CSE123: Computer Networks

Winter 2021, Discussion - 2

P1 TA: Palash Chauhan
Overview

1. Implementing communication between two or more hosts
2. Hosts implemented as Threads
3. Network link is simulated
4. 2 types of hosts → Senders and Receivers
5. Sender hosts must transmit messages typed in at the command line to a corresponding receiver host.
6. Messages can be dropped
7. **Messages can be corrupted!**
8. Improve performance using SWP.
CRC

- **Sender**
  - When constructing the frame (header+payload), set CRC field to 0.
  - Calculate the CRC for the frame and set the CRC field.
  - Convert frame to char array.
  - Append to outgoing frames list.

- **Receiver**
  - Check for corruption.
  - If corrupted, drop. Otherwise, send ACK.
**SWP: Sequence Number Wrap Around**

- You should NOT use more than 8 bits (unsigned char) for seq/ack numbers.
- You need to handle sequence number wrap around once the value reaches 255. Your seq/ack number should wrap back to 0.
- How to do this?
- Answer: % modulus
Sliding Window Protocol

• struct Sender_t
  • SWS – Sliding window size
  • LAR (Last Acknowledgement Received) - Sequence number of last acknowledgement received, defines lower bound of the sender window
  • LFS (Last Frame Sent)- Sequence number of the last frame sent, defines upper bound of the window
  • Window is from [LAR+1, LFS], that is all frames that have been sent but not yet Acked.
Frame Sequence Number in Sender

CASE 1: Usual Case
\[ \text{LAR} \leq \text{LFS} \]
\[ \text{LAR} \leq \text{LFS} \land \text{seqNo} > \text{LAR} \land \text{seqNo} \leq \text{LFS} \]

CASE 2: Sequence Number Wrap Around
\[ \text{LAR} > \text{LFS} \]
\[ \text{LAR} > \text{LFS} \land (\text{seqNo} > \text{LAR} \lor \text{seqNo} \leq \text{LFS}) \]

In this case, we are not using the full window of 4.

Sender with SWS = 4, sequence number in [0,7]
Sliding Window Protocol

- `struct Receiver_t`
  - `RWS` - Max receiver window size
  - `NFE` - Next Frame Expected
  - `LFR` - Sequence number of largest consecutive frame received
  - `LAF` - Sequence number of largest acceptable frame
  - `LFR = NFE - 1`
  - `LAF = NFE + RWS - 1`
Frame Sequence Number in Receiver

CASE 1: Usual Case
NFE + RWS - 1 >= NFE
NFE + RWS - 1 >= NFE && seqNo >= NFE && seqNo <= NFE + RWS - 1

Remember NFE is just LFR + 1 and LAF is just NFE + RWS - 1.

Green sequence numbers are in window and grey are outside.

CASE 2: Sequence Number Wrap Around
NFE + RWS - 1 < NFE
NFE + RWS - 1 < NFE && (seqNo >= NFE || seqNo <= NFE + RWS - 1)

Receiver with RWS = 4, sequence number in [0,7]
Sender Buffer/Window

• Sender need to maintain window(buffer) while sending packets out
• The window is like this:
  • struct sendQ_slot {
      struct timeval* timeout;  // event associate with send timeout
      Frame frame;
  } sendQ[SWS];

• Timeout is of type struct timeval (declared in sys/time.h)
• Index in to the sender buffer using (sequence number % SWS)
Receiver Buffer/Window

• Similarly, it is better for receiver to maintain a window too.
• Example:
  ```c
  struct recvQ_slot {
    struct Frame_t* frame
  } recvQ[RWS]
  ```

• Why don’t we need a timeout here?

  • Index in to the receiver buffer using (sequence number % RWS)