

**CSE 4215/5431:**  
**Mobile Communications**  
Winter 2013

**Suprakash Datta**

[datta@cse.yorku.ca](mailto:datta@cse.yorku.ca)

Office: CSEB 3043

Phone: 416-736-2100 ext 77875

Course page: <http://www.cse.yorku.ca/course/4215>

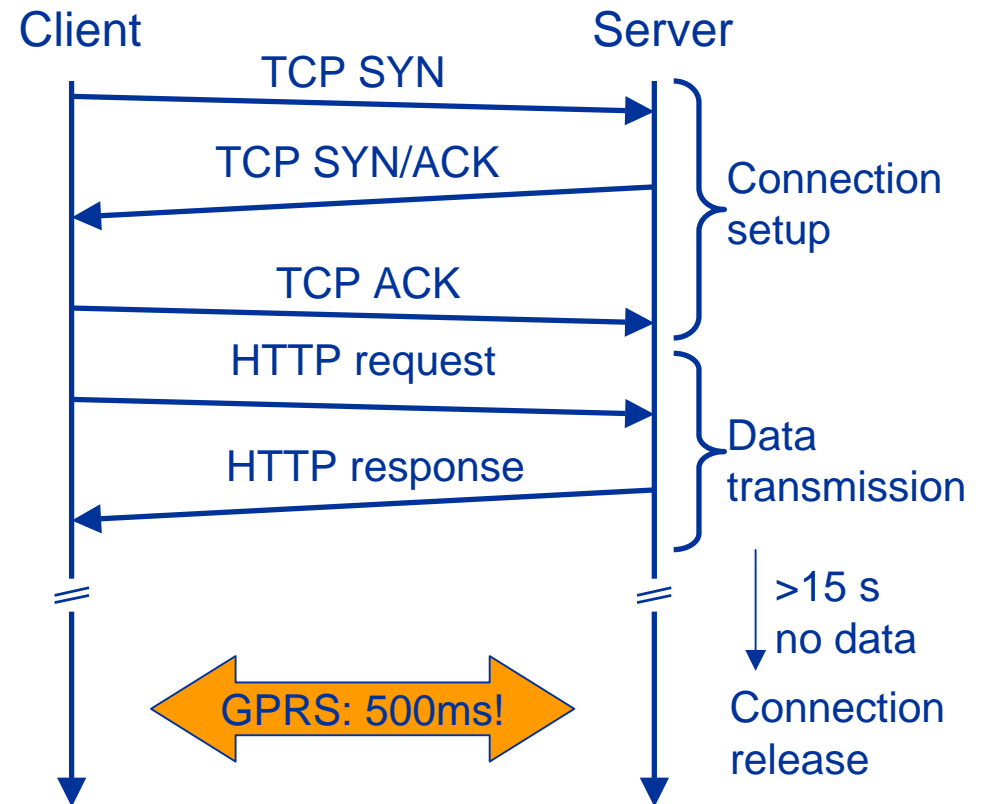
Some slides are adapted from the Schiller book website

# Next

- Transport layer for mobile networks

# Transport Layer

- E.g. HTTP (used by web services) typically uses TCP
  - Reliable transport between client and server required
- TCP
  - Stream oriented, not transaction oriented
  - Network friendly: time-out
    - ➔ congestion
    - ➔ slow down transmission
- Well known – TCP guesses quite often wrong in wireless and mobile networks
  - Packet loss due to transmission errors
  - Packet loss due to change of network
- Result
  - Severe performance drop



# Motivation I

- Transport protocols typically designed for
  - Fixed end-systems
  - Fixed, wired networks
- Research activities
  - Performance
  - Congestion control
  - Efficient retransmissions
- TCP congestion control
  - packet loss in fixed networks typically due to (temporary) overload situations
  - router have to discard packets as soon as the buffers are full
  - TCP recognizes congestion only indirect via missing acknowledgements, retransmissions unwise, they would only contribute to the congestion and make it even worse
  - slow-start algorithm as reaction

