


Computer Networks

CMSC 417 : Spring 2024

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

Topics:

A) TCP vulnerabilities (Research paper)

B) Link layer: Introduction, Ethernet (Textbook chapter 2)

Nirupam Roy

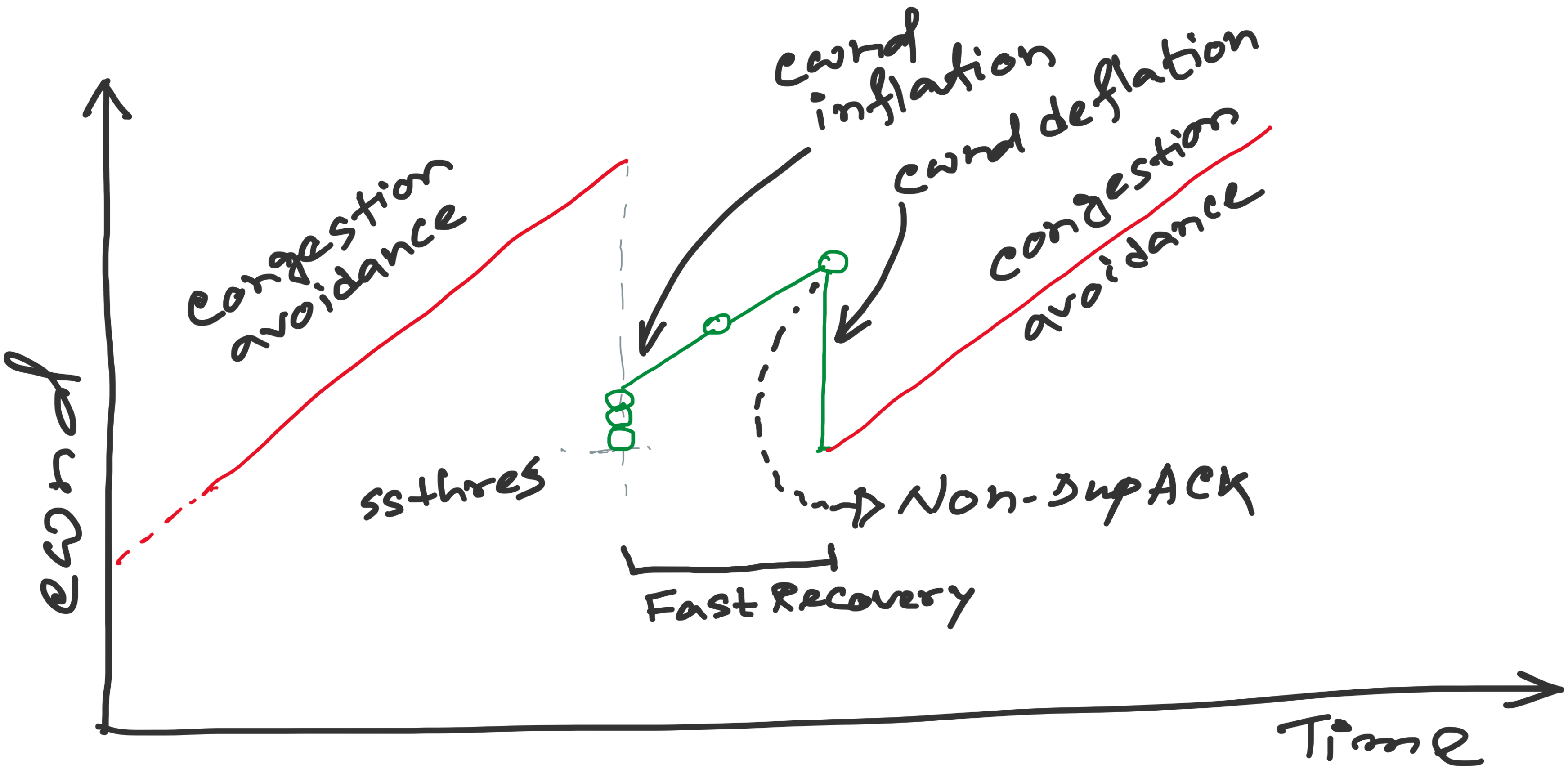
Tu-Th 2:00-3:15pm

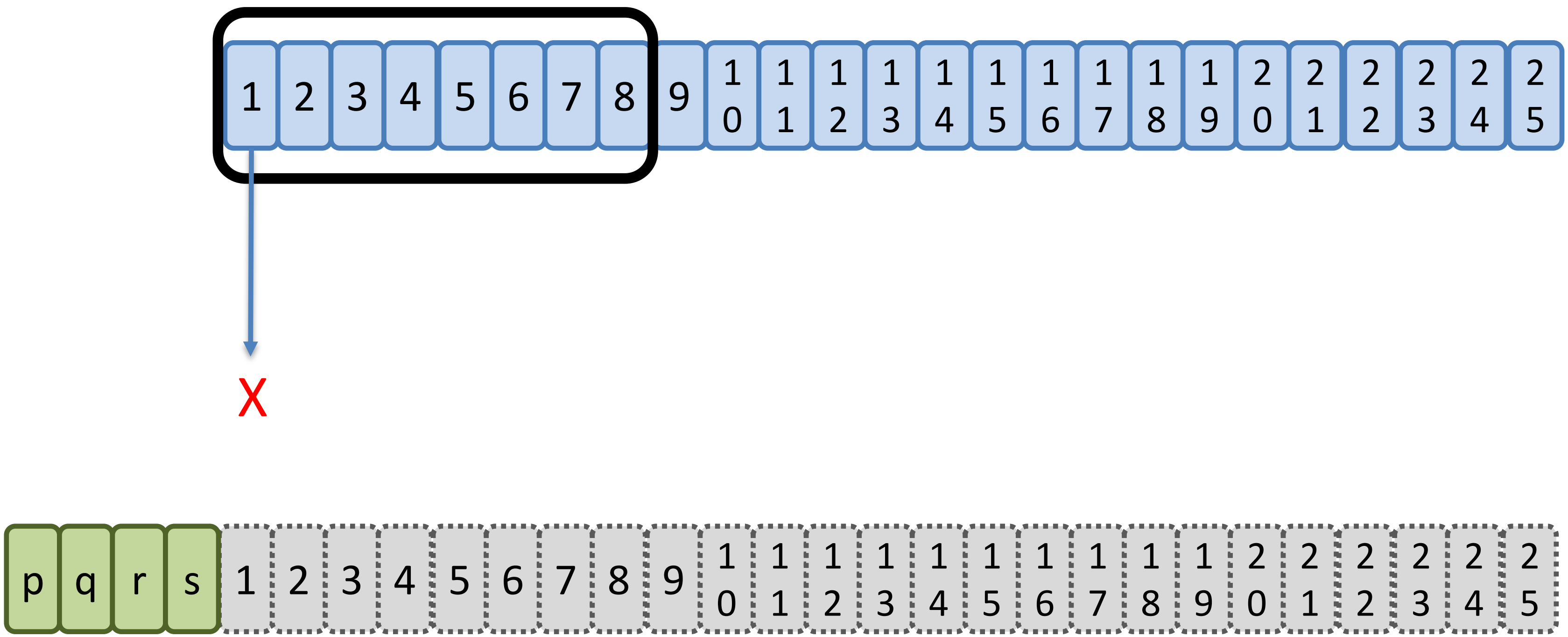
CSI 2117

