retransmitting or on next packet \*/*

    printf("Packet *%d* ACK received \n", packet.*seqnum*);

    globalseqnum **++**;

    pktintransit **=** false;

  }

}

*/\* called when A's timer goes off, retransmit packet \*/*

A\_timerinterrupt()

{

  printf("Packet *%d* timed-out, retransmitting \n", globalseqnum);

  tolayer3(0, pktsend);

  starttimer(0, 40.0);   */\* Start time to watch for packet loss, timeout at 4x RTT \*/*

}

*/\* the following routine will be called once (only) before any other \*/*

*/\* entity A routines are called. You can use it to do any initialization \*/*

A\_init()

{

  pktintransit **=** false;

  globalseqnum **=** 1;

  totalpackets **=** 0;

}

*/\* Note that with simplex transfer from a-to-B, there is no B\_output() \*/*

*/\* called from layer 3, when a packet arrives for layer 4 at B\*/*

B\_input(packet)

  struct pkt packet;

{

**for**(*int* i**=**0; i**<**20; i**++**) {     */\* Copy over the payload to ACK for passing back                 \*/*

      response.*payload*[i] **=** packet.*payload*[i];

    }

  response.*seqnum* **=** packet.*seqnum*;   */\* Set ack seqnum equal to packet seqnum \*/*

**if** (**!**test\_checksum(packet)) {      */\* Checksum does not pass, respond with NACK \*/*

    printf("Packet *%d* corrupted at Host B, responding with NACK \n", packet.*seqnum*);

    response.*acknum* **=** 0;    */\* Assign response an acknum of 0 to denote that it is a NACK \*/*

    response.*checksum* **=** calc\_checksum(response);

    tolayer3(1, response);  */\* Respond with NACK \*/*

  }

**else** {  */\* On correct receipt, respond with ACK and send to layer 5 \*/*

    printf("Packet *%d* received by Host B \n", packet.*seqnum*);

    response.*acknum* **=** 1;    */\* Assign response an acknum of 1 to denote that it is an ACK, not a NACK \*/*

    packet.*acknum* **=** 1;      */\* Assign acknum 1 so application layer knows it was correctly received \*/*

    response.*checksum* **=** calc\_checksum(response);

    tolayer5(1,packet.*payload*);

    tolayer3(1, response);

  }

}

*/\* called when B's timer goes off \*/*

B\_timerinterrupt()

{

}

*/\* the following rouytine will be called once (only) before any other \*/*

*/\* entity B routines are called. You can use it to do any initialization \*/*

B\_init()

{

}