CS 480: MOBILE NETWORKS

Lecture PowerPoints

Lecture 12

Mobile Transport Layer

Topics Covered

- Introduction
- Traditional TCP
- Classical TCP improvements

Introduction

Introduction (1)

- Supporting mobility only on lower layers up to the network layer is not enough to provide mobility support for applications
- Most applications rely on a transport layer, such as TCP or UDP in the case of the internet
- Two functions of the transport layer in the internet are
  - checksumming over user data and
  - multiplexing/ demultiplexing of data from/to applications
- While the network layer only addresses a host, ports in UDP or TCP allow dedicated applications to be addressed

Introduction (2)

- UDP is connectionless and does not give certain guarantees about reliable data delivery, TCP is much more complex and needs special mechanisms to be useful in mobile environments
- Mobility support in IP (such as mobile IP) is already enough for UDP to work
- The main difference between UDP and TCP is that TCP offers connections between two applications
- Within a connection, TCP can give certain guarantees, such as in-order delivery or reliable data transmission using retransmission techniques

Traditional TCP

Congestion control (1)

- A transport layer protocol such as TCP has been designed for fixed networks with fixed end-systems
- The probable reason for packet loss in a fixed network is a temporary overload at some point in the transmission path, i.e., a state of congestion at a node
- Congestion may appear from time to time even in carefully designed networks
- If the packet buffers of a router are filled and the router cannot forward the packets fast enough the only thing a router can do in this situation is to drop packets.
  - This is because the sum of the input rates of packets destined for one output link is higher than the capacity of the output link,
- A dropped packet is lost for the transmission, and the receiver notices a gap in the packet stream

Congestion control (2)

- The receiver does not directly tell the sender which packet is missing, but continues to acknowledge all in-sequence packets up to the missing one
- The sender notices the missing acknowledgement for the lost packet and assumes a packet loss due to congestion
- To mitigate congestion, TCP slows down the transmission rate dramatically.
- All other TCP connections experiencing the same congestion do exactly the same so the congestion is soon resolved
- This cooperation of TCP connections in the internet is one main reason for its survival as it is today

Slow start (1)

- TCP's reaction to a missing acknowledgement is quite dramatic, but it is necessary to get rid of congestion quickly
- The slowing of transmission rates after the detection of congestion is called *slow start*
- The sender always calculates a *congestion window* for a receiver.
- The start size of the congestion window is one segment (TCP packet).
- The sender sends one packet and waits for acknowledgement
  - If the acknowledgement arrives, the sender increases the congestion window by one, now sending two packets (congestion window =2).

Slow start (2)

- After arrival of two corresponding acknowledgements, the sender adds 2 to the congestion window, one for each of the acknowledgements (congestion window =4).
- This scheme doubles the congestion window every time the acknowledgements come back, which takes one round trip time (RTT).
- This is called the exponential growth of the congestion window in the slow start mechanism
- The exponential growth stop at the *congestion threshold*.
- When congestion window reaches the congestion threshold, further increase of the transmission rate is only linear by adding 1 to the congestion window each time the acknowledgment comes back

Slow start (3)

- Linear increase continues until a time-out at the sender occurs due to a missing acknowledgement, or until the sender detects a gap in transmitted data because of continuous acknowledgements for the same packet
- In either case the sender sets the congestion threshold to half of the current congestion window
- The congestion window itself is set to one segment and the sender starts sending a single segment.
- The exponential growth starts once more up to the new congestion threshold, then the window grows in linear fashion

Fast retransmit/ fast recovery (1)

Two things lead to a reduction of the congestion threshold

- One is a sender receiving continuous acknowledgements for the same packet. This informs the sender of two things.
  - one is that the receiver got all packets up to the acknowledged packet in sequence.
    - In TCP, a receiver sends acknowledgements only if it receives any packets from the sender.
  - Receiving acknowledgements from a receiver also shows the receiver continuously receives something from the sender.
- The gap in the packet stream is not due to severe congestion, but a simple packet loss due to a transmission error.
- The sender can now retransmit the missing packet(s) before the timer expires.

Fast retransmit/ fast recovery (2)

- This behavior is called fast retransmit.
- The receipt of acknowledgements shows that there is no congestion to justify a slow start.
- The sender can continue with current congestion window
- The sender performs a *fast recovery* from packet loss
- This mechanism can improve the efficiency of TCP dramatically
- The other reason for activating a slow start is a time-out due to a missing acknowledgement.
  - TCP using fast retransmit/fast recovery interprets this congestion in the network and activates the slow start mechanism.

Implications on mobility (1)

- While slow start is one of the most useful mechanisms in fixed networks, it drastically decreases the efficiency of TCP if used together with mobile receivers or senders.
- The reason for this is the use of slow start under wrong assumptions.
- From a missing acknowledgement, TCP concludes a congestion situation.
- While this may also happen in networks with mobile and wireless end-systems, it is not the main reason for packet loss.
- Error rates on wireless links are orders of magnitude higher compared to fixed fiber or copper links.
- Mobility itself can cause packet loss.

