Mobile Networking

Mohammad Hossein Manshaei

manshaei@gmail.com

1393

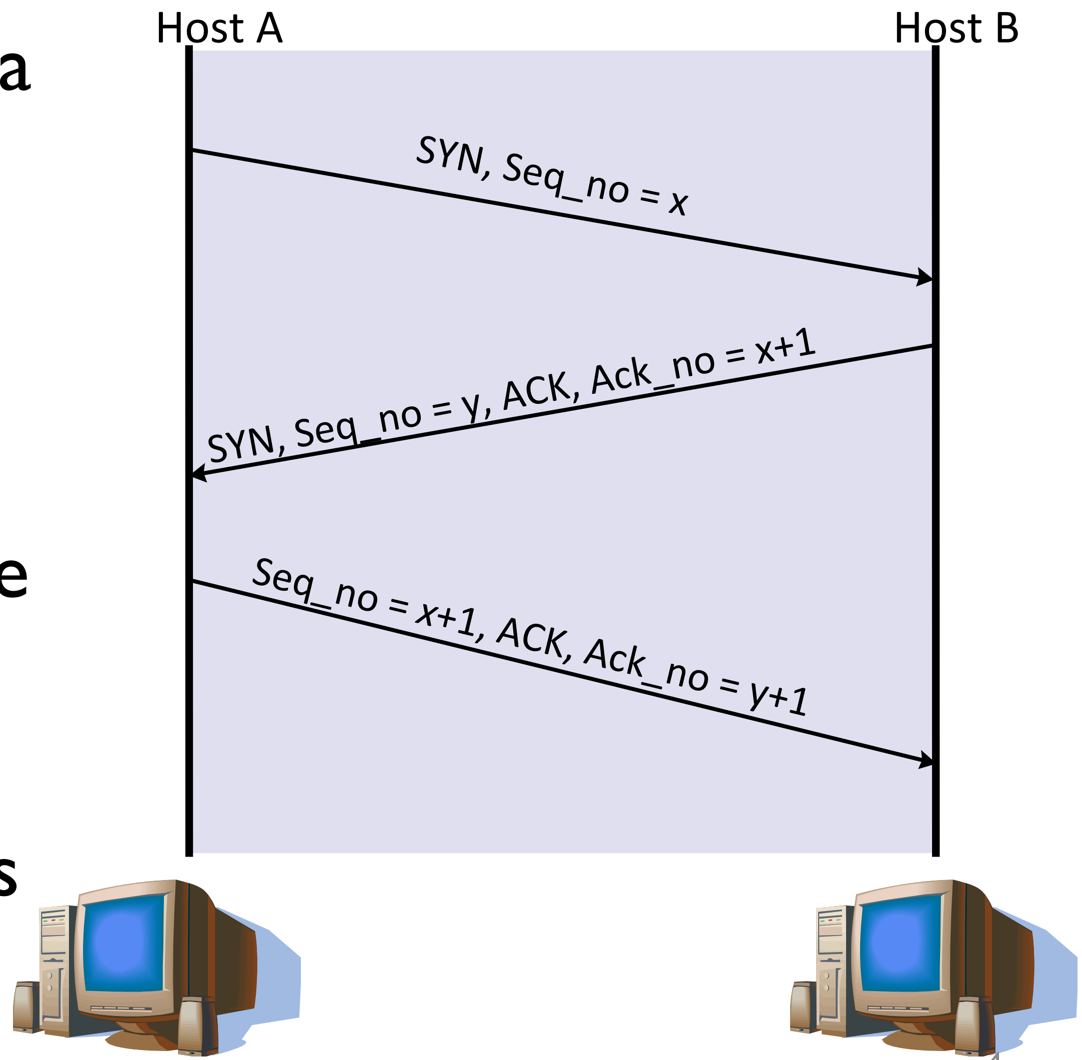
TCP over Wireless

MOBILE TRANSPORT LAYER

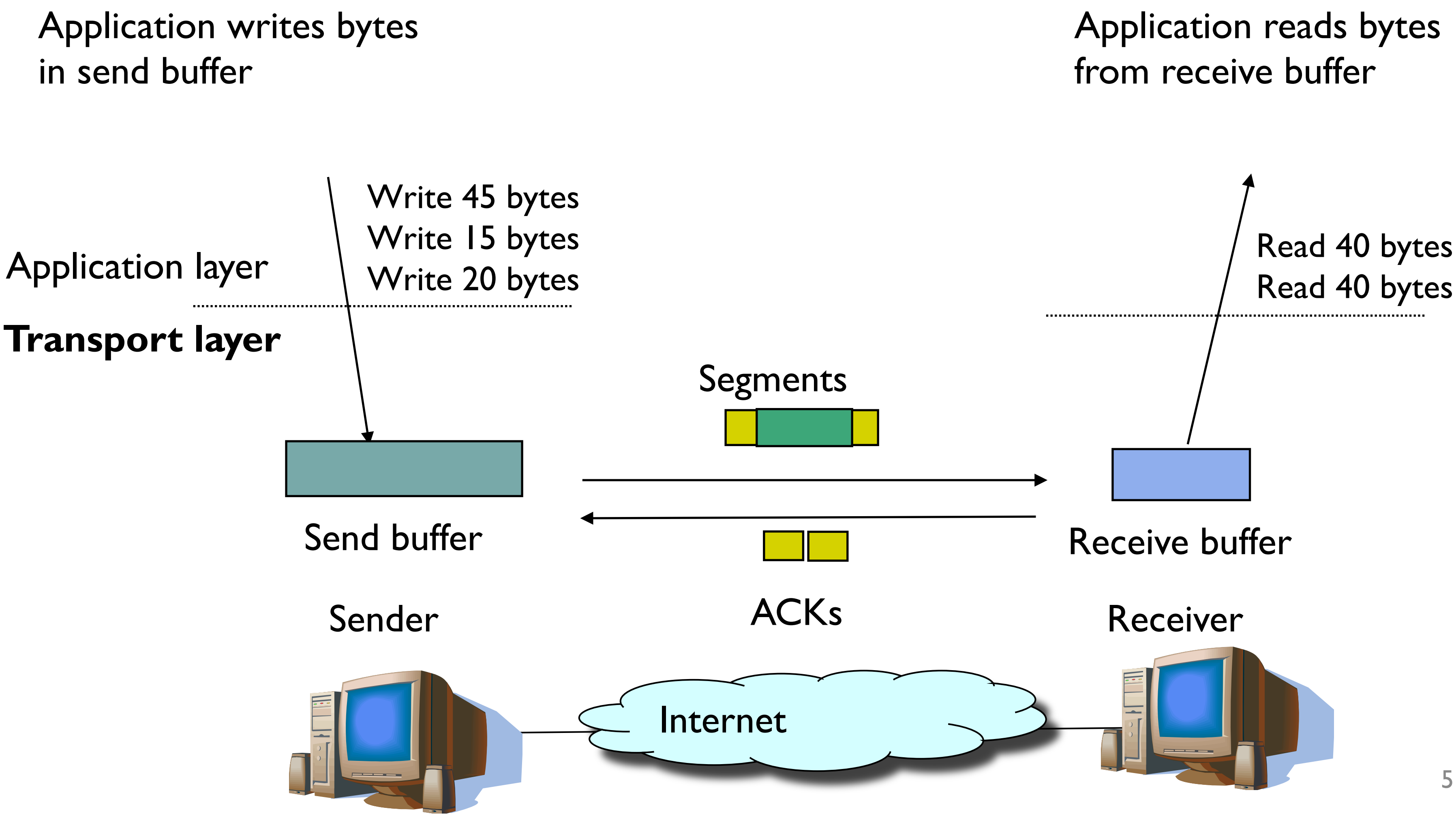
Contents

Transmission Control Protocol (TCP)

- Reliable, in-order data delivery
- Flow control
- Congestion avoidance and control
- End-to-end semantics



TCP Basic Operation



TCP flow control

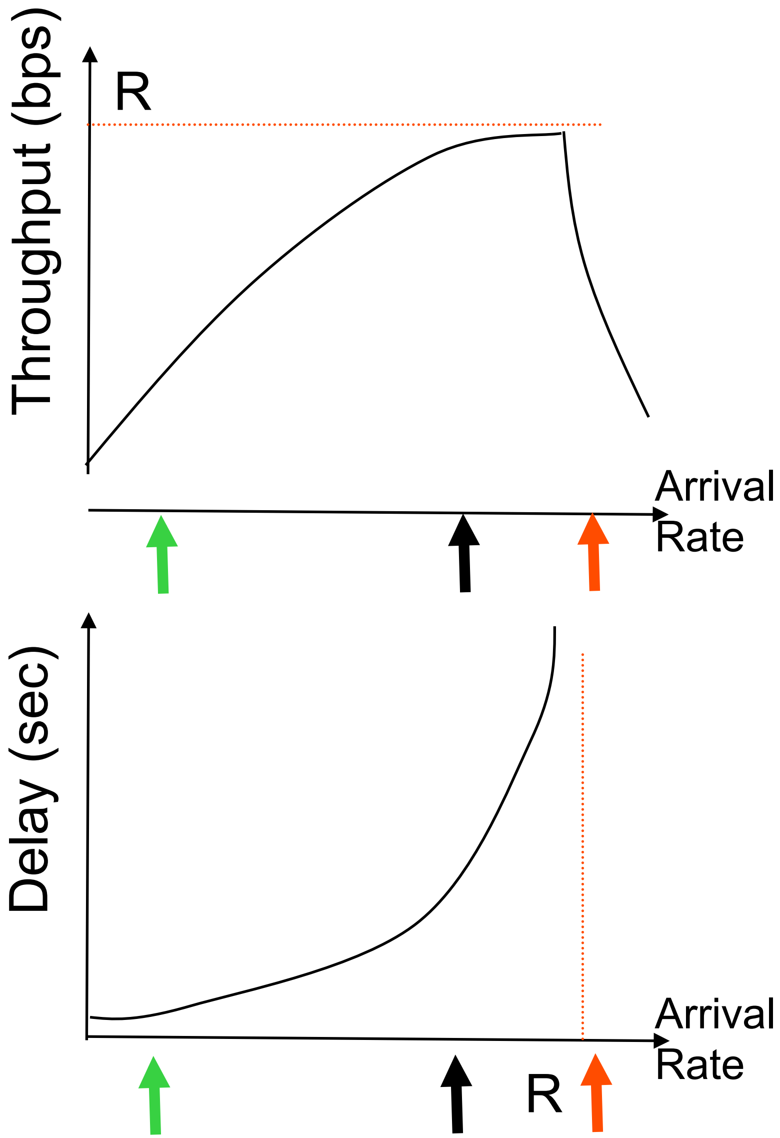
- Flow control is a speed-matching service
  – Sender adjusts the transmission rate to the receiver