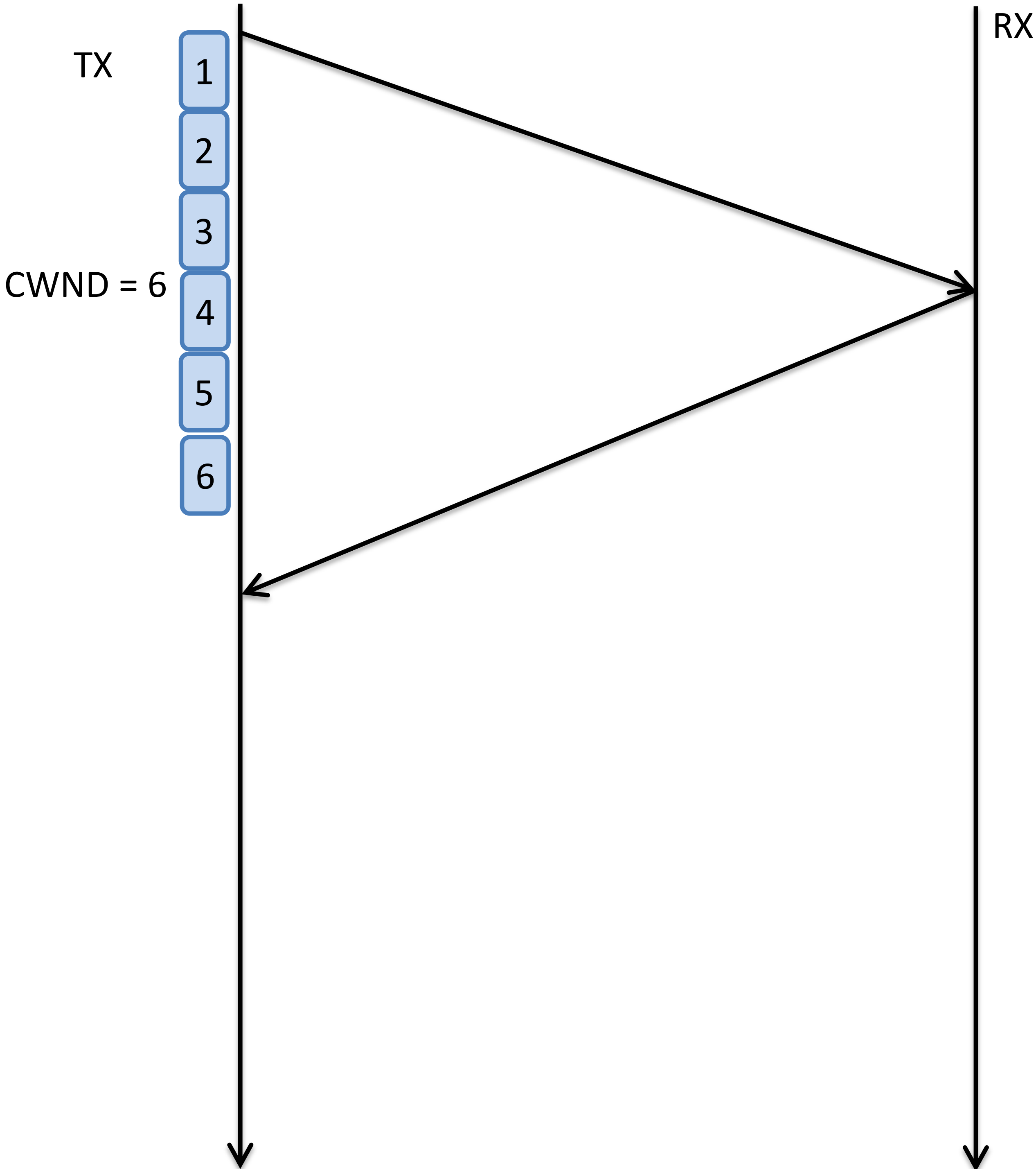
April 4th, 2024



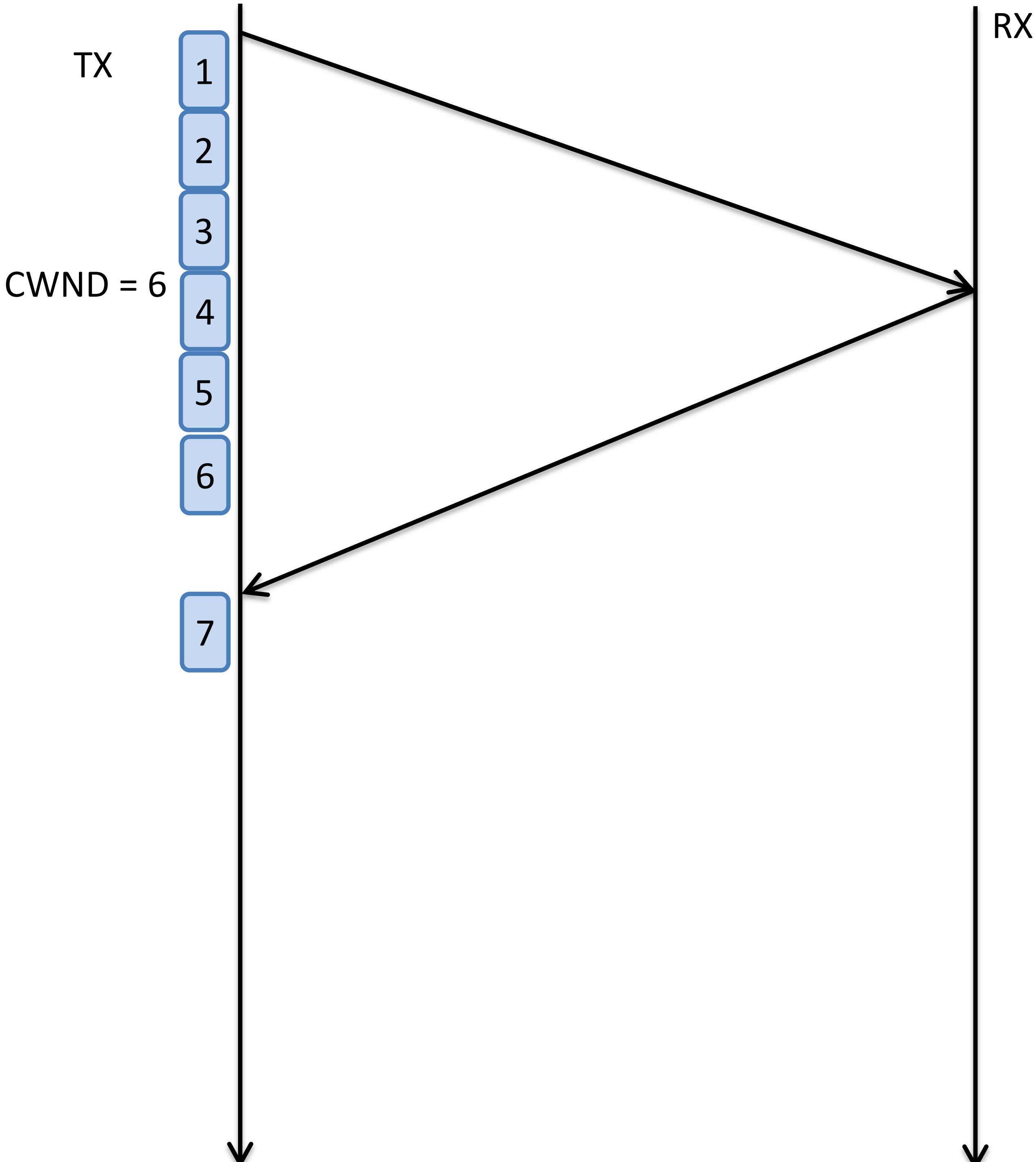

TCP Reno: details of Fast Recovery

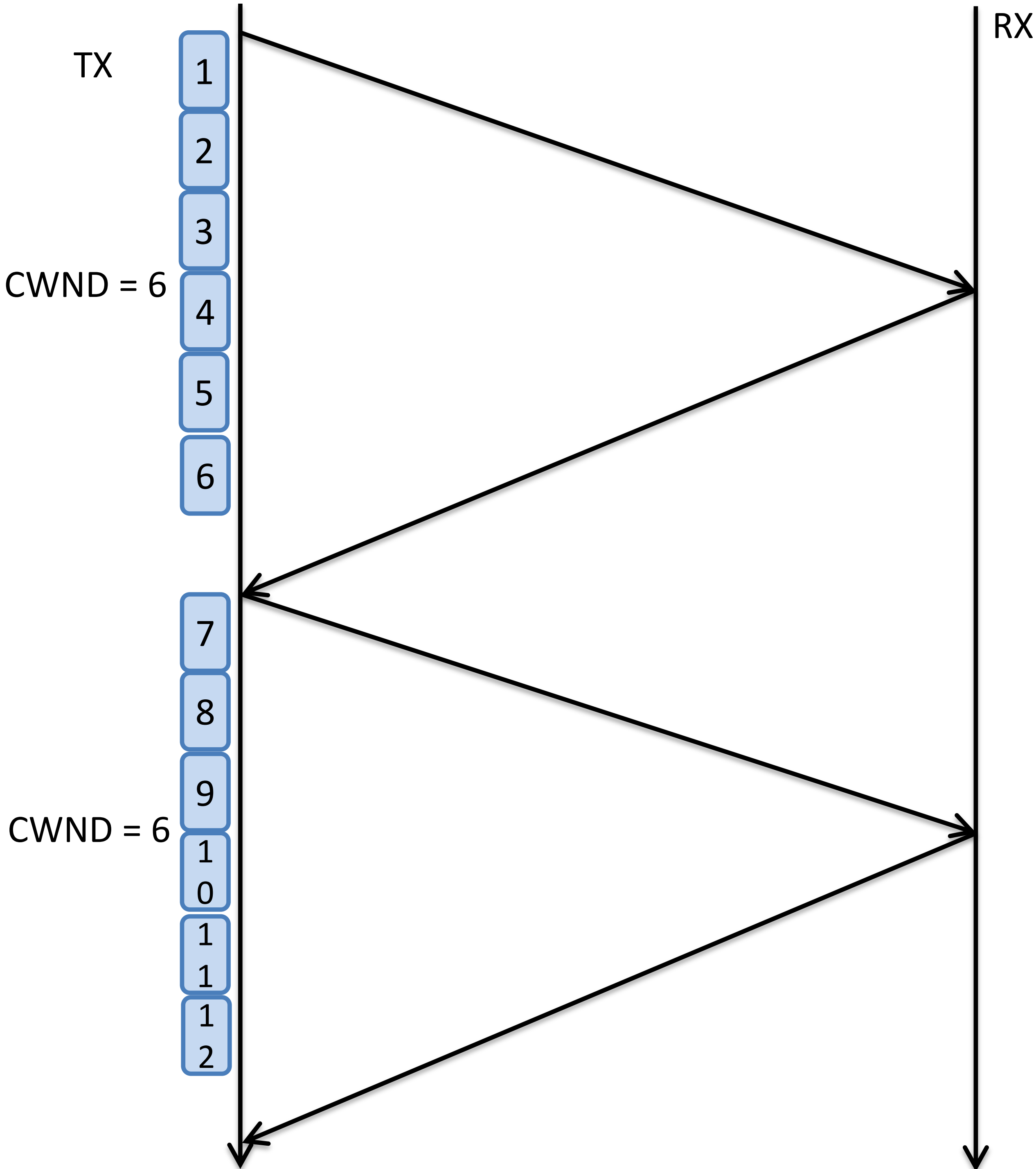