# Motivation II

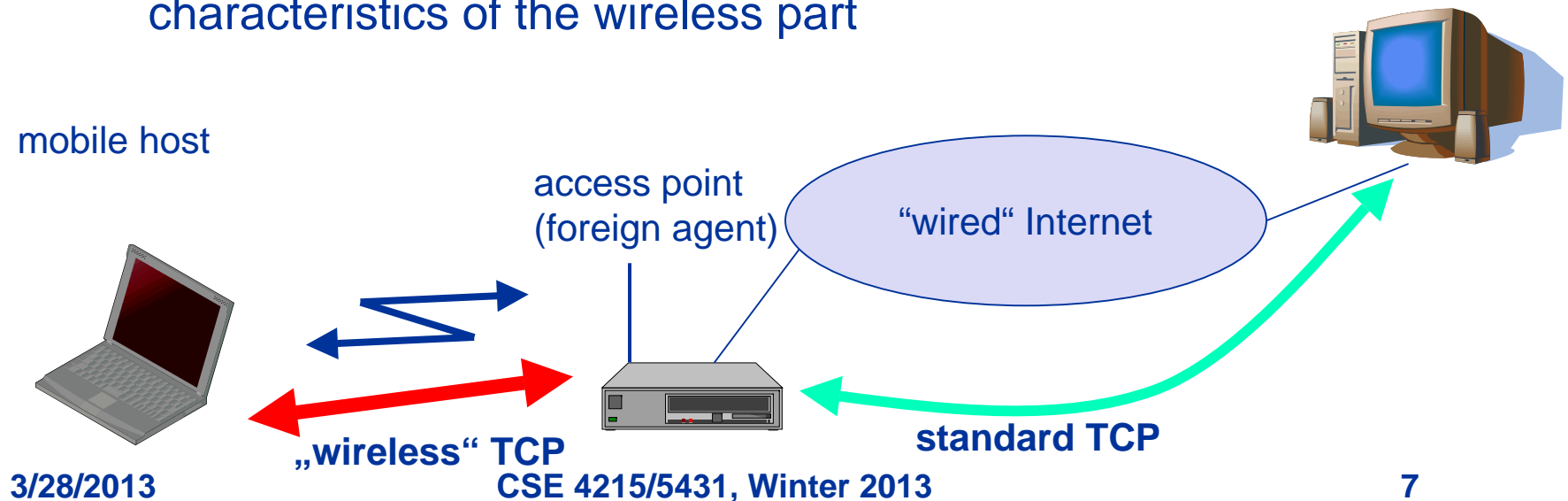
- TCP slow-start algorithm
  - sender calculates a congestion window for a receiver
  - start with a congestion window size equal to one segment
  - exponential increase of the congestion window up to the congestion threshold, then linear increase
  - missing acknowledgement causes the reduction of the congestion threshold to one half of the current congestion window
  - congestion window starts again with one segment
- TCP fast retransmit/fast recovery
  - TCP sends an acknowledgement only after receiving a packet
  - if a sender receives several acknowledgements for the same packet, this is due to a gap in received packets at the receiver
  - however, the receiver got all packets up to the gap and is actually receiving packets
  - therefore, packet loss is not due to congestion, continue with current congestion window (do not use slow-start)

