Course Overview

- Introduction and History
- Regulatory Issues
- Fundamentals of Cellular Systems
- 1st Generation: AMPS and CDPD
- 2nd Generation: GSM
- 3rd Generation: IMT-2000
- Mobile IP
- TCP over Wireless Link

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: Basics

- description based on BSD 4.3 “Reno” implementation
- loss detection:
  - timeouts, timeout value calculated by exponentially weighted mean round-trip acknowledgment delay plus a multiple of the deviation of this round-trip acknowledgment delay
    - RTO is kept using coarse-grained timer with 500 millisecond resolution
  - auxiliary mechanism: repeat acknowledgments
    - fast retransmission: retransmit segment when receiving 3 consecutive acknowledgments
  - detection of packet loss triggers congestion control:
    - shrinking the congestion window
    - expanding RTO

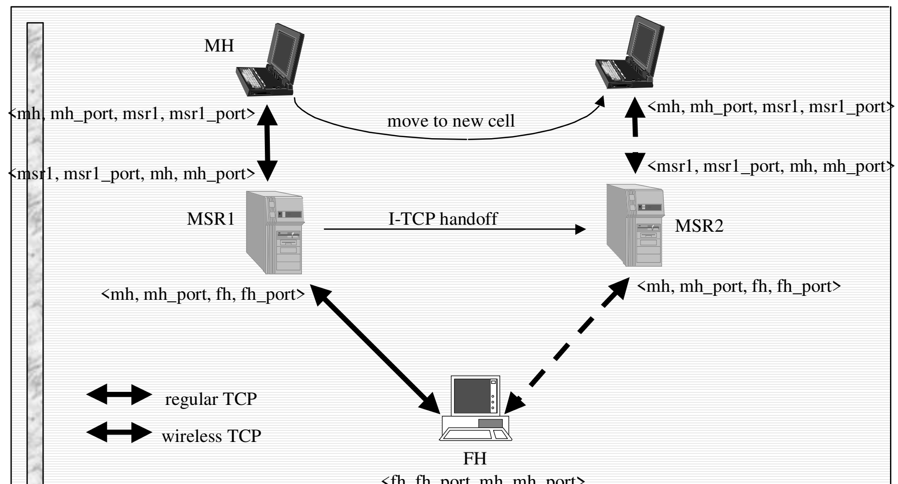
TCP and Mobile Computing

- 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: interact in “funny” ways with TCP retransmissions, can actually degrade TCP performance
  - 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

I-TCP

- basic idea: split communication between mobile host (MH) and fixed host (FH) into two separate interactions
  - one between MH and mobile support router (MSR), e.g., base station
  - one between MSR and FH
- each connection can be tuned to accommodate the special characteristics of the underlying physical media
  - use standard TCP between MSR and FH, both on wired backbone
  - special wireless TCP between MH and MSR, where packet loss does not trigger congestion avoidance

I-TCP: Connection Setup



I-TCP

– throughput improved, particularly for wide-area connections, compared to regular TCP

Connection Type	No moves	Overlapped cells	Disjoint cells, 0 sec between	Disjoint cells, 1 sec between
Regular TCP	65.49 kB/s	62.59 kB/s	38.66 kB/s	23.73 kB/s
I-TCP	70.06 kB/s	65.37 kB/s	44.83 kB/s	36.31 kB/s

I-TCP performance over local area

Connection Type	No moves	Overlapped cells	Disjoint cells, 0 sec between	Disjoint cells, 1 sec between
Regular TCP	13.35 kB/s	13.26 kB/s	8.89 kB/s	5.19 kB/s
I-TCP	26.78 kB/s	27.97 kB/s	19.12 kB/s	16.01 kB/s

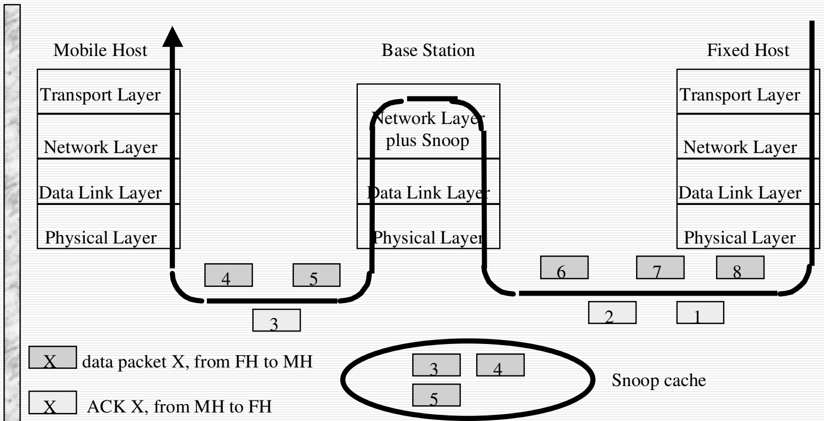
I-TCP performance over wide area

(from: Bakre and Badrinath, "I-TCP: Indirect TCP for Mobile Hosts", Proceedings of the International Conference on Distributed Computing Systems (ICDCS 15), May 1995, Vancouver, Canada, pages 136-143)

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

- 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