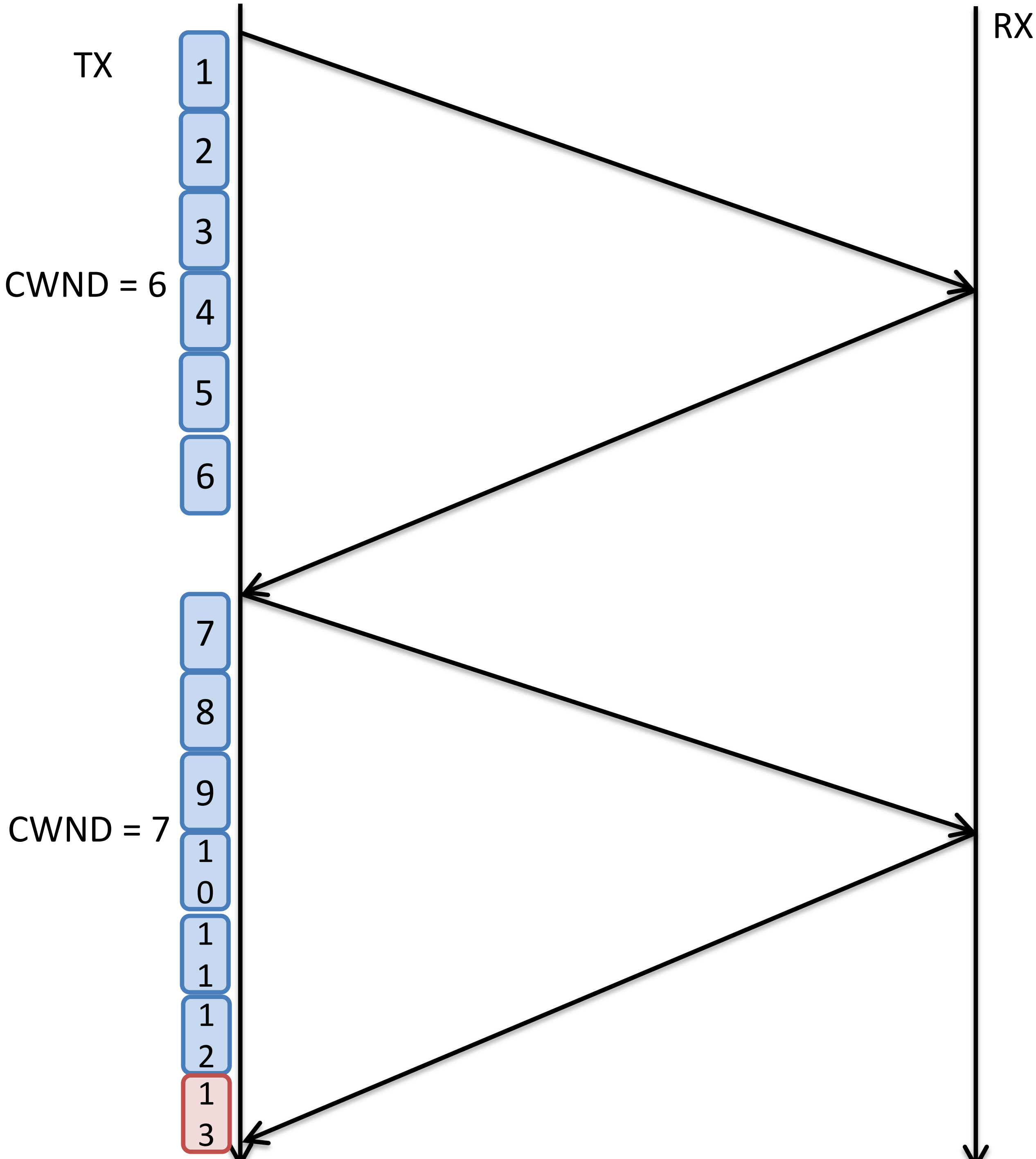

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
X
p q r s 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