# Influences of mobility on TCP-mechanisms

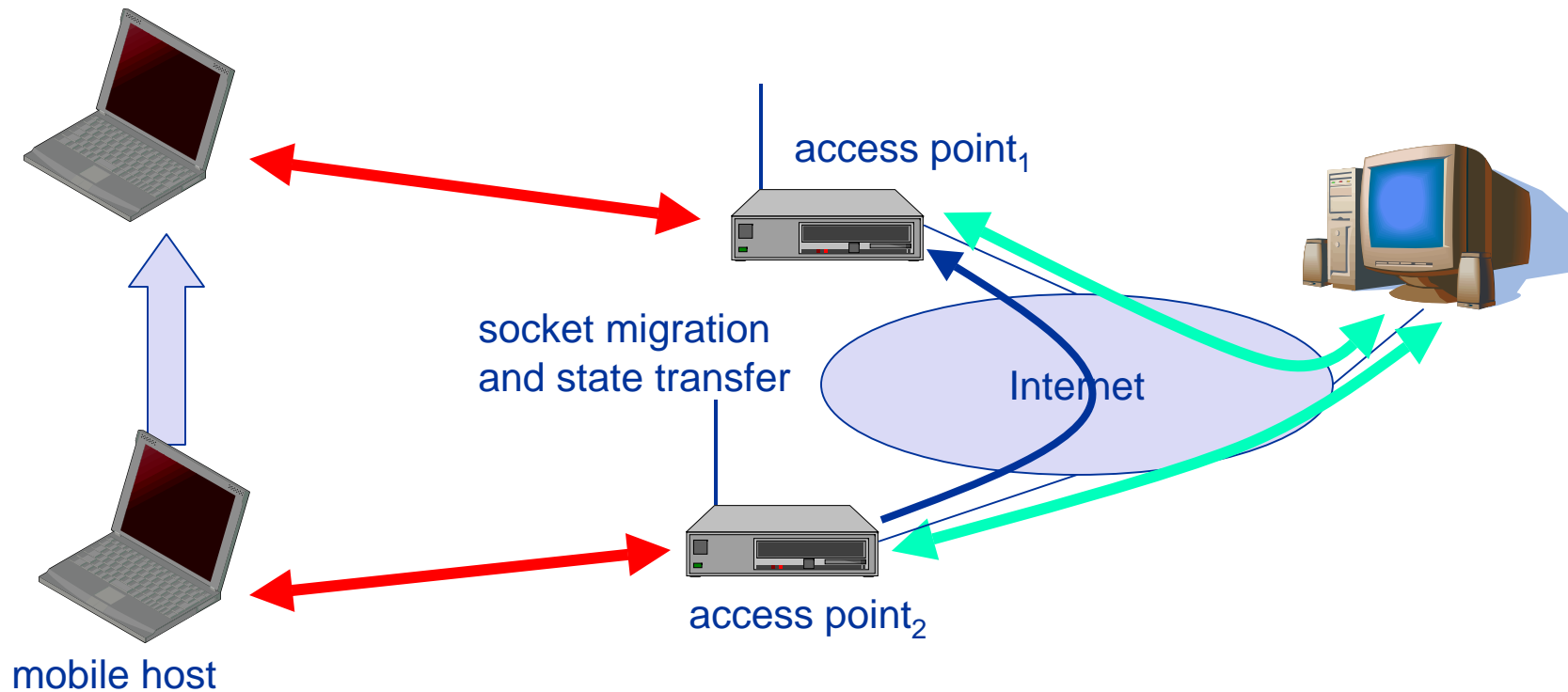
- TCP assumes congestion if packets are dropped
  - typically wrong in wireless networks, here we often have packet loss due to *transmission errors*
  - furthermore, *mobility* itself can cause packet loss, if e.g. a mobile node roams from one access point (e.g. foreign agent in Mobile IP) to another while there are still packets in transit to the wrong access point and forwarding is not possible
- The performance of an unchanged TCP degrades severely
  - however, TCP cannot be changed fundamentally due to the large base of installation in the fixed network, TCP for mobility has to remain compatible
  - the basic TCP mechanisms keep the whole Internet together

# Early approach: Indirect TCP I

- Indirect TCP or I-TCP segments the connection
  - no changes to the TCP protocol for hosts connected to the wired Internet, millions of computers use (variants of) this protocol
  - optimized TCP protocol for mobile hosts
  - splitting of the TCP connection at, e.g., the foreign agent into 2 TCP connections, no real end-to-end connection any longer
  - hosts in the fixed part of the net do not notice the characteristics of the wireless part



