

# CSE 123b Communications Software

Spring 2002

## Lecture 3: Reliable Communications

Stefan Savage

Some slides courtesy David Wetherall

## Administrativa

- Home page is up and working
  - <http://www-cse.ucsd.edu/classes/sp02/cse123B/>
  - Class notes included
  - Sign up for the class mailing list (instructions on the Web page)
- First homework will be assigned next Tuesday
- Class on Thursday is \*\*\***cancelled**\*\*\*

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

2

## Last Time

- We talked about the network layer (IP) and internetworking.
- We assume: the network provides best-effort (i.e. unreliable) delivery of packets from one host to another
  - How that is done, **routing**, is left for a future class

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

3

## Today's class

- We begin on the transport layer
  - Builds on the services of the Network layer
  - Communication between processes running on hosts
- Principle focus
  - How do we ensure that a message is **reliably** communicated from one host to another?
- **Topics**
  - ARQ
  - Sliding windows
  - Retransmission timers

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

4

## Thought experiment

- You want to send a long letter to your friend
- All you have (and all your friend has) is postcards
- Postcards get lost in the mail, delayed, damaged
- How do you send the letter?

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

5

## Reliable Transmission

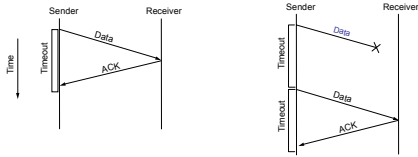
- How do we reliably send a message when packets can be lost in the network?
- Some options
  - Detect a loss and retransmit (**Cerf&Kahn74**, and others)
  - Send redundantly (Byers et al.98, and others)

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

6

## Automatic Repeat Request (ARQ)



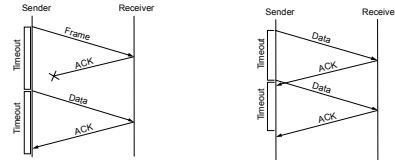
- Packets can be corrupted or lost. How do we add reliability?
- Acknowledgments (ACKs) and retransmissions after a timeout
- ARQ is generic name for protocols based on this strategy

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

7

## The Need for Sequence Numbers



- In the case of ACK loss (or poor choice of timeout) the receiver can't distinguish this message from the next
  - Need to understand how many packets can be outstanding and number the packets; here, a single bit will do

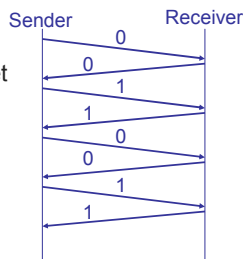
May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

8

## Stop-and-Wait

- Only one outstanding packet at a time
- Also called alternating bit protocol



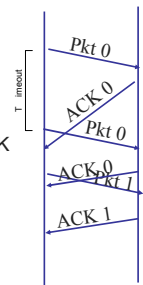
May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

9

## How does receiver recognize a duplicate?

- Sequence # in packet is finite
  - How many bits do we need?
    - One bit for stop and wait
    - Won't send seq #1 until receive ACK for seq #0
    - Only allows one packet in flight



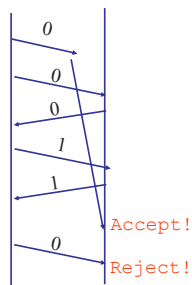
May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

10

## What if packets are delayed?

- Never reuse a seq #? Finite...
- Require in order delivery?
- Prevent very late delivery?
  - TTL: Decrement hop count per packet, discard if exceeded
  - Seq #s not reused within delay bound
- Trust issues?



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

11

## What happens if a machine crashes?

- How do we distinguish packets sent before and after reboot? Which seq# to use?
- Solutions
  - Restart sequence # at 0?
  - Assume boot takes max packet delay?
  - Choose seq # at random and hope?
  - Use stable storage and increment high order bits of seq # on every boot
- Reality: People don't worry about this
  - Slow reboots, explicit connection management, tolerant users

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

12

## Limitations of Stop-and-Wait



- Lousy performance if wire time  $\ll$  prop. delay
  - How bad?
- Want to utilize all available bandwidth
  - Need to keep more data "in flight"
  - How much? Remember the bandwidth-delay product?
- Also limited by quality of timeout (how long?)