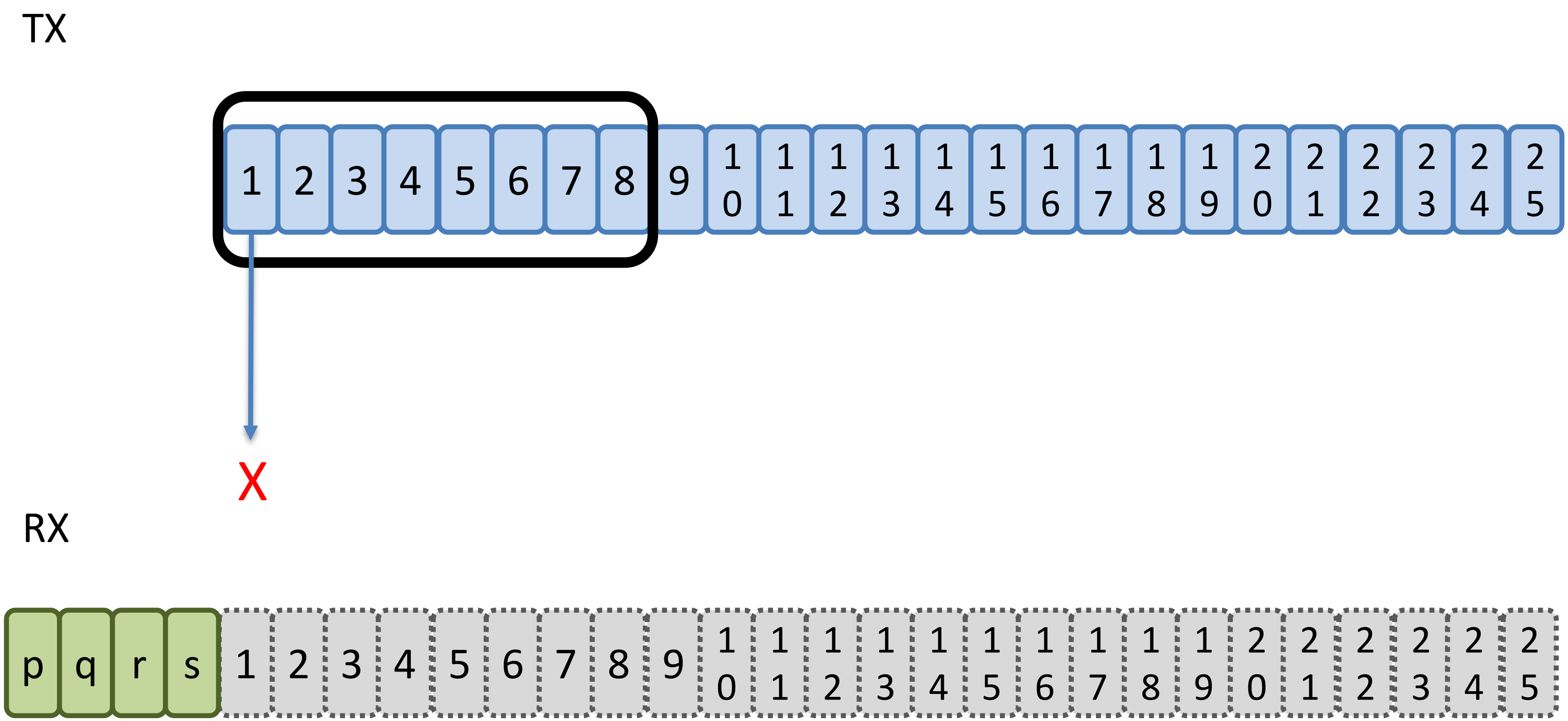


TX
RX
CWND = 6
1
2
3
4
5
6


TX
RX
CWND = 6
1
2
3
4
5
6
7


TX
RX
CWND = 6
1
2
3
4
5
6
7
8
9
10
11
12
CWND = 6


TX
RX
CWND = 6
1
2
3
4
5
6
7
8
9
CWND = 7
10
11
12
13


TX
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
X
RX
p q r s 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

TCP Congestion Control with a Misbehaving Receiver

Stefan Savage, Neal Cardwell, David Wetherall, and Tom Anderson
Department of Computer Science and Engineering
University of Washington, Seattle

Slides courtesy: https://cseweb.ucsd.edu/classes/wi01/cse222/lectures/attack-18.pdf

The problem

- Bandwidth sharing on the Internet
  - Hosts **voluntarily** limit own data rate
  - Mechanism implemented in TCP
  - “Fair” rate determined by testing the network
  - Relies on **cooperation** between endpoints
- Why doesn’t everyone cheat?

One explanation

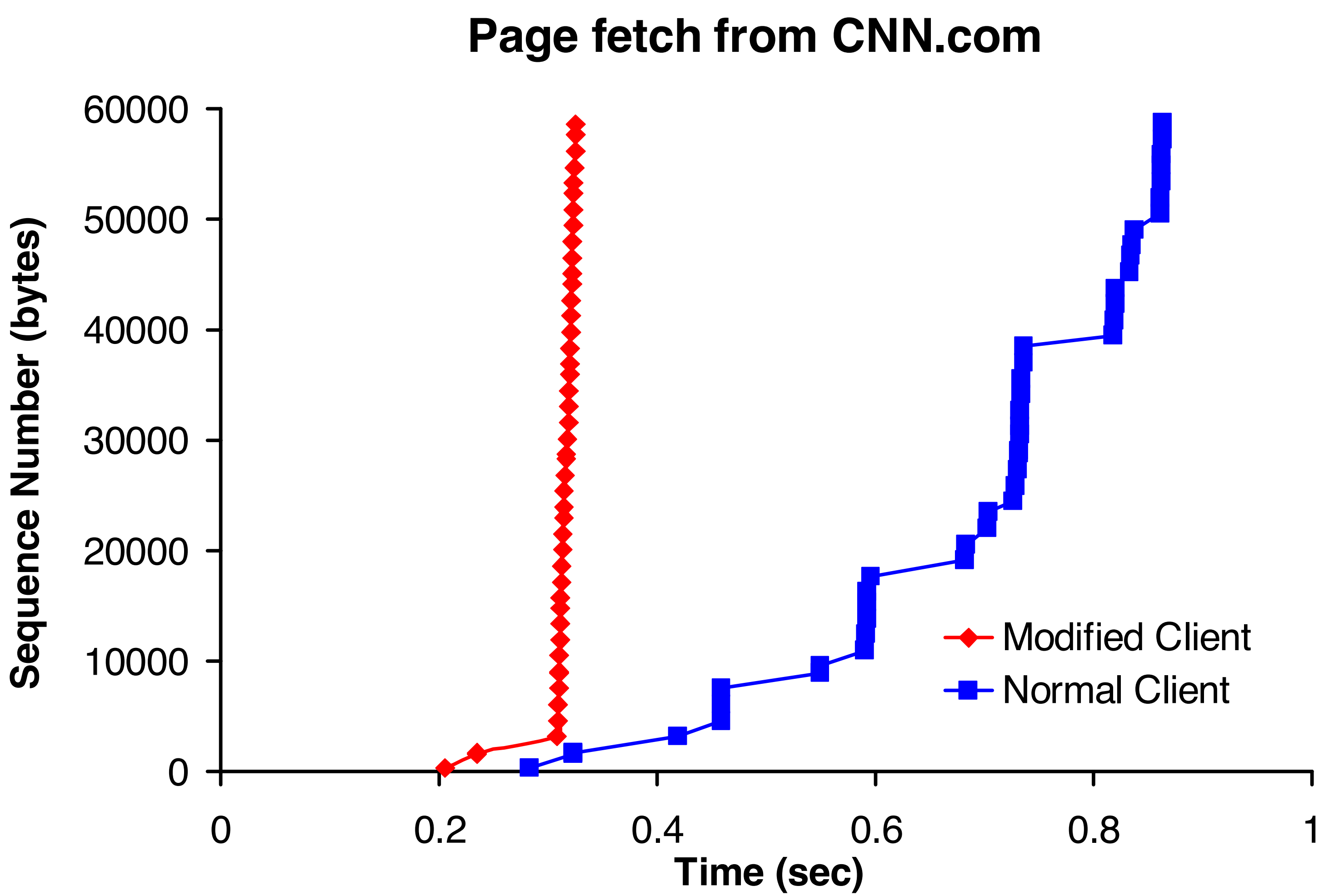
- Cheating requires motive **and** opportunity