# I-TCP socket and state migration



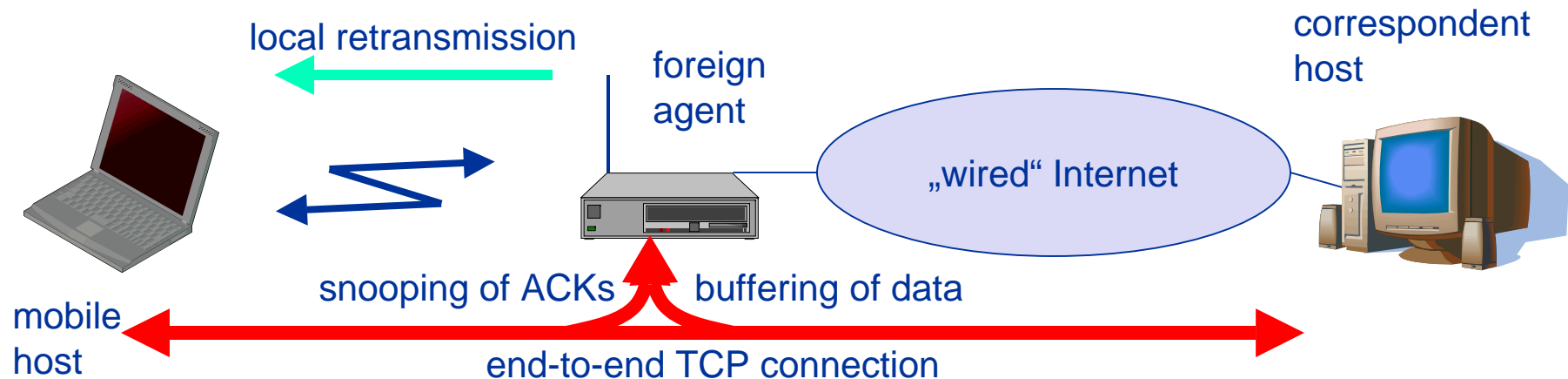


# Indirect TCP II

- Advantages
  - no changes in the fixed network necessary, no changes for the hosts (TCP protocol) necessary, all current optimizations to TCP still work
  - transmission errors on the wireless link do not propagate into the fixed network
  - simple to control, mobile TCP is used only for one hop between, e.g., a foreign agent and mobile host
  - therefore, a very fast retransmission of packets is possible, the short delay on the mobile hop is known
- Disadvantages
  - loss of end-to-end semantics, an acknowledgement to a sender does now not any longer mean that a receiver really got a packet, foreign agents might crash
  - higher latency possible due to buffering of data within the foreign agent and forwarding to a new foreign agent

# Early approach: Snooping TCP I

- “Transparent” extension of TCP within the foreign agent
  - buffering of packets sent to the mobile host
  - lost packets on the wireless link (both directions!) will be retransmitted immediately by the mobile host or foreign agent, respectively (so called “local” retransmission)
  - the foreign agent therefore “snoops” the packet flow and recognizes acknowledgements in both directions, it also filters ACK’s
  - changes of TCP only within the foreign agent



# Snooping TCP II

- Data transfer to the mobile host
  - FA buffers data until it receives ACK of the MH, FA detects packet loss via duplicated ACK's or time-out
  - fast retransmission possible, transparent for the fixed network
- Data transfer from the mobile host
  - FA detects packet loss on the wireless link via sequence numbers, FA answers directly with a NACK to the MH
  - MH can now retransmit data with only a very short delay
- Integration of the MAC layer
  - MAC layer often has similar mechanisms to those of TCP
  - thus, the MAC layer can already detect duplicated packets due to retransmissions and discard them
- Problems
  - snooping TCP does not isolate the wireless link as well as I-TCP
  - snooping might be useless depending on encryption schemes

# Early approach: Mobile TCP

- Special handling of lengthy and/or frequent disconnections
- M-TCP splits as I-TCP does
  - unmodified TCP fixed network to supervisory host (SH)
  - optimized TCP SH to MH
- Supervisory host
  - no caching, no retransmission
  - monitors all packets, if disconnection detected
    - **set sender window size to 0**
    - **sender automatically goes into persistent mode**
  - old or new SH reopen the window
- Advantages
  - maintains semantics, supports disconnection, no buffer forwarding
- Disadvantages
  - loss on wireless link propagated into fixed network
  - adapted TCP on wireless link