- Receiver advertises the remaining buffer space (**rwnd)** to the sender

- The sender keeps unacknowledged data below **rwnd**

  LastByteSent – LastByteAcked ≤ rwnd

Congestion



Light traffic

- Arrival Rate << R
- Low delay
- Can accommodate more

Congestion onset

- Arrival rate approaches R
- Delay increases rapidly
- Throughput begins to saturate

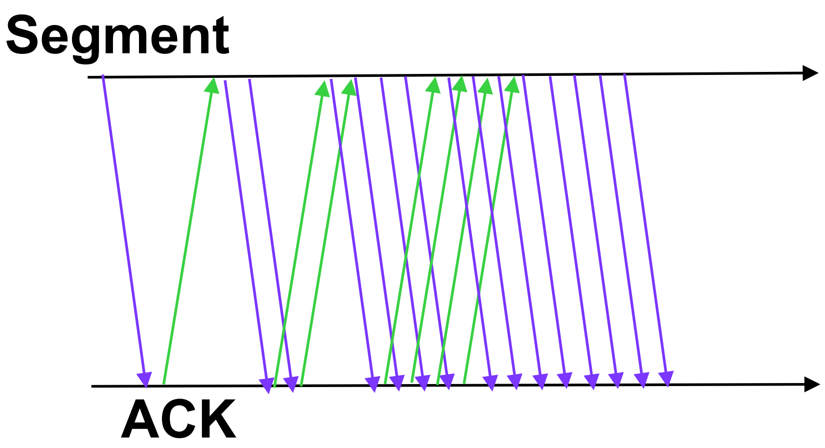
Saturation

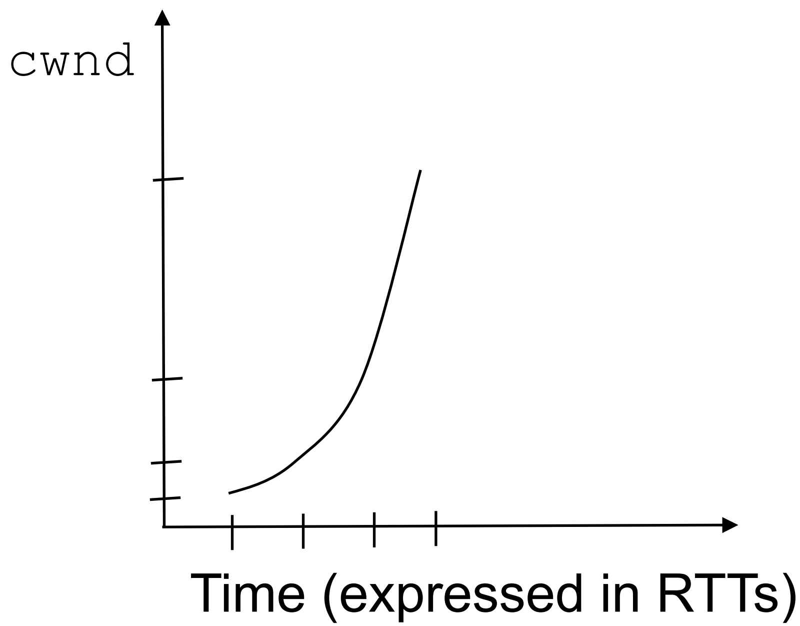
- Arrival rate > R
- Large delays, packet loss
- Useful application throughput drops

TCP Congestion Control

- Keeps TCP off the congestion collapse cliff

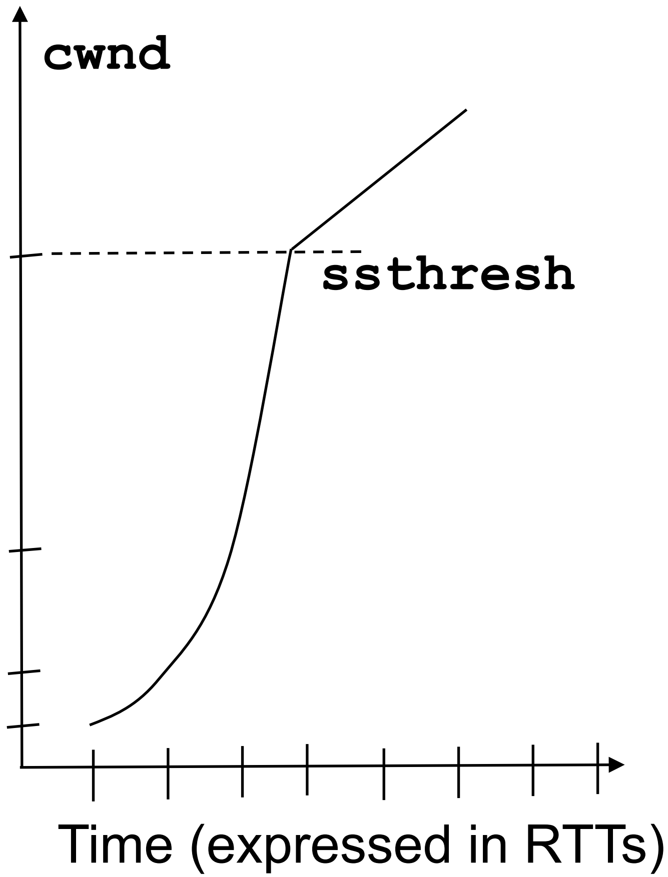
- Congestion window mechanism
  LastByteSent – LastByteAcked ≤ min{cwnd, rwnd}
- Slow Start phase
  - Increase congestion window size (cwnd) by one segment for each received ACK
  - Congestion window increases exponentially



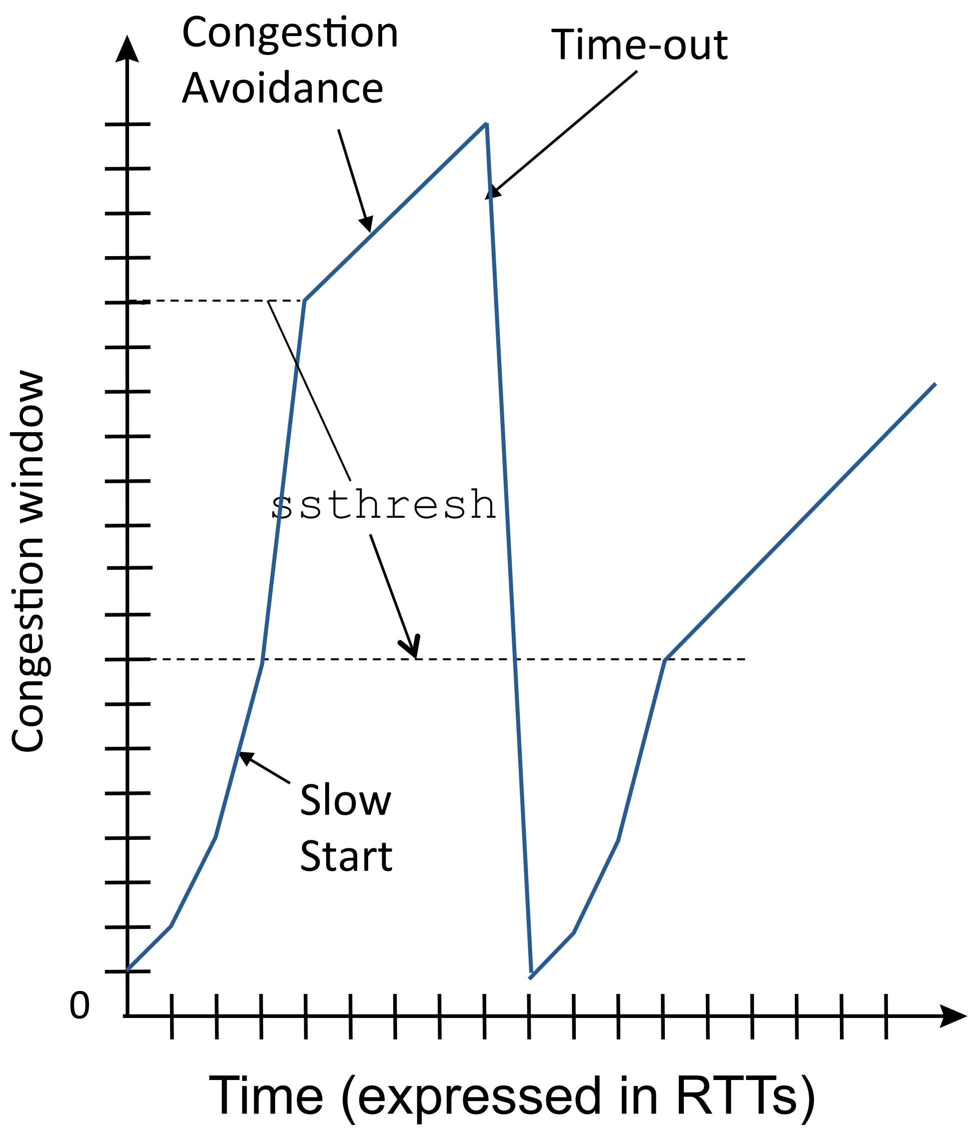


TCP Congestion Control

- Congestion Avoidance phase
  - Congestion threshold **ssthresh**
  - When **cwnd > ssthresh**, increase **cwnd** slowly
  - **cwnd++** per round-trip-time (RTT)
    - Each time an ACK arrives, **cwnd** is increased by 1/**cwnd**
    - In one RTT, **cwnd** segments are sent, so total increase in **cwnd** is **cwnd** x 1/**cwnd** = 1
    - **cwnd** grows linearly