- Senders (e.g., Web servers)
  - Have opportunity (could send too fast)
  - Limited motive (economic incentive to share)
- Receivers (e.g., Web clients)
  - Have competitive motive (faster Web surfing)
  - No opportunity (only receive data)… right?

What if receivers misbehave?

- A client can **implicitly** control the data rate of a remote server
  - This is not an implementation error
  - It is a weakness in the TCP specification
  - TCP's design does not consider that senders and receivers might have disjoint interests
- The vulnerability is significant…

Why the Web is faster in Seattle

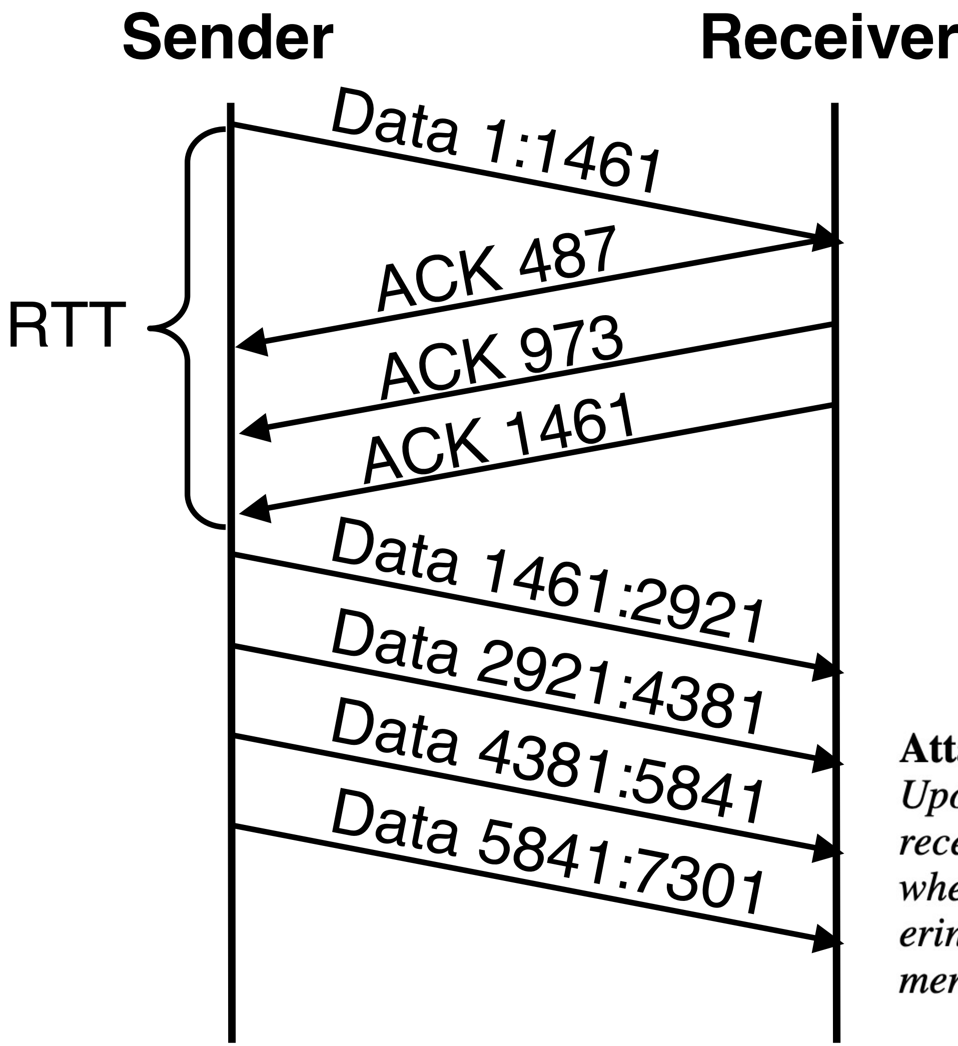


Vulnerability 1: Bytes vs. Segments

- TCP: reliable byte stream w/ cum. ACKs
- Cwnd limits unacknowledged data
- TCP begins a session in *slow start*:

  During slow start, TCP increments cwnd by at most SMSS bytes for each ACK received that acknowledges new data.