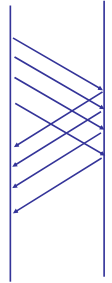
May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

13

## Pipelined transmission

- Send multiple packets without waiting for the first to be ACKed
- Reliable, unordered delivery:
  - Send new packet after each ACK
  - Sender keeps list of unACK'ed packets and resends after timeout
  - Receiver same as stop & wait
- Prob: What if packet 2 keeps being lost?
  - Receiver must buffer all packets after 2
  - Potential buffer overflow

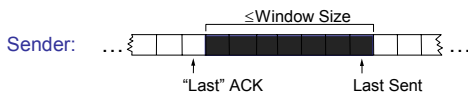


May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

14

## Sliding Window – Sender



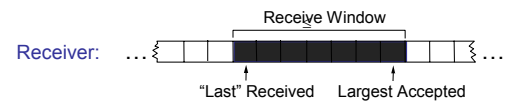
- Window bounds outstanding data
  - Implies need for buffering at sender
- "Last" ACK applies to in-order data
- What to do on a timeout?
  - Go-Back-N: one timer, send all unacknowledged data on timeout
  - Selective Repeat: timer per packet, resend as needed

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

15

## Sliding Window – Receiver



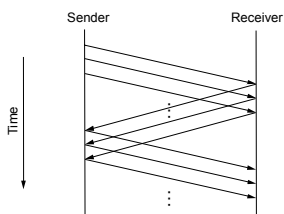
- Receiver buffers too:
  - data may arrive out-of-order
  - or faster than can be consumed (flow control)
- Receiver ACK choices:
  - Individual, Cumulative (TCP), Selective (newer TCP), Negative

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

16

## Sliding Window – Timeline



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

17

## Sliding Window Functions

- Sliding window is a mechanism
- It supports multiple functions:
  - Reliable delivery
  - In-order delivery
  - Flow control

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

18

## Deciding When to Retransmit

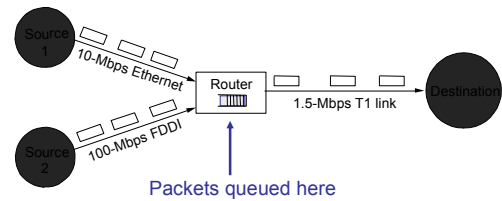
- How do you know when a packet has been lost?
  - Ultimately sender uses timers to decide when to retransmit
- But how long should the timer be?
  - Too long: inefficient (large delays, poor use of bandwidth)
  - Too short: may retransmit unnecessarily (causing extra traffic)
- Right timer is based on the round trip time (RTT)
  - Which varies greatly (path length and queuing)

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

19

## A Simple Network Model



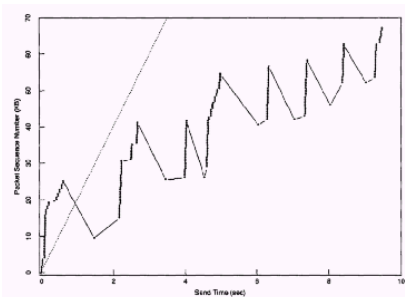
- Buffers at routers used to absorb bursts when input rate > output
- Loss (drops) occur when sending rate is persistently > drain rate

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

20

## Effects of Early Retransmissions



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

21

## Congestion Collapse

- In the limit, early retransmissions lead to congestion collapse
  - Sending more packets into the network when it is overloaded exacerbates the problem of congestion
  - Network stays busy but very little useful work is being done
- This happened in real life ~1987
  - Led to Van Jacobson's TCP algorithms, which form the basis of congestion control in the Internet today [See "Congestion Avoidance and Control", SIGCOMM'88]

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

22

## Estimating RTTs

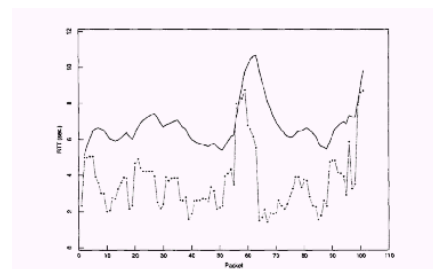
- Idea: Adapt based on recent past measurements
- Simple algorithm:
  - For each packet, note time sent and time ack received
  - Compute RTT samples and average recent samples for timeout
- EstimatedRTT =  $\alpha \times \text{EstimatedRTT} + (1 - \alpha) \times \text{SampleRTT}$
- This is an exponentially-weighted moving average that smoothes the samples. Typically,  $\alpha = 0.8$  to  $0.9$ .
- Set timeout to small multiple (2) of the estimate to capture variation around mean.

