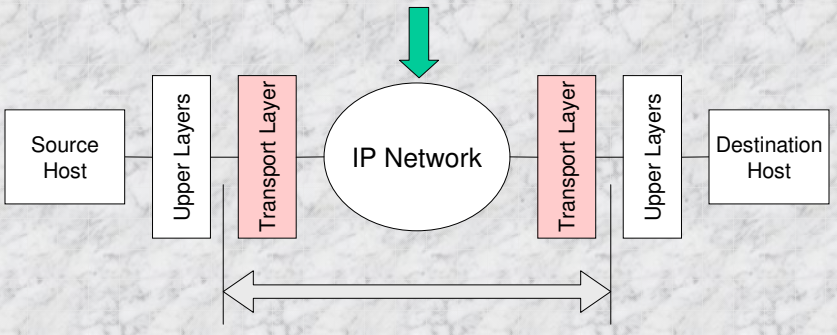
Course Overview

- Introduction
- Data in Wireless Cellular Systems
- Data in Wireless Local Area Networks
- Internet Protocols
- TCP over Wireless Link
- Ad-Hoc Networks, Sensor Networks
- Services and Service Discovery
- System Support for Mobile Applications

Transport Protocol

- What is the role of the "Transport Layer" ?

The IP Network DOES NOT guarantee delivery !!



The transport layer provides more reliable delivery

TCP and Mobile Computing

- TCP is (most?) popular transport layer protocol
- designed for wired networks
  - low error rate
  - requirement to share bottlenecks
- key assumptions in TCP are:
  - packet loss is indication of congestion, not transmission error
  - rather aggressive response to congestion is needed to ensure fairness and efficiency
- wireless links and mobile computing violate these assumptions:
  - packets lost due to unreliable physical media
  - packets can get lost due to handover

TCP and Mobile Computing

- packet losses over wireless link often in bursts
  - will trigger slow start rather than fast retransmit
- packet loss no indication of congestion
  - reduction of congestion window will reduce throughput
  - getting back to previous window size may take long
- problem caused by mismatch of wireless link properties with assumptions underlying TCP design
- multiple suggestions to improve TCP performance:
  - link-level retransmissions: improve reliability of wireless link
  - network layer solutions: SNOOP
  - transport layer solutions: I-TCP (indirect TCP), Mowgli
  - session layer solutions: establish end-to-end session layer connection, manages two separate TCP connections

Link Layer Mechanisms
Forward Error Correction

- Forward Error Correction (FEC) can be use to correct small number of errors
- Correctable errors hidden from the TCP sender
- FEC incurs overhead even when errors do not occur
  - Adaptive FEC schemes can reduce the overhead by choosing appropriate FEC dynamically
- FEC does not guard/protect from packet loss due to handover

Link Layer Mechanisms
Link Level Retransmissions

- Link level retransmission schemes retransmit a packet at the link layer, if errors are detected
- Retransmission overhead incurred only if errors occur
  - unlike FEC overhead

In general

- Use FEC to correct a small number of errors
- Use link level retransmission when FEC capability is exceeded

Link Level Retransmissions An Early Study

- The sender's Retransmission Timeout (RTO) is a function of measured RTT (round-trip times)
  - **Link level retransmits increase RTT, therefore, RTO**
- **If errors not frequent**, RTO will **not** account for RTT variations due to link level retransmissions
  - When errors occur, the sender may timeout & retransmit before link level retransmission is successful
  - Sender and link layer both retransmit
  - Duplicate retransmissions (interference) waste wireless bandwidth
  - Timeouts also result in reduced congestion window

A More Accurate Picture

- Early analysis does not accurately model real TCP stacks
- With large **RTO granularity**, interference is unlikely, if time required for link-level retransmission is small compared to TCP RTO
  - Standard TCP RTO granularity is often large (500 ms)
  - Minimum RTO (2*granularity) is large enough to allow a small number of link level retransmissions, if link level RTT is relatively small
  - Interference due to timeout not a significant issue when wireless RTT small, and RTO granularity large

Link Level Retransmissions A More Accurate Picture

- **Frequent errors** increase RTO significantly on slow wireless links
  - RTT on slow links large, retransmissions result in large variance, pushing RTO up
  - Likelihood of interference between link layer and TCP retransmissions smaller
  - But congestion response will be delayed due to larger RTO
  - When wireless losses do cause timeout, much time wasted

Link Layer Schemes: Summary

When is a reliable link layer beneficial to TCP performance?