(1) ACK Division



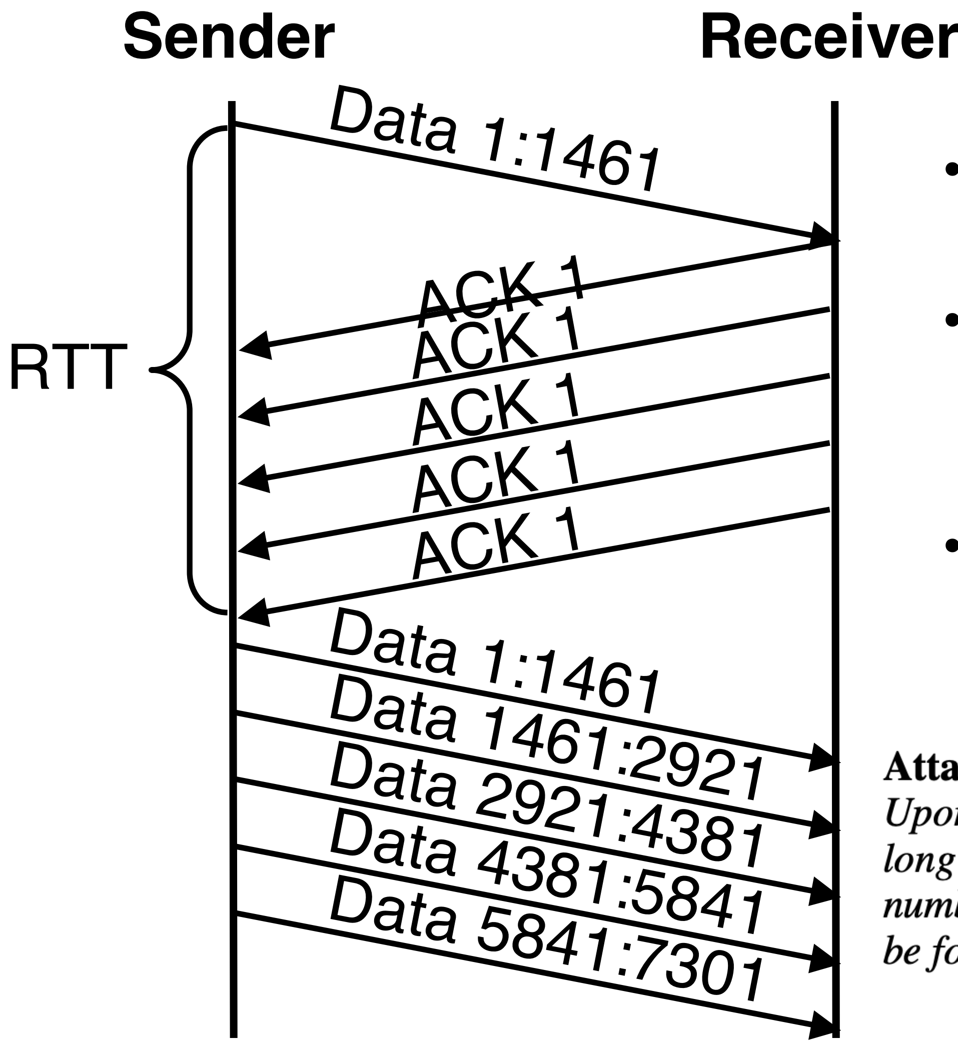
- Send M ACKs for one pkt
- Exponential growth factor proportional to M!
- Preserves end-to-end semantics

Attack 1:
Upon receiving a data segment containing N bytes, the receiver divides the resulting acknowledgment into M, where $M \leq N$, separate acknowledgments – each covering one of M distinct pieces of the received data segment.

Vulnerability 2: Fast Retransmit and Recovery

- Receive out-of-order segment => send dupack
- Sender receives 3 dupacks => fast retransmits, enters fast recovery
  - Cwnd = cwnd/2 + 3*SMSS
  - On a dupack, cwnd += SMSS

(2) DupACK Spoofing



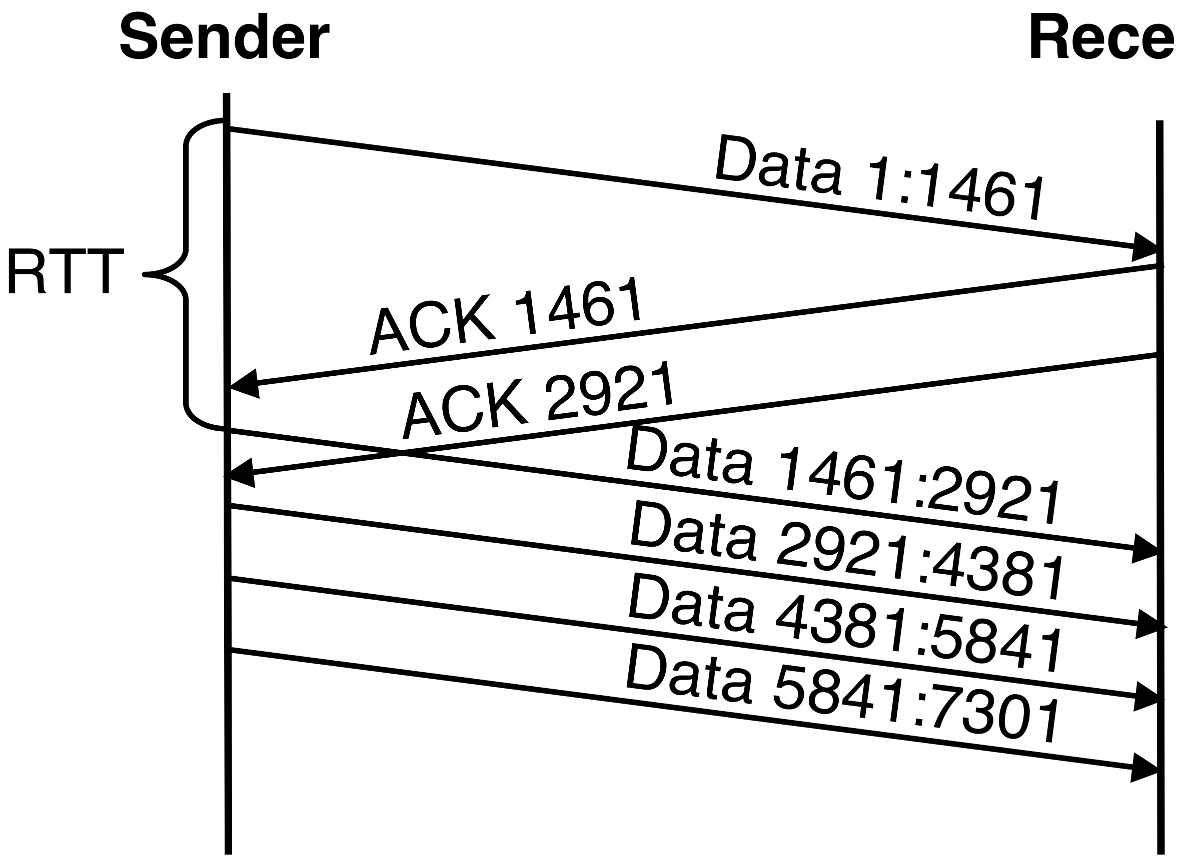
- Send extra duplicate ACKs
- Sender sends one pkt for each duplicate ACK
- Preserves end-to-end semantics