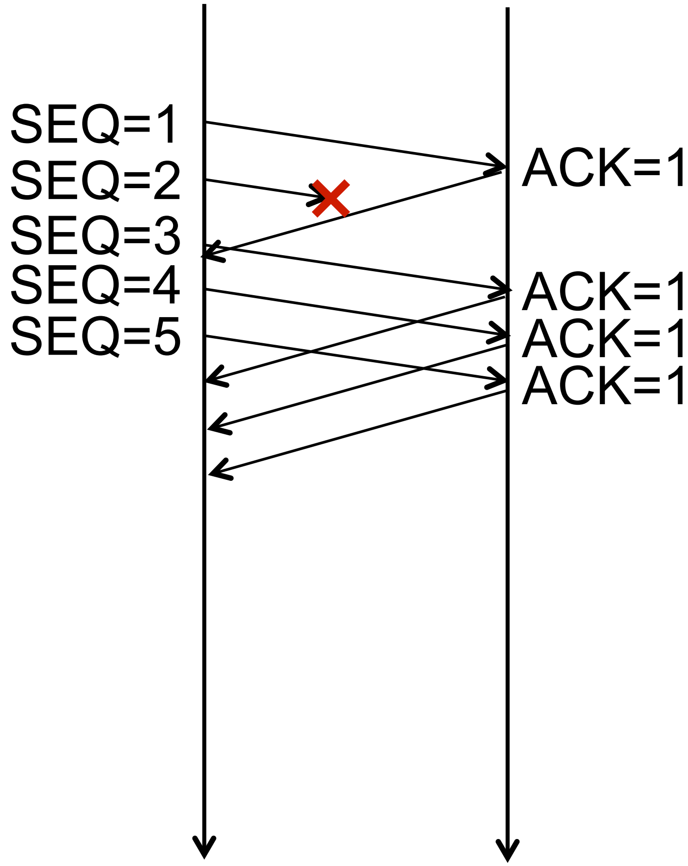
TCP Congestion Control



- **Congestion detection:**
  - Timeout or
  - Receipt of duplicate ACKs (Fast Retransmit)
- **Assumption**: current **cwnd** corresponds to available bandwidth
- TCP Tahoe
  - **ssthresh** = ½ **cwnd**
  - **cwnd** = 1
  - Go back to Slow Start
- Over several cycles expect to converge to **ssthresh** equal to about ½ the available bandwidth

TCP Congestion Control

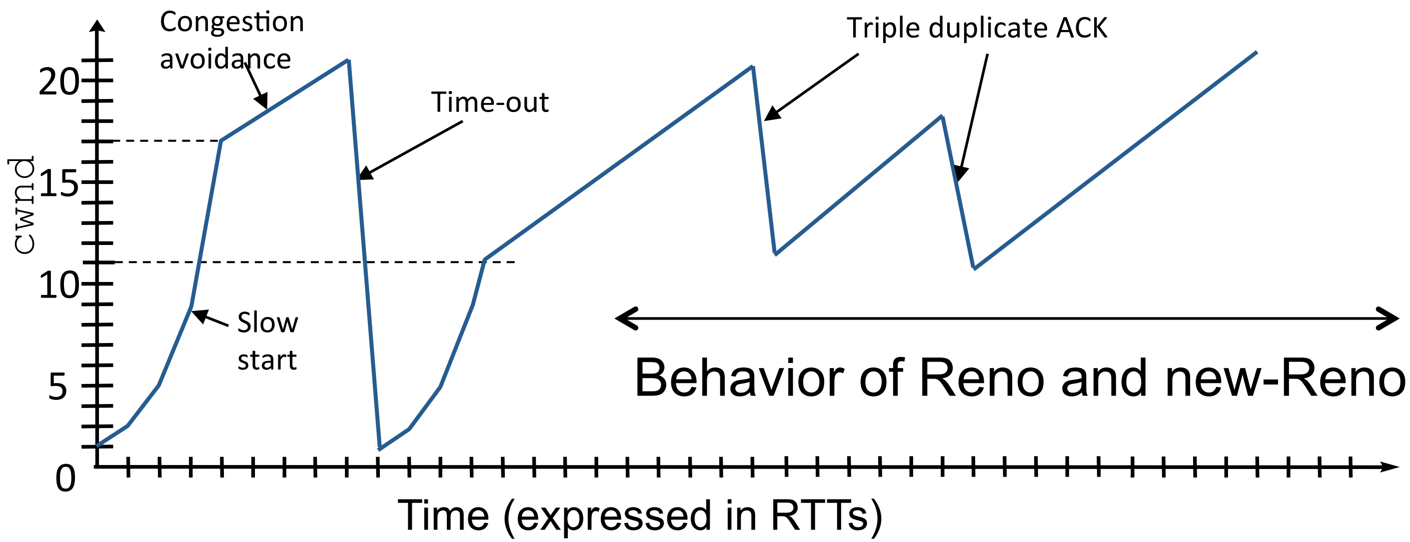
- **Fast Retransmit mechanism**
  - If a segment is dropped, subsequent segments trigger duplicate ACKs
  - Sender retransmits segment instantly (without waiting for a timeout) when duplicate ACKs are received (typically 3)



- **Improves performance**
  - Faster reaction to packet loss
- Implemented in **TCP-Reno** (more recent than TCP-Tahoe)

Performance comparison of TCP variants

- With single packet lost in congestion window, TCP Reno and TCP new-Reno avoid Slow Start, and outperform TCP Tahoe
  - **cwnd** oscillates around the optimal

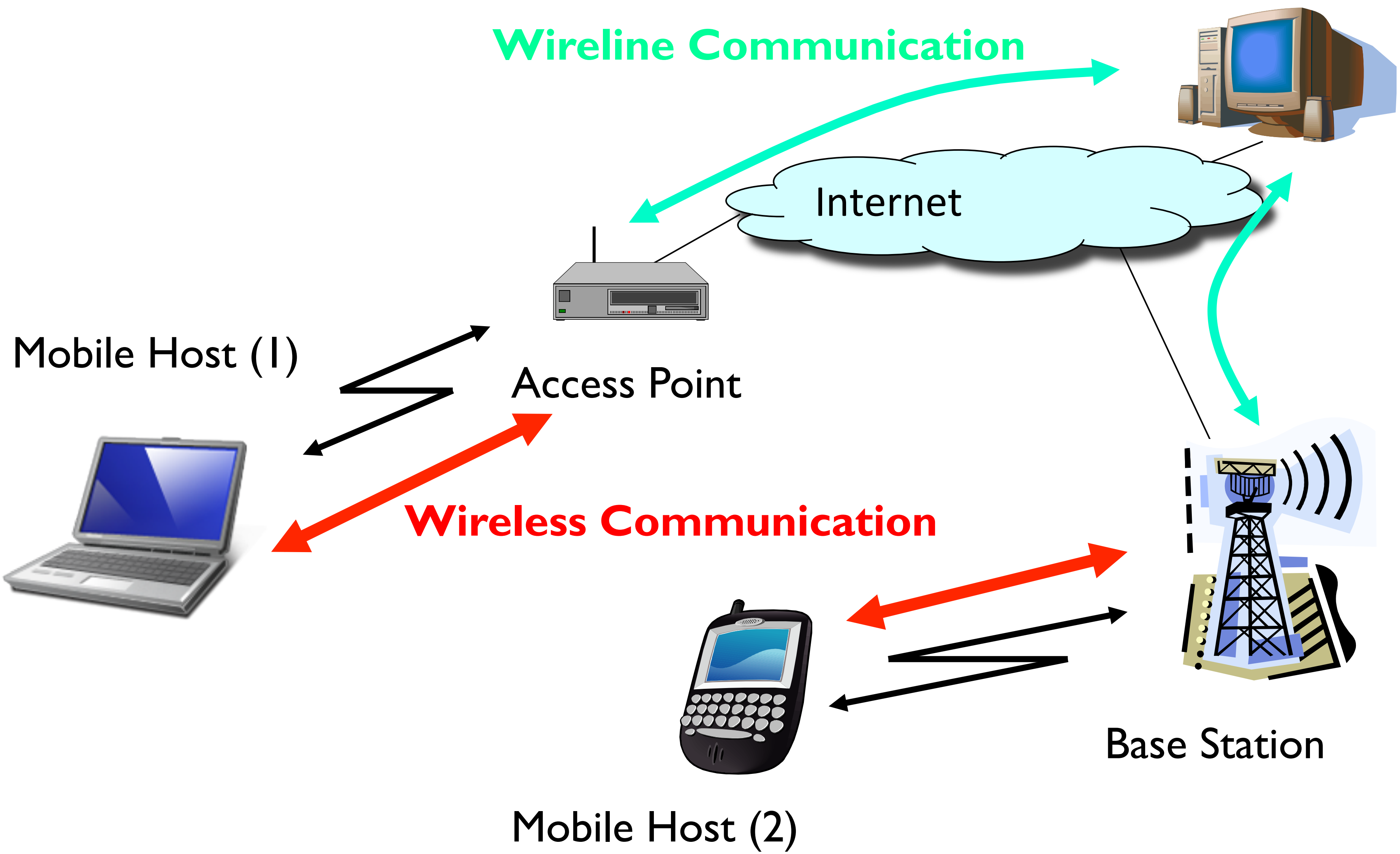


- With multiple packet lost per congestion window (not shown in the figure), TCP Reno underperforms severely; new-Reno introduced to resolve this
  - Bursty packet loss common in mobile networks