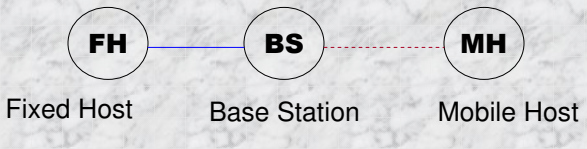
- if it provides *almost in-order* delivery
- TCP retransmission timeout large enough to tolerate additional delays due to link level retransmits
- Basic ideas:
  - Hide wireless losses from TCP sender
  - Link layer modifications needed at both ends of wireless link
    - TCP need not be modified

Split Connection Approach

- End-to-end TCP connection is broken into one connection on the wired part of route and one over wireless part of the route
- A single TCP connection split into two TCP connections
  - if wireless link is not last on route, then more than two TCP connections may be needed

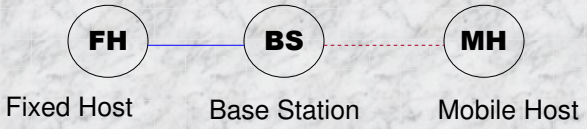
Split Connection Approach

- Connection between wireless host MH and fixed host FH goes through base station BS
- FH-MH = FH-BS + BS-MH



Split Connection Approach

- Split connection results in independent flow control for the two parts
- Flow/error control protocols, packet size, time-outs, may be different for each part



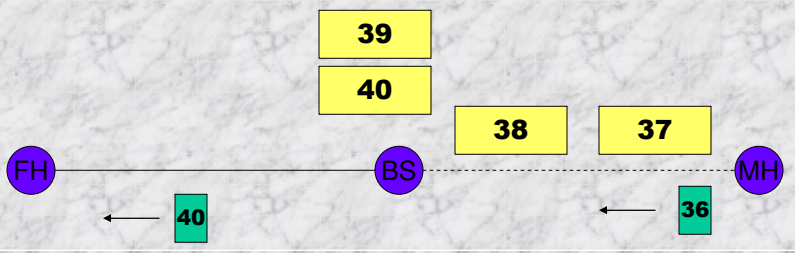
Fixed Host Base Station Mobile Host

Split Connection Approach: Advantages

- BS-MH connection can be *optimized* independent of FH-BS connection
  - Different flow / error control on the two connections
- Local recovery of errors
  - Faster recovery due to relatively shorter RTT on wireless link
- Good performance achievable using **appropriate** BS-MH protocol
  - Standard TCP on BS-MH performs poorly when multiple packet losses occur per window (timeouts can occur on the BS-MH connection, stalling during the timeout interval)
  - **Selective acks improve performance for such cases**

Split Connection Approach: Disadvantages

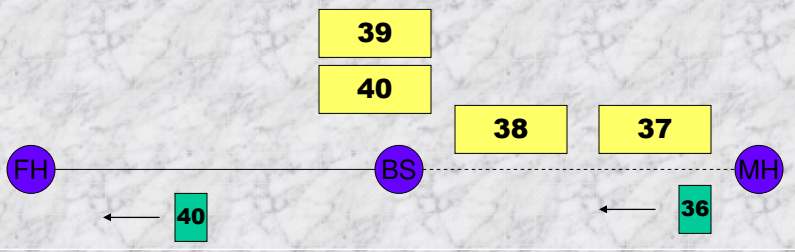
- End-to-end semantics violated
  - ack may be delivered to sender, before data delivered to the receiver
  - May not be a problem for applications that do not rely on TCP for the end-to-end semantics



Split Connection Approach: Disadvantages

- BS retains hard state

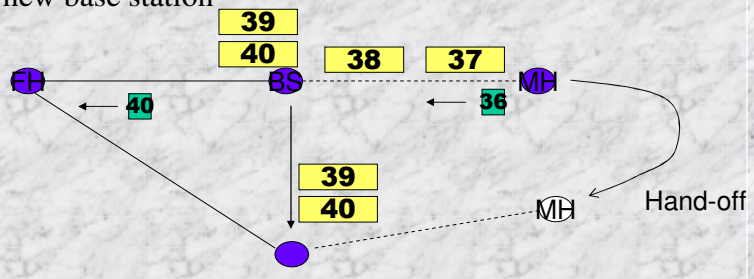
  BS failure can result in loss of data (unreliability)
  - If BS fails, packet 40 will be lost
  - Because it is ack'd to sender, the sender does not buffer 40



Split Connection Approach: Disadvantages

- BS retains hard state

  Hand-off latency increases due to state transfer
  - Data that has been ack'd to sender, must be moved to new base station

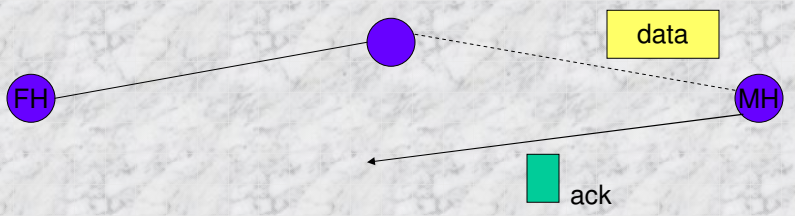


Split Connection Approach: Disadvantages

- Buffer space needed at BS for each TCP connection
  - BS buffers tend to get full, when wireless link slower (one window worth of data on wired connection could be stored at the base station, for each split connection)
- Window on BS-MH connection reduced in response to errors
  - may not be an issue for wireless links with small delay-bw product