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

23

## Estimated Retransmit Timer



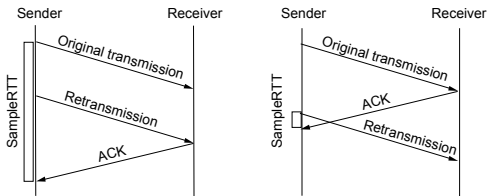
May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

24

## Karn/Partridge Algorithm

- Problem: RTT for retransmitted packets ambiguous



- Solution: Don't measure RTT for retransmitted packets and do not relax backed of timeout until valid RTT measurements

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

25

## Jacobson/Karels Algorithm

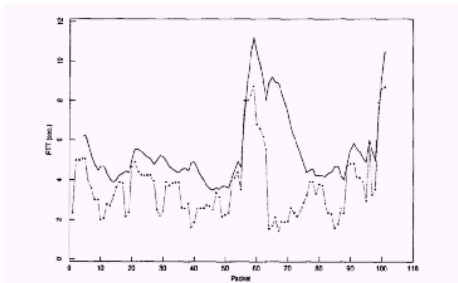
- Problem:
  - Variance in RTTs gets large as network gets loaded
  - So an average RTT isn't a good predictor when we need it most
- Solution: Track variance too.
  - Difference = SampleRTT - EstimatedRTT
  - EstimatedRTT = EstimatedRTT + ( $\delta$  x Difference)
  - Deviation = Deviation + ( $\delta$ (|Difference| - Deviation))
  - Timeout =  $\mu$  x EstimatedRTT +  $\phi$  x Deviation
  - In practice,  $\delta = 1/8$ ,  $\mu = 1$  and  $\phi = 4$

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

26

## Estimate with Mean + Variance



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

27

## Can we shortcut the timeout?

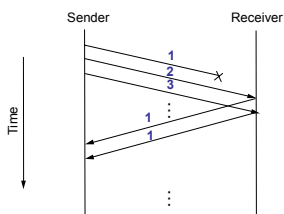
- Problem
  - If a packet is dropped in the network, the sender has to wait until timeout occurs before reacting
- If packets are usually in order then out-of-order packets imply that a packet was lost
  - Negative ACK
    - » Receiver requests missing packet
  - *Fast retransmit*
    - » Receiver ACKs out-of-order packets with seq# of last contiguous packet
    - » When sender receives multiple **duplicate** acknowledgements resends missing packet

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

28

## Fast retransmit



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

29

## Alternatives to retransmission?

- Redundancy
  - Send additional data to compensate for lost packets
- Why not use retransmission
  - Multicast
    - » Lots of receivers
      - If each one ACK/NAK then hard to scale
        - Lots of messages
        - Lots of state
    - » Heterogeneous receivers
      - Modem vs 100Mbps connected hosts
  - One-way or very long delay channels (spacecraft)

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

30

## Simplest version

- Send every packet twice
- Must lose both packets in a pair to prevent message from being delivered

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

31

## Generalization: Forward Error Correction (FEC)

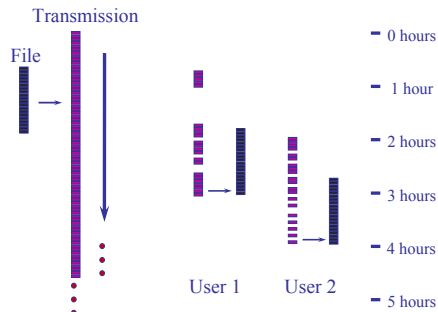
- Use erasure codes to redundantly encode  $k$  source packets into  $k*m$  encoded packets
  - Reed Solomon Codes
  - Tornado codes
- Multicast/broadcast encoded packets continually
- Any receiver can reconstruct message from any  $k$  packets in the set of  $k*m$

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

32

## Sometimes referred to as a “Digital Fountain”



May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

33

## Pros and Cons of Forward Error Correction

- Pro
  - Every packet can be useful for all clients
  - Well suited to multicast situation
- Con
  - Sends more data than ideally necessary
  - Need large block sizes for efficiency

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

34

## Summary

- Transport layer allows processes to communicate with stronger guarantees, e.g., reliability
- Reliability mechanisms
  - ARQ
    - » Sliding Window + retransmission for efficiency
    - » Retransmission timer must be adaptive
  - FEC
    - » In restricted settings

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

35

## For next time...

- Remember that next time is a week from today
- Read Ch 6.3-6.4

May 8, 2002

CSE 123b -- Lecture 3 -- Reliable Transmission

36