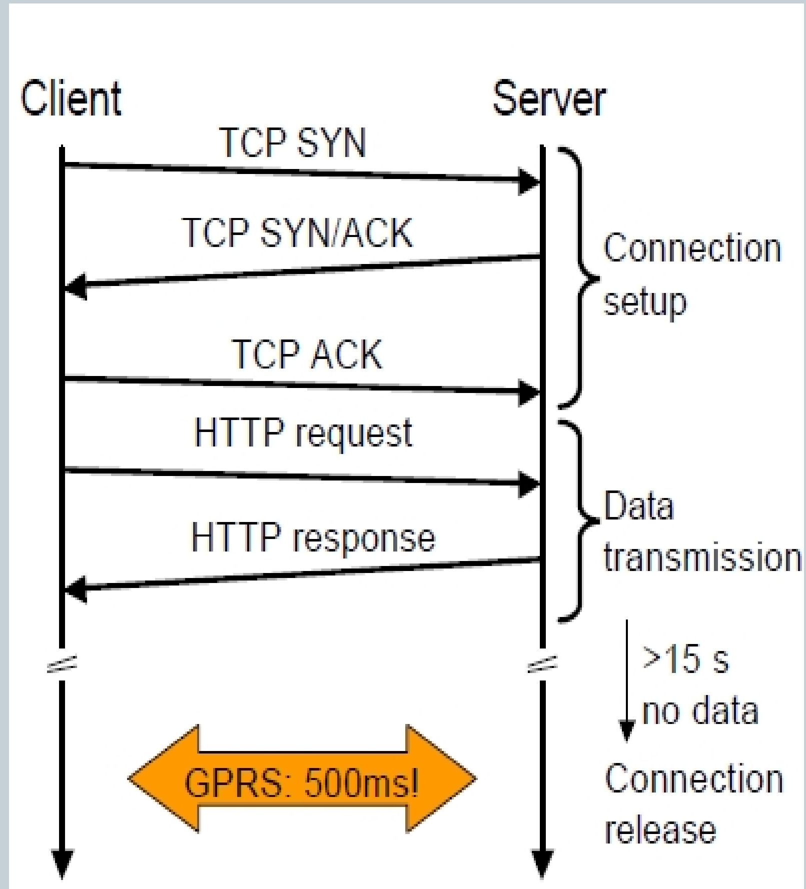
Mobile Computing
Lecture 21
Protocols for Mobile Computing 2

Contents

- Transport Layer
- Motivation
- Snooping TCP
- Mobile TCP
- Fast retransmit/fast recovery
- Selective retransmission
- Transaction oriented TCP
- Comparison

Transport Layer

- E.g. HTTP (used by web services) typically uses TCP
  - Reliable transport between client and server required
- TCP
  - Steam oriented, not transaction oriented
  - Network friendly: time-out
    - Congestion
    - slow down transmission
- Well known – TCP wrongly assumes congestion in wireless and mobile networks when
  - Packet losses due to transmission errors
  - Packet loss due to change of network
- Result
  - Severe ***performance degradation***



Motivation I

- Transport protocols typically designed for
  - Fixed end-systems
  - Fixed, wired networks
- Research activities
  - How to improve TCP performance in wireless networks
  - Maintain congestion control behavior
  - Efficient retransmissions
- TCP congestion control in fixed networks
  - Timeouts/Packet loss typically due to (temporary) overload
  - Routers discard packets when buffers are full
  - TCP recognizes congestion only indirectly via missing ACKs, retransmissions unwise, since they increase congestion
  - slow-start algorithm as reaction

Motivation II

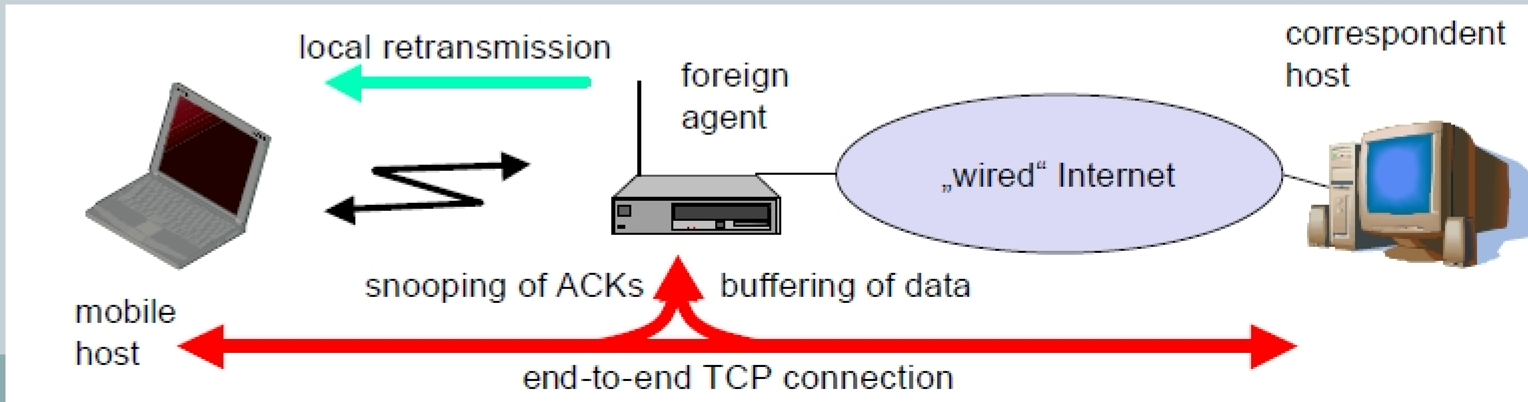
- TCP slow-start algorithm
  - sender calculates a congestion window for a receiver
  - start with a congestion window size equal to one segment (packet)
  - Exponentially increase congestion window till congestion threshold, then linear increase
  - Timeout/missing acknowledgement causes reduction of congestion threshold to half of the current congestion window
  - congestion window starts again with one segment
- TCP fast retransmit/fast recovery
  - TCP sends an ACK only after receiving a packet
  - If sender receives duplicate ACKs, this is due to gap in received packets at the receiver
  - Receiver got all packets up to the gap and is actually receiving packets
  - **Conclusion: packet loss *not due to congestion, retransmit,*** continue with current congestion window (do not use slow-start)

Influences of Wireless/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 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 packets in transit to the old access point and forwarding is not possible
- The performance of an unchanged TCP degrades severely
  - TCP cannot be changed fundamentally due to large installed base in the fixed network, TCP for mobility has to remain compatible
  - the basic TCP mechanisms keep the whole Internet together

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 ACKs
  - 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 ACKs 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 with 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 good as I-TCP
  - snooping might be tough if packets are encrypted

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
  - Cooperation required between 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 mux. 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/ fast recovery	avoids slow-start after roaming	simple and efficient	mixed layers, not transparent
Transmission/ 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