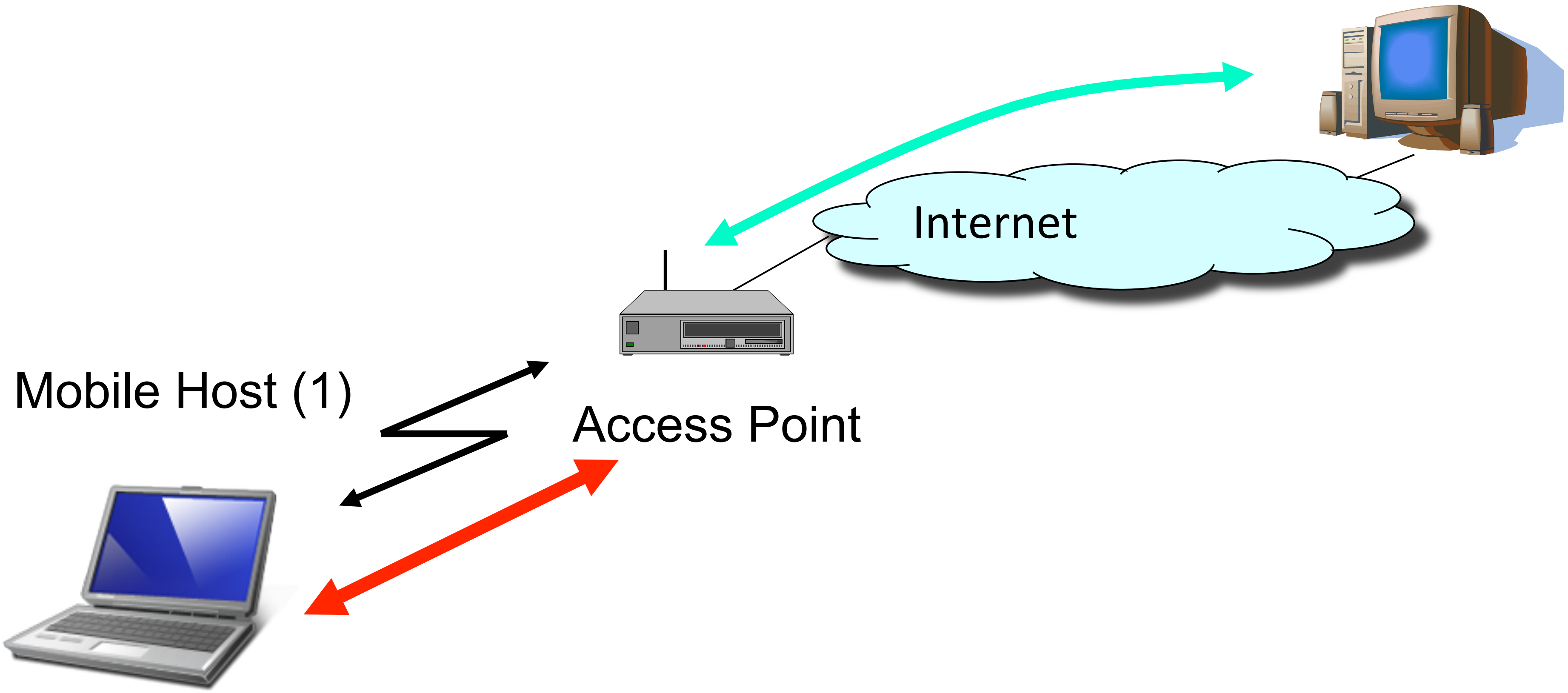
Contents

- Traditional TCP: A Brief Review
- TCP over Wireless: Challenges
- TCP over Wireless: Solutions
  - Indirect TCP
  - Snooping TCP
  - Mobile TCP
  - Fast Retransmit/Fast Recovery
  - Transmission/Time-out Freezing
  - Selective Retransmission
  - Transaction-Oriented TCP
- TCP over 2.5/3G Wireless Networks
- Summary

Wireless and Mobile Networks

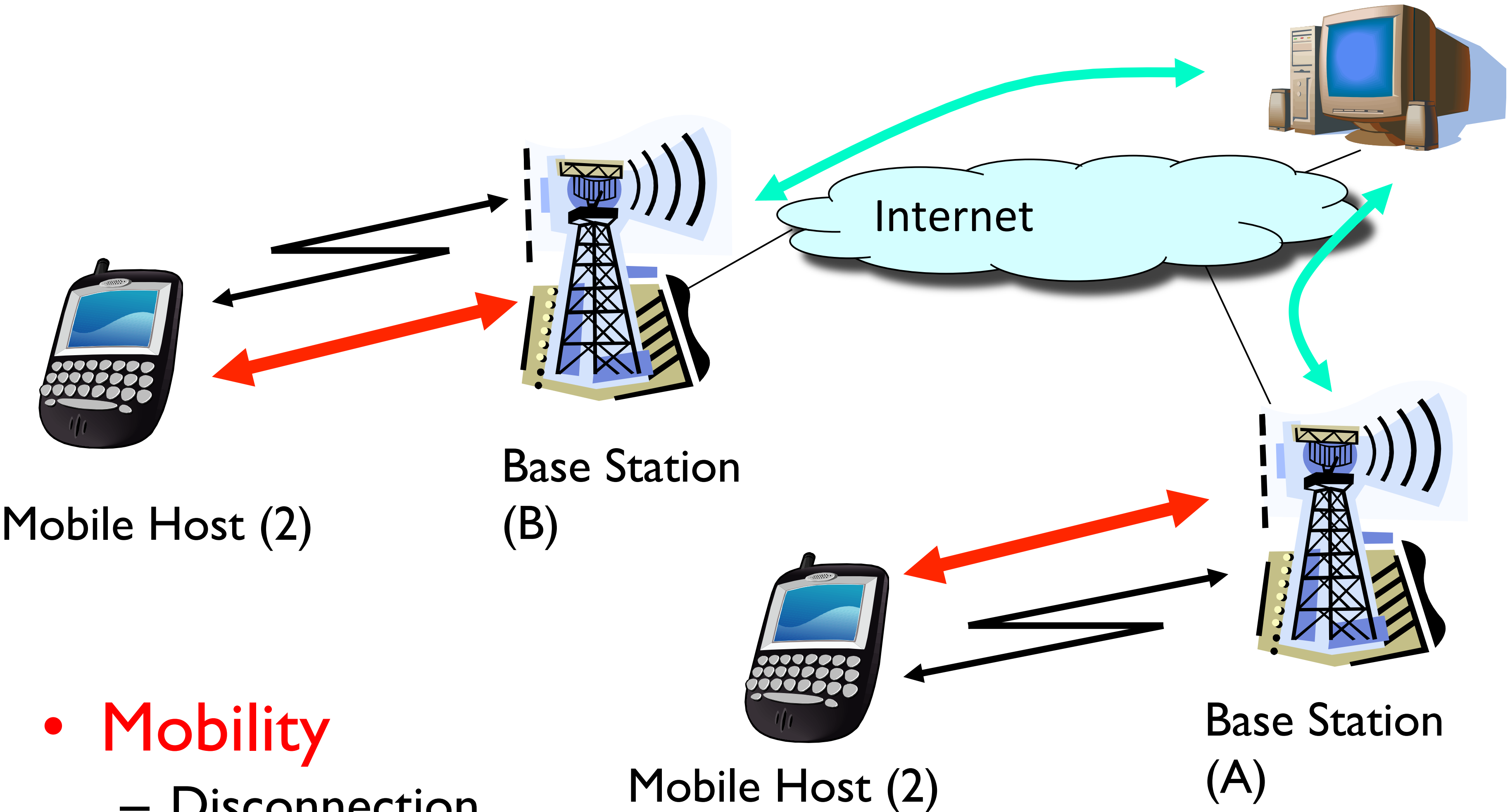


Wireless and Mobile Networks



- Wireless transmission errors
  - High Bit Error Rates
  - Packet (frame) loss

Wireless and Mobile Networks



- Mobility
  - Disconnection
  - Hand-offs
  - Delays

Challenge

- **TCP**
  - Assumptions for the Wire-line Internet
    - Packet loss only due to congestion
    - Packet loss is rare
- **Wireless and mobile networks**
  - TCP assumptions are not valid
- **Problem: TCP under-performs**
  - TCP can**<u>not</u>** distinguish between packet losses due to congestion and transmission or disconnection errors
  - Reducing the congestion window when an error or a disconnection occurs is **<u>not</u>** necessary
  - **<u>Throughput suffers</u>**

Questions

- What can we do about…
  - transmission errors?
  - errors due to mobility?

- Which part of the system functionality should we modify…
  - The sender?
  - The receiver?
  - An intermediate node?
  - Some or all of the above?