Attack 2: *Upon receiving a data segment, the receiver sends a long stream of acknowledgments for the last sequence number received (at the start of a connection this would be for the SYN segment).*

Vulnerability 3

- When sender receives a new ACK, it increases cwnd
- *But how do you know the receiver got the data?*

(3) Optimistic ACKing



- Send ACKs early
- Sender sends pkts in proportion to ACK rate
- Violates end-to-end semantics
- Lose reliability

Attack 3:
Upon receiving a data segment, the receiver sends a stream of acknowledgments anticipating data that will be sent by the sender.

Implementation experience

- "TCP Daytona"
  - Easy to implement (<75 lines in Linux)
  - Works against all popular sender TCP stacks
    - Solaris, NT, Linux, FreeBSD, Tru64, IRIX, HPUX, AIX
      - Linux 2.2 immune to ACK division
      - NT 4 immune to DupACK spoofing
- Fetches most web pages in 2 RTT
  - We have the world's fastest Web browser!

Simple Countermeasures

- Combating ACK Division:
  - Only increase cwnd when receiver ACKs >= 1 segment - Linux 2.2
  - Byte counting [Allman98, Allman99]
- Combating DupACK Spoofing:
  - Count outstanding segments
  - Ignore extra DupACKs
- Optimistic ACKing:
  - Randomize segment boundaries
  - Ignore ACKs unless they match a real boundary

“Semantics”

- Meaning of message
  - Literal
  - Implied by assumptions about other party
- How message is acted upon
- Two levels:
  - TCP <-> TCP
  - TCP->application




Computer Networks

CMSC 417 : Spring 2024

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

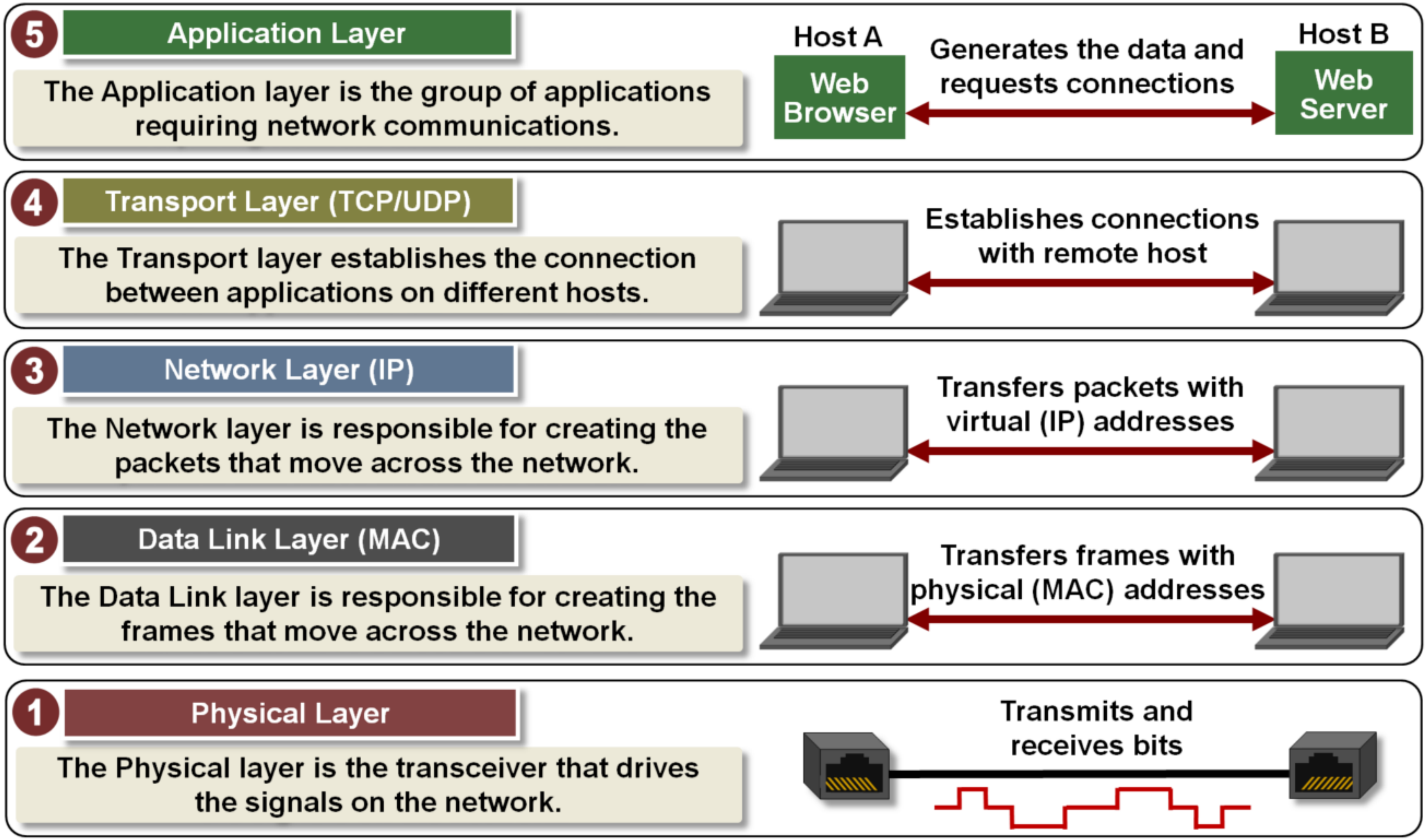
Topic: Link layer
(Textbook chapter 2)

Nirupam Roy

Tu-Th 2:00-3:15pm
CSI 2117




Protocol Layers



Link = Medium + Adapters

What is a Link?

Communication Medium

Network Adapter