# Fast retransmit/fast recovery

- Change of foreign agent often results in packet loss
  - TCP reacts with slow-start although there is no congestion
- Forced fast retransmit
  - as soon as the mobile host has registered with a new foreign agent, the MH sends duplicated acknowledgements on purpose
  - this forces the fast retransmit mode at the communication partners
  - additionally, the TCP on the MH is forced to continue sending with the actual window size and not to go into slow-start after registration
- Advantage
  - simple changes result in significant higher performance
- Disadvantage
  - further mix of IP and TCP, no transparent approach

# Transmission/time-out freezing

- Mobile hosts can be disconnected for a longer time
  - no packet exchange possible, e.g., in a tunnel, disconnection due to overloaded cells or multiplex with higher priority traffic
  - TCP disconnects after time-out completely
- TCP freezing
  - MAC layer is often able to detect interruption in advance
  - MAC can inform TCP layer of upcoming loss of connection
  - TCP stops sending, but does now not assume a congested link
  - MAC layer signals again if reconnected
- Advantage
  - scheme is independent of data
- Disadvantage
  - TCP on mobile host has to be changed, mechanism depends on MAC layer

# Selective retransmission

- TCP acknowledgements are often cumulative
  - ACK n acknowledges correct and in-sequence receipt of packets up to n
  - if single packets are missing quite often a whole packet sequence beginning at the gap has to be retransmitted (go-back-n), thus wasting bandwidth
- Selective retransmission as one solution
  - RFC2018 allows for acknowledgements of single packets, not only acknowledgements of in-sequence packet streams without gaps
  - sender can now retransmit only the missing packets
- Advantage
  - much higher efficiency
- Disadvantage
  - more complex software in a receiver, more buffer needed at the receiver

# Transaction oriented 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, T-TCP, 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
- Advantage
  - efficiency
- Disadvantage
  - requires changed TCP
  - mobility not longer transparent



# Comparison of different approaches for a “mobile” TCP

| 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                               |
| Snooping TCP                       | “snoops” data and acknowledgements, local retransmission    | transparent for end-to-end connection, MAC integration possible               | problematic with encryption, bad isolation of wireless link                     |
| 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 |
| Fast retransmit/<br>fast recovery  | avoids slow-start after roaming                             | simple and efficient  | mixed layers, not transparent   |
| Transmission/<br>time-out freezing | freezes TCP state at disconnect, resumes after reconnection | independent of content or encryption, works for longer interrupts             | changes in TCP required, MAC dependant  |
| Selective retransmission           | retransmit only lost data                                   | very efficient  | slightly more complex receiver software, more buffer needed                     |
| Transaction oriented TCP           | combine connection setup/release and data transmission      | Efficient for certain applications  | changes in TCP required, not transparent  |

# TCP Improvements I

- Initial research work
  - Indirect TCP, Snoop TCP, M-TCP, T/TCP, SACK, Transmission/time-out freezing, ...
- TCP over 2.5/3G wireless networks
  - Fine tuning today's TCP
  - Learn to live with
    - Data rates: 64 kbit/s up, 115-384 kbit/s down; asymmetry: 3-6, but also up to 1000 (broadcast systems), periodic allocation/release of channels
    - High latency, high jitter, packet loss
  - Suggestions
    - Large (initial) sending windows, large maximum transfer unit, selective acknowledgement, explicit congestion notification, time stamp, no header compression
  - Widespread use
    - i-mode running over FOMA
    - WAP 2.0 ("TCP with wireless profile")

$$BW \leq \frac{0.93 * MSS}{RTT * \sqrt{p}}$$

- max. TCP BandWidth
- Max. Segment Size
- Round Trip Time
- loss probability

# TCP Improvements II

- Performance enhancing proxies (PEP, RFC 3135)

- Transport layer
  - Local retransmissions and acknowledgements
- Additionally on the application layer
  - Content filtering, compression, picture downscaling
  - E.g., Internet/WAP gateways
  - Web service gateways?
- Big problem: breaks end-to-end semantics
  - Disables use of IP security
  - Choose between PEP and security!

- More open issues

- RFC 3150 (slow links)
  - Recommends header compression, no timestamp
- RFC 3155 (links with errors)
  - States that explicit congestion notification cannot be used
- In contrast to 2.5G/3G recommendations!