Directions

- **Transmissions errors**
  – Hide error losses from the sender
    - If the sender is unaware of the packet losses due to errors, it does not reduce the congestion window
  – Inform sender of packet loss cause
    - If the loss is due to an error, the congestion window is not reduced

- **Errors due to mobility**
  – Hide mobility from the TCP sender
  – Make TCP adaptive to mobility

Contents

Solution Classification

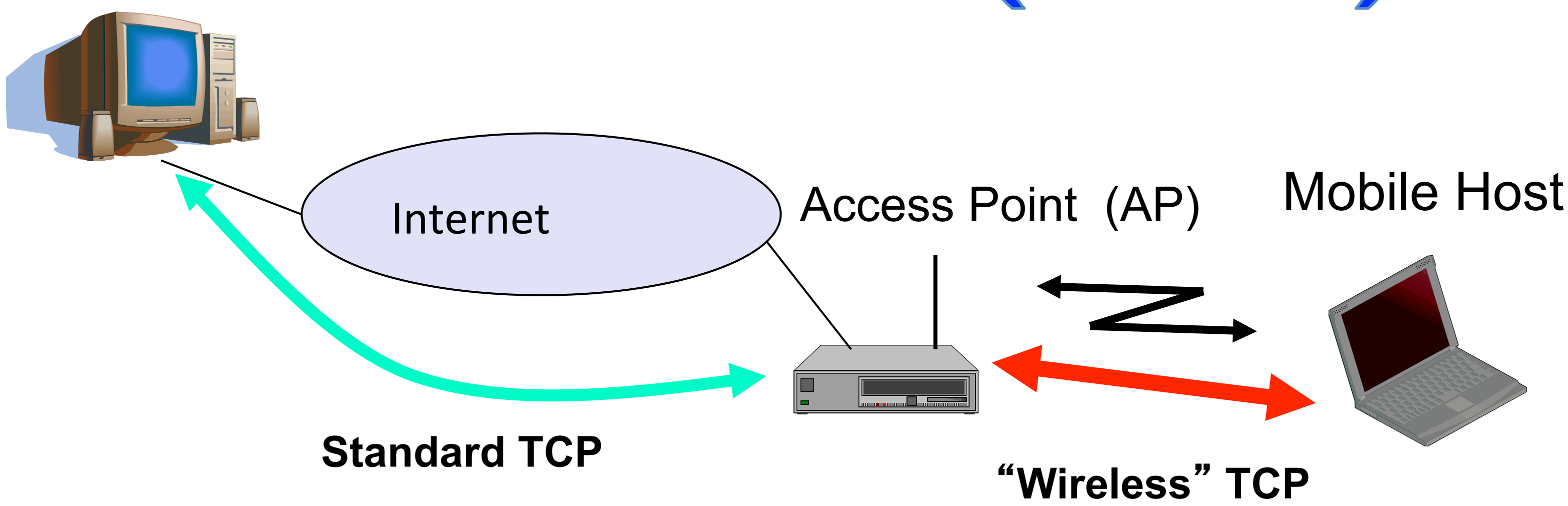
- **Split-connection approaches**
  - Split a TCP connection into two: the *wire-line* part and *wire-less* part at a Base Station or Access Point (Foreign Agent)
- **Link layer approaches**
  - Improve link layer reliability
- **End-to-end approaches**
  - Modify TCP congestion control mechanism
- **Hybrid approaches**

I-TCP, Snooping TCP, and M-TCP

SPLIT-CONNECTION APPROACHES

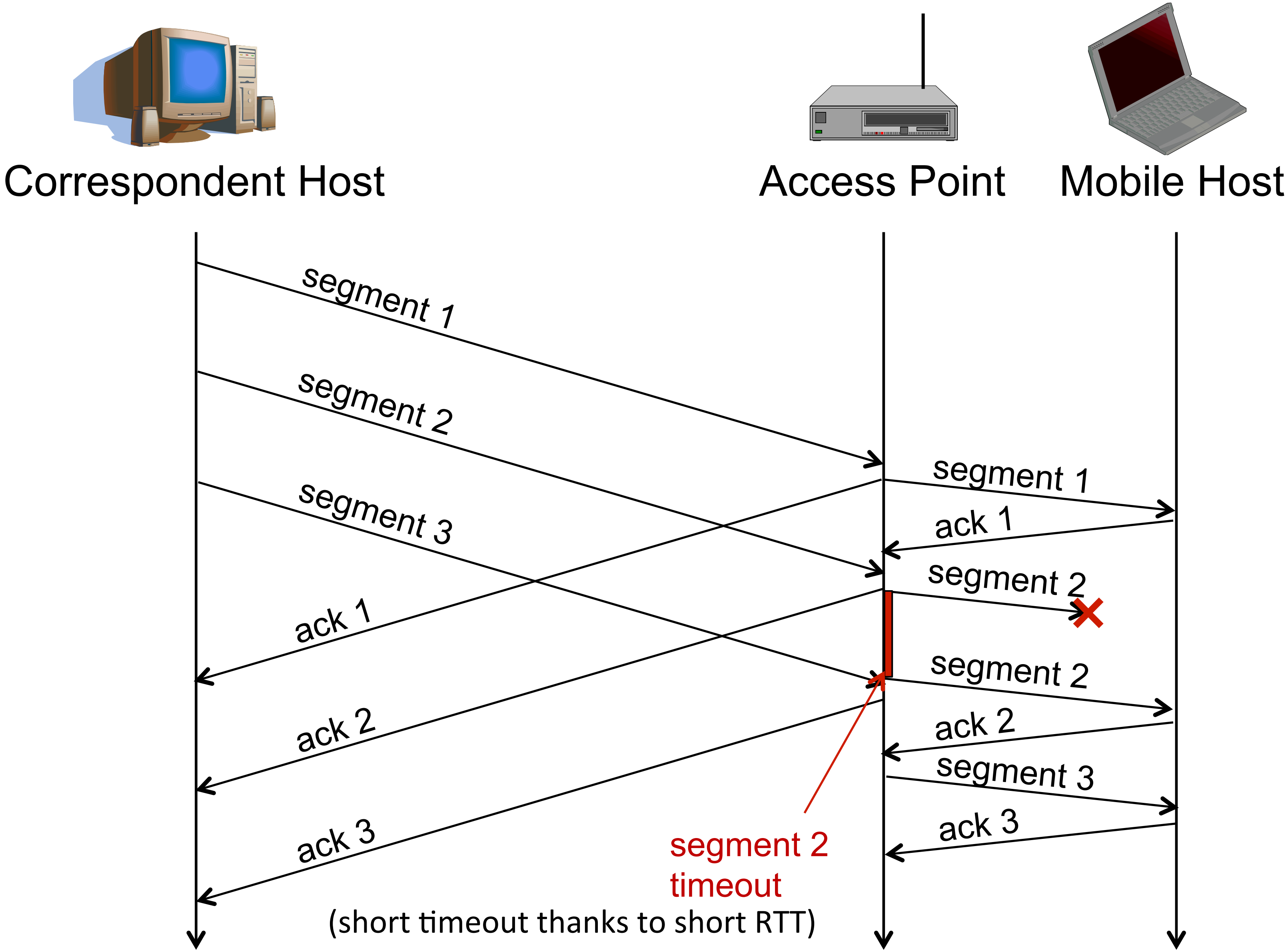
Contents

Indirect TCP (I-TCP)

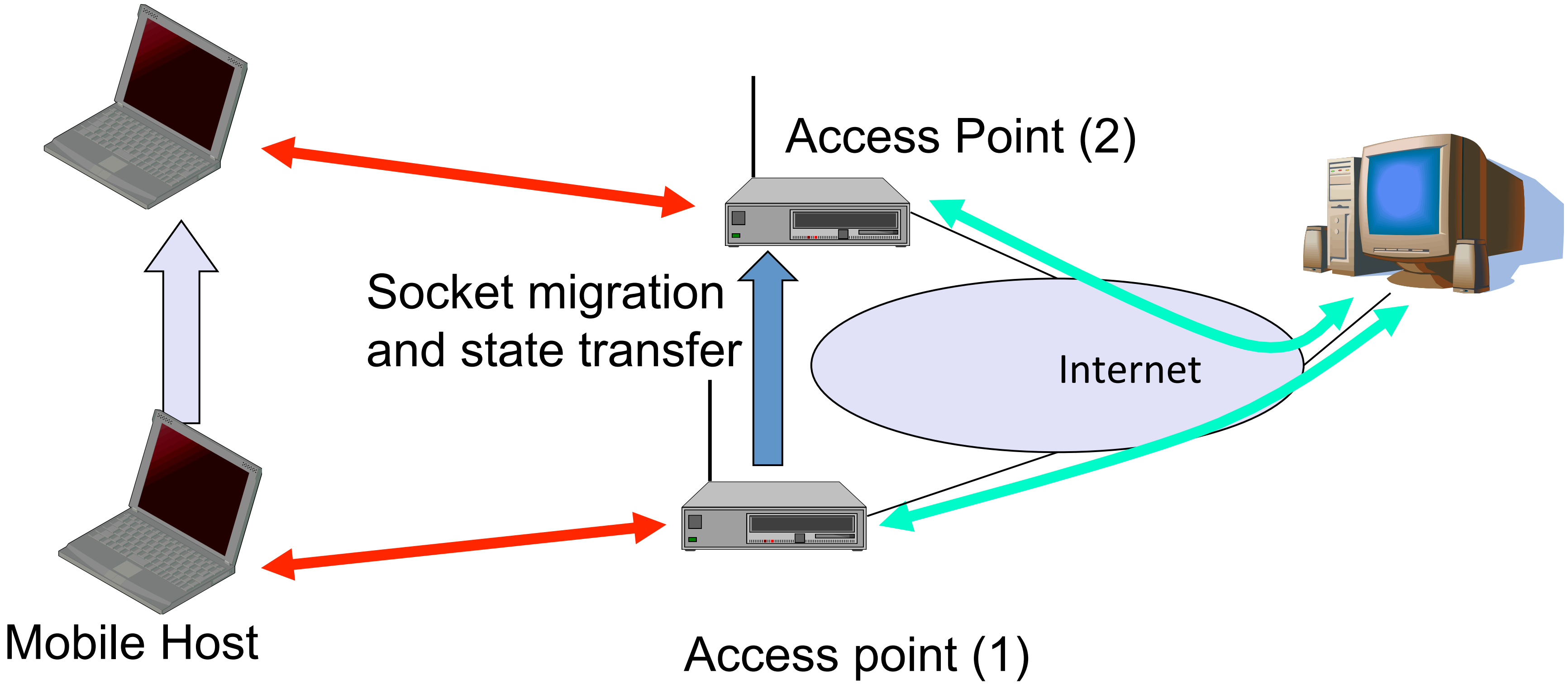


- Split the TCP connection at the AP into two parts
  - AP buffers and retransmits received segments
  - AP sends ACKs for the received segments
- Standard TCP on the wire-line link
- On the wireless link:
  - TCP optimized for wireless
  - Even standard TCP benefits from shorter RTT
    - Shorter timeout
    - Faster retransmissions