Split Connection Approach: Disadvantages

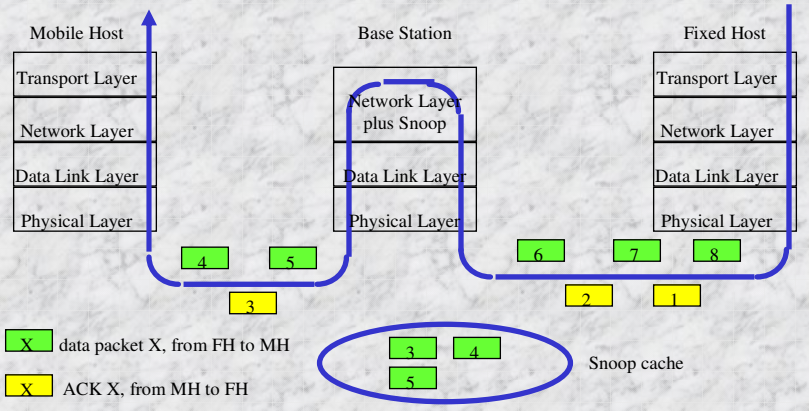
- Extra copying of data at BS
  - copying from FH-BS socket buffer to BS-MH socket buffer
  - increases end-to-end latency
- May not be useful if data and acks traverse different paths (both do not go through the base station)
  - Example: data on a satellite wireless hop, acks on a dial-up channel



Snoop: Network Layer Solution

- idea: modify network layer software at base station
- changes are transparent to MH and FH
  - no changes in TCP semantics (unlike I-TCP)
  - less software overhead (packets pass TCP layer only twice)
  - no application relinking on mobile host
- modifications are mostly in caching packets and performing local retransmissions across the wireless link by monitoring (*snooping*) on TCP acks
- results are impressive:
  - speedups of up to 20 times over regular TCP
  - more robustness when dealing with multiple packet losses

Snoop: Architecture



Snoop: Description of Protocol

- processing packets from FH
  - new packet in the normal TCP sequence:
    - cache and forward to MH
  - packet out-of sequence and cached earlier:
    - sequence number > last ack from MH: packet probably lost, forward it again
    - otherwise, packet already received at MH, so drop
      - but: original ACK could have been lost, so fake ACK again
  - packet out-of sequence and not cached yet:
    - packet either lost earlier due to congestion or delivered out-of-order: cache packet and mark as retransmitted, forward to MH

Snoop: Description of Protocol

- processing ACKs from MH:
  - new ACK: common case, initiates cleaning up of snoop cache, update estimate of round-trip time for wireless link, forward ACK to FH
  - spurious ACK: less than last ACK seen, happens rarely. Just drop ACK and continue
  - duplicate ACK: indicates packet loss, one of several actions:
    - packet either not in cache or marked as retransmitted: pass duplicate ACK on to FH
    - first duplicated ACK for cached packet: retransmit cached packet immediately and at high priority, estimate number of expected duplicate ACKs, based on # of packets sent after missing one
    - expected successive duplicate ACKs: ignore, we already initiated retransmission. Since retransmission happens at higher priority, we might not see total number of expected duplicate ACKs

Snoop: Description of Protocol

- design does not cache packets from MH to FH
  - bulk of packet losses will be between MH and base
  - but snooping on packets generates requests for retransmissions at base much faster than from remote FH
  - enhance TCP implementation at MH with “selective ACK” option:
    - base keeps track of packets lost in a transmission window
    - sends bit vector back to MH to trigger retransmission of lost packets
- mobility handling:
  - when handoff is requested by MH or anticipated by base station, nearby base stations begin receiving packets destined for MH, priming their cache
  - caches synchronized during actual handoff (since nearby bases cannot snoop on ACKs)

Snoop: Performance

- no difference in very low error rate environment (bit error rate < $5x10^{-7}$)
- for higher bit error rates, Snoop outperforms regular TCP by a factor of 1 to 20, depending on the bit error rate (the higher, the better Snoop's relative performance)
- even when every other packet was dropped over the wireless link, Snoop still allowed for progress in transmission, while regular TCP came to a grinding halt
- Snoop provides high and consistent throughput, regular TCP triggers congestion control often, which leads to periods of no transmission and very uneven rate of progress

Snoop: Evaluation

- most effort spent on direction FH->MH
  - authors argue that not much can be done for MH->FH
    - losses occur over first link, the unreliable wireless link
- Internet drops 2%-5% of IP packets, tendency rising
  - assume that IP packet is lost in wired part of network:
    - receiver (FH) will issue duplicate ACKs
    - this should trigger fast retransmit rather than slow start (?)
    - nothing is done to ensure that ACKs are not dropped over last link
    - retransmission of data packet over wireless link is subject to unreliable link and low bandwidth again
  - Snoop could potentially benefit from caching packets in both directions
    - how would this differ from link-layer retransmission policy?

TCP over Wireless: Summary

- Discussed only a few ideas, for a more complete discussion, see Tutorial on TCP for Wireless and Mobile Hosts by Nitin Vaidya, http://www.cs.tamu.edu/faculty/vaidya/presentations.html
- Topics ignored:
  - asymmetric bandwidth on uplink and downlink (for example in some cable or satellite networks)
  - wireless link extends over multiple hops, such as in an ad-hoc network
  - connections fail due to spurious disconnections or route failures in ad-hoc networks
- Many proposals focus on downlink only
- Many proposals, most try to avoid changing TCP interface or semantics, but more work necessary