6.033 Spring 2017

Lecture #10

• Reliable Transport
• Window-based Congestion Control
How do we route (and address) scalably, while dealing with issues of policy and economy?

How do we transport data scalably, while dealing with varying application demands?

How do we adapt new applications and technologies to an inflexible architecture?
Reliable Transport

Sending Application

Reliable Sender

unreliable network

Receiving Application

Reliable Receiver

each byte of data is delivered exactly once and in-order
notice that this timeout happened before the sender got an ACK indicating that 7 had been received
question: what is the correct value for $w$?

too small $\rightarrow$ underutilized network

too large $\rightarrow$ congestion
**question:** how can a single reliable sender, using a sliding-window protocol, set its window size to maximize utilization — but prevent congestion and unfairness — given that there are many other end points using the network, all with different, changing demands?
something has happened to packet 7
AIMD + Slow Start

Window Size vs. Time (RTTs)

Retransmission due to timeout
• **TCP** provides **reliable transport** along with **congestion control**: senders increase their window additively until they experience loss, and then back off multiplicatively. Senders also use slow-start and fast-retransmit/fast-recovery to quickly increase the window and recover from loss.

• **TCP** has been a massive success, but **senders don’t react to congestion until queues are already full**. Is there a better way?