I-TCP example



I-TCP and Mobility



- Moving to a new access point requires transfer of socket state
  - Including segments buffered at the FA

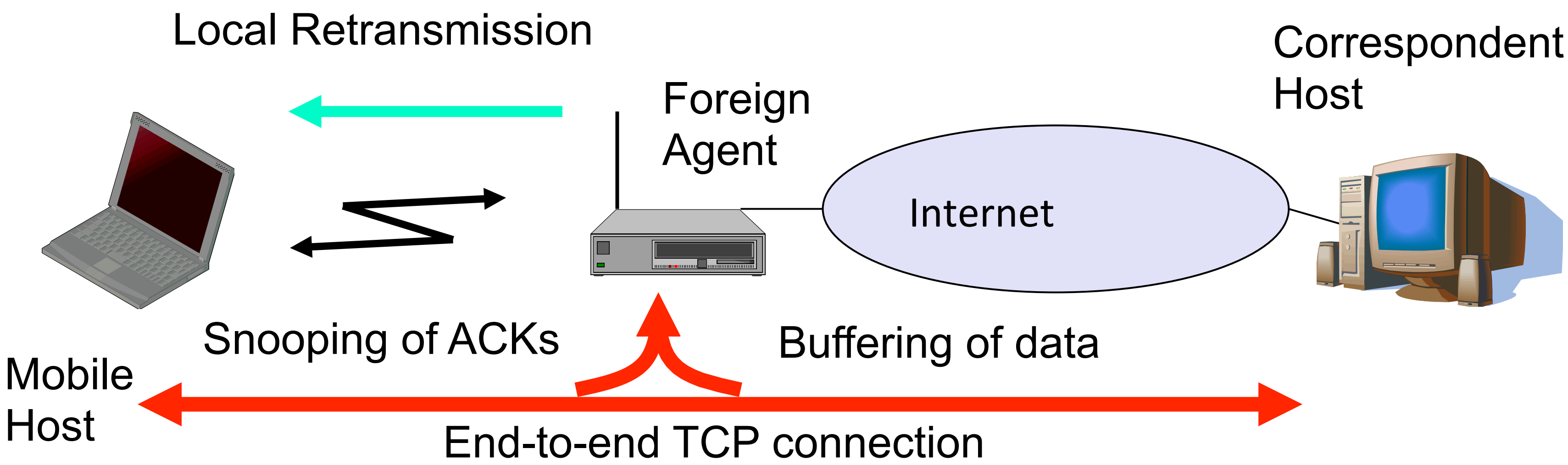
I-TCP summary

- **I-TCP Advantages**
  - No changes in the fixed network or hosts (TCP protocol), all current TCP optimizations still work
    - Potentially no changes in mobile hosts
  - Wireless transmission errors do not "propagate" to the wire-line network

- **I-TCP Disadvantages**
  - Loss of end-to-end semantics
    - An ACK does not imply that the receiver got the segment
    - For mobility support, **all** FAs need to be I-TCP compatible, and the state needs to be transferred to maintain end-to-end semantics
  - Higher end-to-end delays due to buffering and forwarding to a new agent
  - Problem with security mechanisms, e.g. IPsec
    - FA needs to spoof ACKs

Contents

Snooping TCP



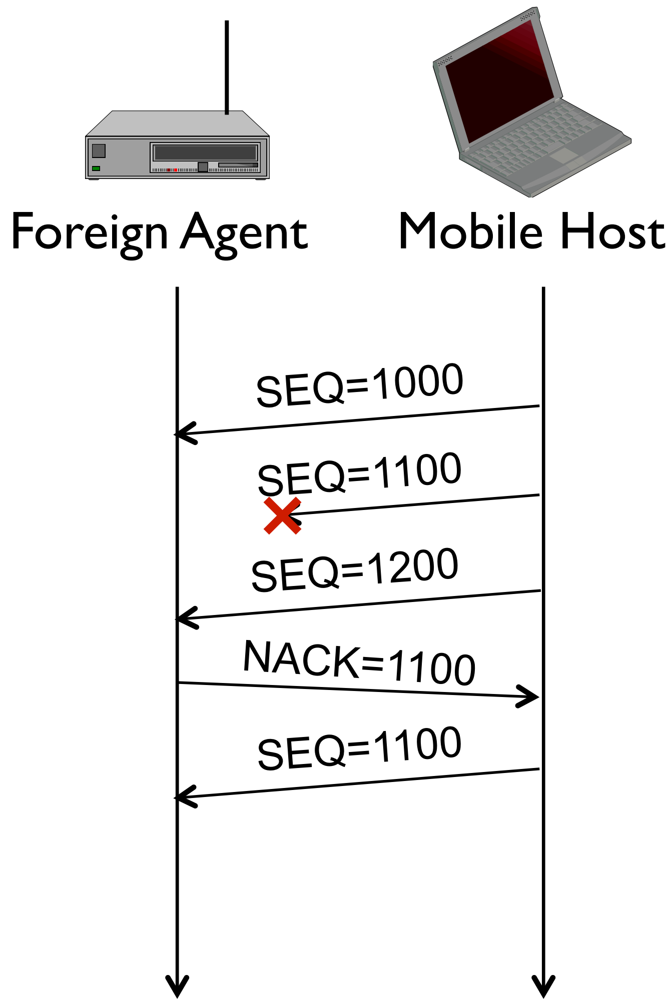
- "TCP-Aware Link Layer"
- Splits connection like I-TCP
  - FA buffers and retransmits segments (if necessary)
  - FA does not ACK buffered packets as I-TCP does (preserves end-to-end semantics)

Snooping TCP: Data Transfer

- Data transfer to the Mobile Host
  - FA buffers a segment until it receives an ACK from the MH
  - FA detects segment loss via duplicate ACKs or a timeout
    - FA timeout is shorter than the round-trip timeout (RTO) at the sender
  - FA locally retransmits lost segments
  - FA drops duplicated ACKs from MH
    - Prevents unnecessary retransmissions and congestion window reductions at the sender
    - Does **not** violate end-to-end semantics:
      Even if the FA crashes before MH receives the segment, the sender will eventually detect the loss via a timeout and retransmit

Snooping TCP: Data Transfer

- Data transfer from the Mobile Host
  - FA detects segment loss on the wireless link via missing sequence numbers
  - FA triggers retransmission of lost segment at MH
    - E.g., with a NACK mechanism
  - MH retransmits data with a much shorter delay



Snooping TCP and Mobility

- When the MH moves to a new FA, should the buffered segments be transferred from the old FA?
  - Not necessary
    - Even if some of the buffered segments are lost in the transition, the sender will eventually timeout and retransmit them
    - This preserves end-to-end semantics
  - Yet, this buffer transfer would improve performance because the timeout puts the sender into slow start phase

Snooping TCP summary

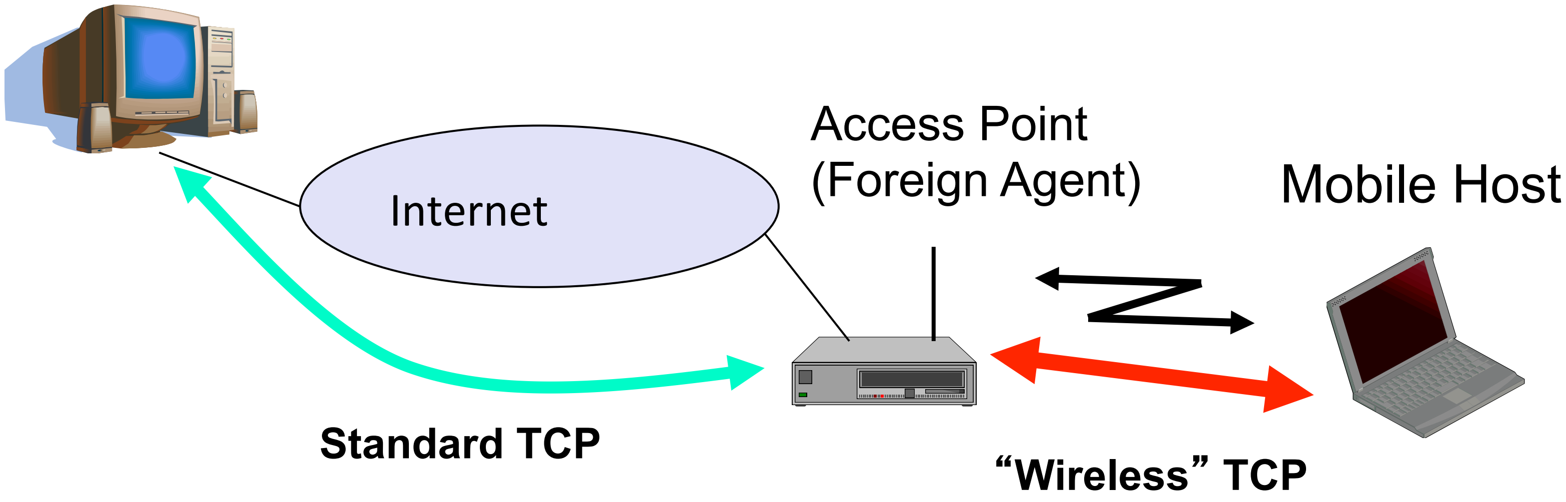
- **Snooping TCP advantages**
  - End-to-end semantics preserved
  - Transfer of buffered segments not necessary during handoff
    - MH can move to FA without Snoop TCP support

- **Snooping TCP disadvantages**
  - Does not isolate wireless link failures as well as I-TCP does
    - If FA takes too long to retransmit a segment, CH will timeout
  - Requires modifications at Mobile Host
    - NACK or similar mechanism to force retransmission
  - Snooping cannot be done if TCP headers are encrypted (like in IPsec EPS; application layer security, e.g. TLS, is compatible)
  - Dropping duplicate ACKs can break integrity

Contents

- Traditional TCP: A Brief Review
- TCP over Wireless: Challenges
- TCP over Wireless: Solutions
  - Indirect TCP
  - Snooping TCP
  - Mobile TCP
  - Fast Retransmit/Fast Recovery
  - Transmission/Time-out Freezing
  - Selective Retransmission
  - Transaction-Oriented TCP
- TCP over 2.5/3G Wireless Networks
- Summary

Mobile TCP (M-TCP)



- Handling of long and frequent disconnections
- Splits connection at FA as I-TCP does
- Foreign agent
  - No caching, no retransmission
  - Monitors all packets
  - If it detects a disconnection (no ACKs from MN for a while)
    - Reports **rwnd** = 0 to sender
    - Sender automatically goes into “persist” mode: Does not timeout or in any other way change the congestion window

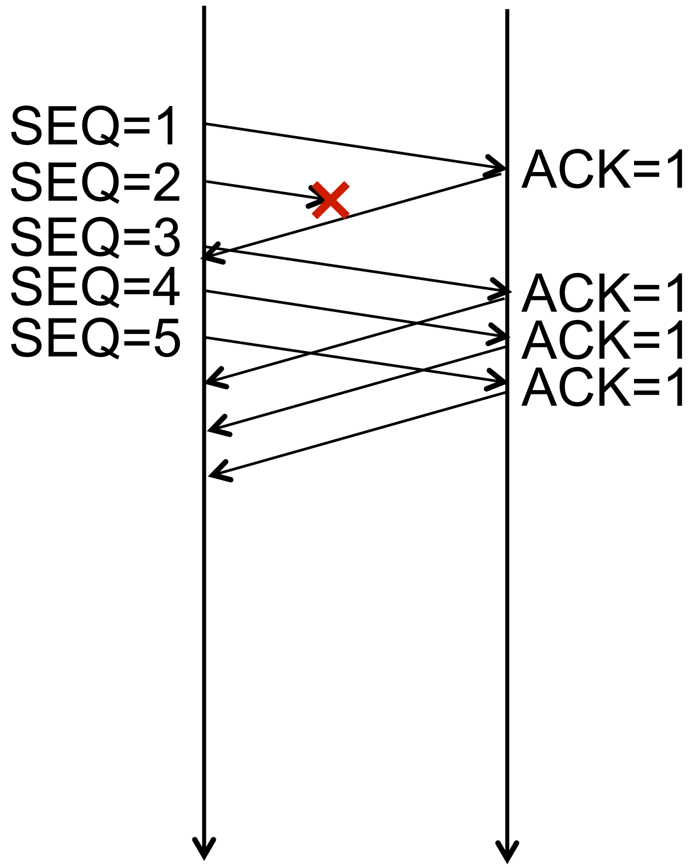
M-TCP summary

- **M-TCP advantages**
  - End-to-end semantics preserved
  - Moving to another FA does not require forwarding buffered packets to new FA (since FA does no buffering)
- **M-TCP disadvantages**
  - Wireless link loss propagates to the wire-line network
  - Packets lost due to link errors need to be retransmitted by the sender
  - Problems with security mechanisms (just like I-TCP)
- ***NOTE: M-TCP handles mobility errors, not transmission errors***

Contents

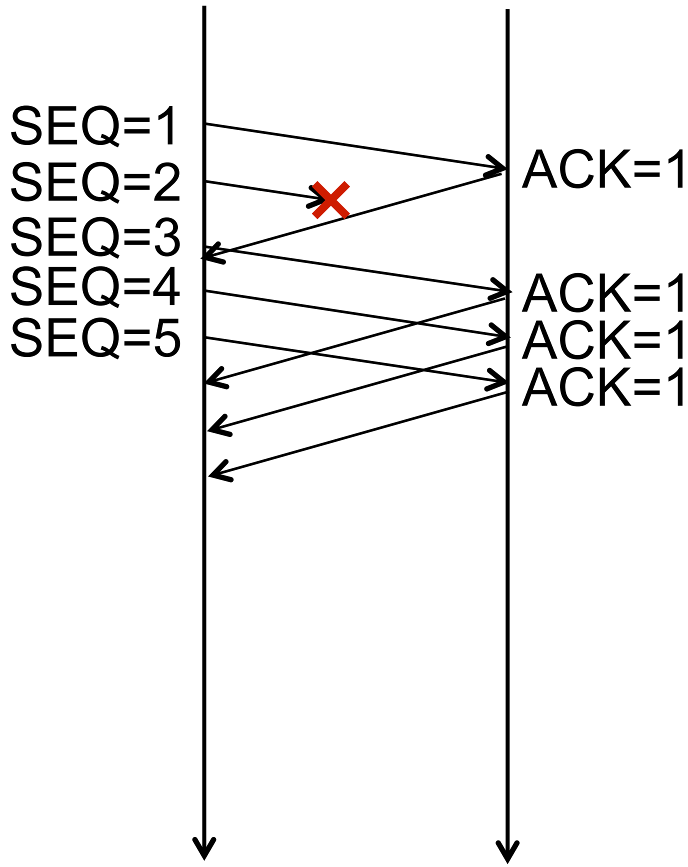
Fast Retransmit and Fast Recovery

- **Assumptions**
  - Congestion causes many segments to be dropped
  - If a single segment is dropped, but the triggered duplicate ACKs are delivered, the network is probably not congested
- **Hence**
  - No need for drastic reduction of congestion window (as in TCP Tahoe)
  - Fast Recovery phase



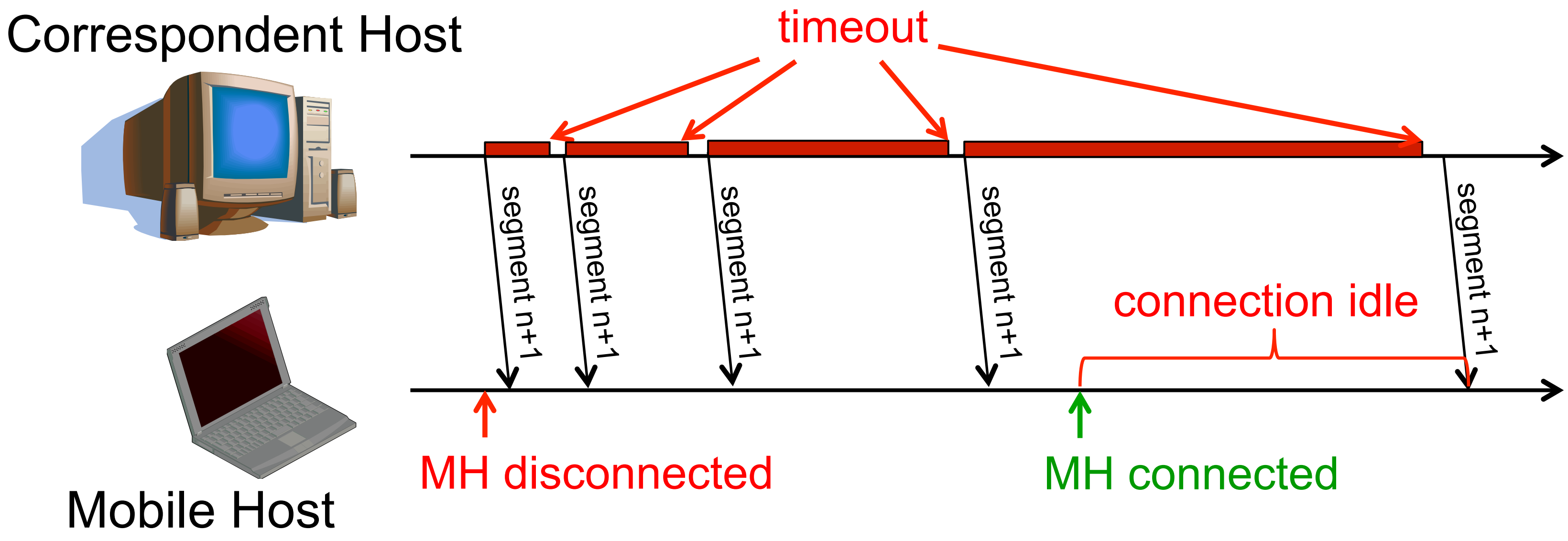
Fast Retransmit and Fast Recovery

- **How it works?**
  - As soon as the mobile host registers at a new foreign agent using mobile IP, **it starts sending duplicated acknowledgements to correspondent hosts** (three duplicates)
  - This forces the corresponding host to go into fast retransmit mode and not to start slow start, i.e., the correspondent host continues to send with the same rate it did before the mobile host moved to another foreign agent



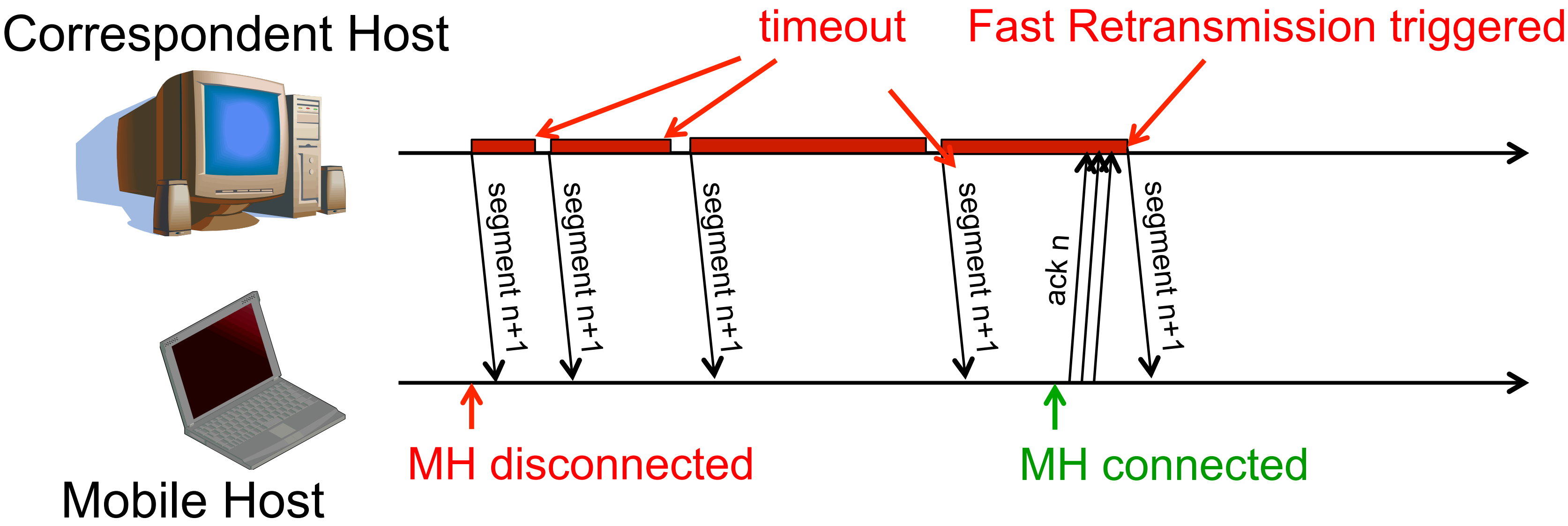
Contents

Long Disconnections



- TCP doubles RTO (Retransmit Time Out) each time a timeout occurs
  - Can cause unnecessary idle time after longer disconnection

Long Disconnections



- TCP doubles RTO each time a timeout occurs
  - Can cause unnecessary idle time after longer disconnection
- A Mobile Host aware of connection state can "wake up" the CH
  - Trigger fast retransmit with duplicate ACKs
- Simple solution
  - No changes at CH necessary
  - But TCP at MH needs to be aware of connectivity

Contents

Selective Acknowledgment (SACK)

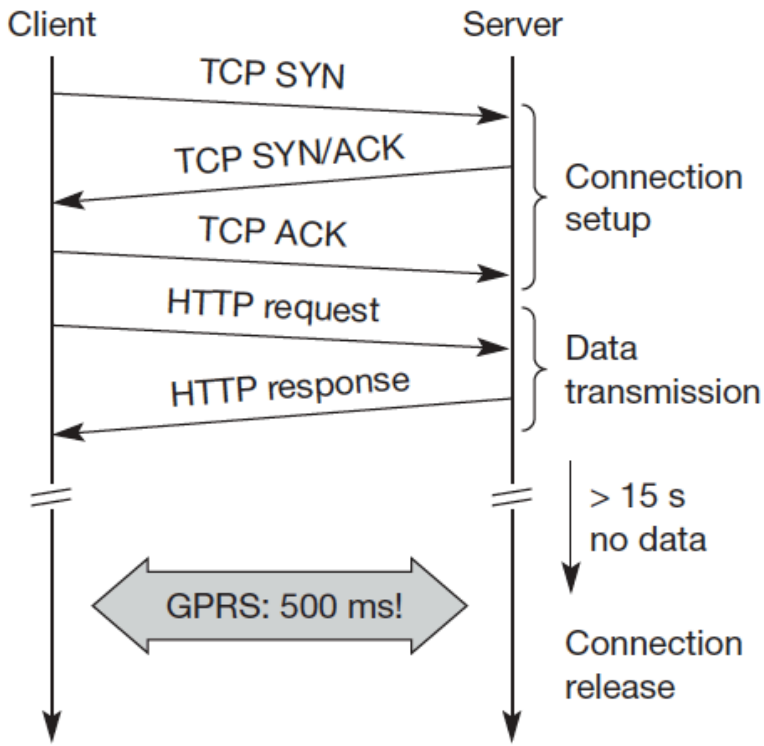
- TCP acknowledgements are cumulative
  - ACK n acknowledges correct and in-sequence receipt of packets up to n
  - The sender only learns about the first lost segment
    - What if more segments are lost?
- Selective retransmission as one solution
  - **RFC2018** allows for **acknowledgments of single packets**, not only acknowledgments of in-sequence packet streams without gaps
  - Sender can retransmit missing packets more efficiently
- **Advantage**
  - Higher efficiency
- **Disadvantage**
  - More complex software in the receiver, more buffer needed at the receiver (CPU-, memory- and power-constraint MH)

Contents

Transaction Oriented TCP (T-TCP)

- **TCP phases**
  - Connection setup, data transmission, connection release
  - Using 3-way-handshake needs 3 packets for setup and release, respectively
  - Thus, even short messages need a minimum of 7 packets!
- **Transaction oriented TCP**
  - RFC1644 describes a TCP version to avoid this overhead
  - Connection setup, data transfer and connection release can be combined
  - Thus, only 2 or 3 packets are needed

Transaction Oriented TCP (T-TCP)



- **Advantage**
  - Efficiency for single packet transaction
- **Disadvantage**
  - Requires TCP modifications at all hosts

Contents

- Traditional TCP: A Brief Review
- TCP over Wireless: Challenges
- TCP over Wireless: Solutions
  - Indirect TCP
  - Snooping TCP
  - Mobile TCP
  - Fast Retransmit/Fast Recovery
  - Transmission/Time-out Freezing
  - Selective Retransmission
  - Transaction-Oriented TCP
- TCP over 2.5/3G Wireless Networks
- Summary

TCP over 2.5/3G wireless networks: Characteristics to be Considered

- Data Rate Asymmetry
  - May reach a factor of 1000 in uplink and downlink
- Latency
  - Using error correction algorithms (e.g., FEC) makes hundred milliseconds of latency
- Jitter
  - Delay spikes in wireless systems
- Packet Loss
  - Packet loss in handover or corruptions

TCP over 2.5/3G wireless networks: Possible Solutions

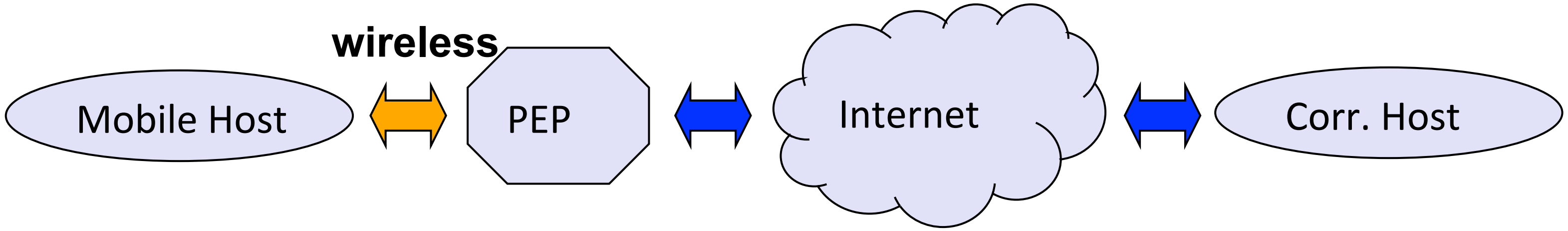
- Large Windows
- Limited Transmit
- Large MTU
- Selective Acknowledgment (SACK)
- Explicit Congestion Notification (ECN)
- Timestamp
- No Header Compression

Contents

Summary of Classical Enhancements to TCP for Mobility

Approach	Mechanism	Advantages	Disadvantages
Indirect TCP	Splits TCP connection into two connections	Isolation of wireless link, simple	Loss of TCP semantics, higher latency at handover, security problems
Snooping TCP	Snoops data and acknowledgements, local retransmission	Transparent for end-to-end connection, MAC integration possible	Insufficient isolation of wireless link, security problems
M-TCP	Splits TCP connection, chokes sender via window size	Maintains end-to-end semantics, handles long term and frequent disconnections	Bad isolation of wireless link, processing overhead due to bandwidth management, security problems
Fast retransmit/ fast recovery	Avoids slow-start after roaming	Simple and efficient	Mixed layers, not transparent
Transmission/ time-out freezing	Freezes TCP state at disconnection, resumes after reconnection	Independent of content, works for longer interruptions	Changes in TCP required, MAC dependent
Selective retransmission	Retransmits only lost data	Very efficient	Slightly more complex receiver software, more buffer space needed
Transaction-oriented TCP	Combines connection setup/release and data transmission	Efficient for certain applications	Changes in TCP required, not transparent, security problems

Performance enhancing proxies (PEP)



- Transport layer proxies
  - Local retransmissions and acknowledgements
  - Any of the approaches reviewed above qualifies
- Application layer proxies
  - HTTP, FTP, …
  - Content caching, filtering, compression, picture downscaling
- Big problem: breaks security end-to-end semantics
  - Disables use of IP security
  - Choose between PEP and security!

Deployment in practice

- [Wei PAM06] reports that split-connection (similar to I-TCP) and application layer proxies are used by US cellular operators (GPRS and CDMA2000 networks)
- Deployed mechanism depends on traffic

	Operator 1	Operator 2	Operator 3
FTP	Split-TCP	Split-TCP	No
HTTP data	Both	Split-TCP	Split-TCP
HTTP image	Proxy	Proxy	Proxy
plain TCP	No	No	Split-TCP (option)

W. Wei, C. Zhang, H. Zang, J. Kurose, and D. Towsley.
Inference and Evaluation of Split-Connection Approaches in Cellular Data Networks.
PAM 2006