Implications on mobility (2)

- For example when using mobile IP, there could still be some packets in transit to the old foreign agent while the mobile node moves to the new foreign agent.
- The old foreign agent may not be able to forward those packets to the new foreign agent or even buffer the packets if disconnection of the mobile node takes too long.
- This packet loss is caused by problems of rerouting traffic.
- The TCP mechanism of detecting missing acknowledgements via time-outs and concluding packet loss due to congestion cannot distinguish between the different causes.

Implications on mobility (3)

- This is a fundamental design problem in TCP
- If acknowledgements are missing , standard TCP reacts with slow start, which does not help in the case of transmission errors over wireless links
  - and which does not help during handover.
- This behavior results in severe performance degradation of an unchanged TCP if used together with wireless links or mobile nodes
- However, one cannot change TCP completely just to support mobile users or wireless links.
- The same arguments given to keep IP unchanged also apply to TCP.

Implications on mobility (4)

- The installed base of computers using TCP is too large to be changed and more important, mechanisms such as slow start keep the internet operable.
- Every enhancement to TCP has to remain compatible with the standard TCP and not affect the cautious behavior of TCP in case of congestion.

Classical TCP improvements

Classical TCP improvements

- Together with the introduction of WLANs in the mid 90s, several research projects were started with the goal to increase TCP’s performance in wireless and mobile environments.

Indirect TCP (1)

- Two competing insights led to the development of indirect TCP (I-TCP)
  - One is that TCP performs poorly together with wireless links
  - The other is that TCP within the fixed network cannot be changed
- I-TCP segments a TCP connection into a fixed part and a wireless part
- Figure 1 shows an example with a mobile host connected via a wireless link and an access point to the “wired” internet where the correspondent host resides.
- The correspondent node could also use wireless access.
- Standard TCP is used between the fixed computer and the access point.

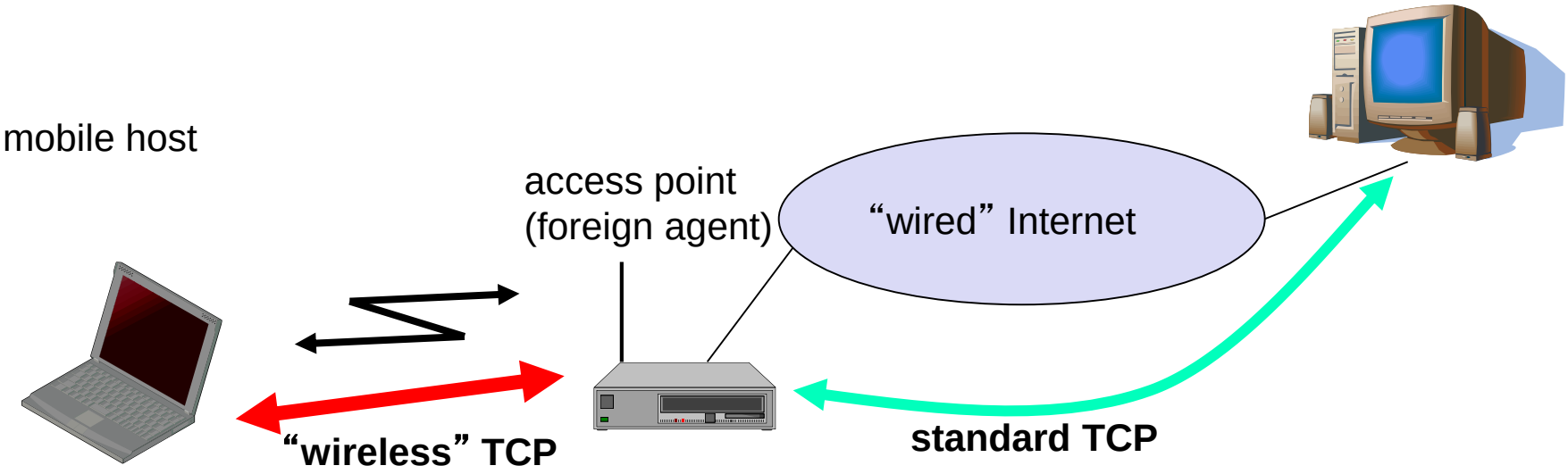

Figure 1: Indirect TCP segments a TCP connection into two parts.

Indirect TCP (2)

- No computer in the internet recognizes any changes to TCP.
  - Instead of the mobile host, the access point now terminates the standard TCP connection, acting as a proxy.
- The access point is now seen as the mobile host for the fixed host and as the fixed host for the mobile host
- Between the access point and the mobile host, a special TCP adapted to wireless links is used
- The correspondent node in the fixed network does not notice the wireless link or the segmentation of the connection.
- I-TCP requires several actions as soon as a handover takes place.

Indirect TCP (3)

- As Figure 2 demonstrates, not only does the packets have to be redirected using e.g., mobile IP.
- In the example shown, the access points acts as a proxy buffering packets for retransmission
- After the handover, the old proxy must forward buffered data to the new proxy because it has already acknowledged the data.
- Besides buffer content, the sockets of the proxy must migrate to the new foreign agent located in the access point
- No new connection may be established for the mobile host and the correspondent node must not see any changes in connection state.

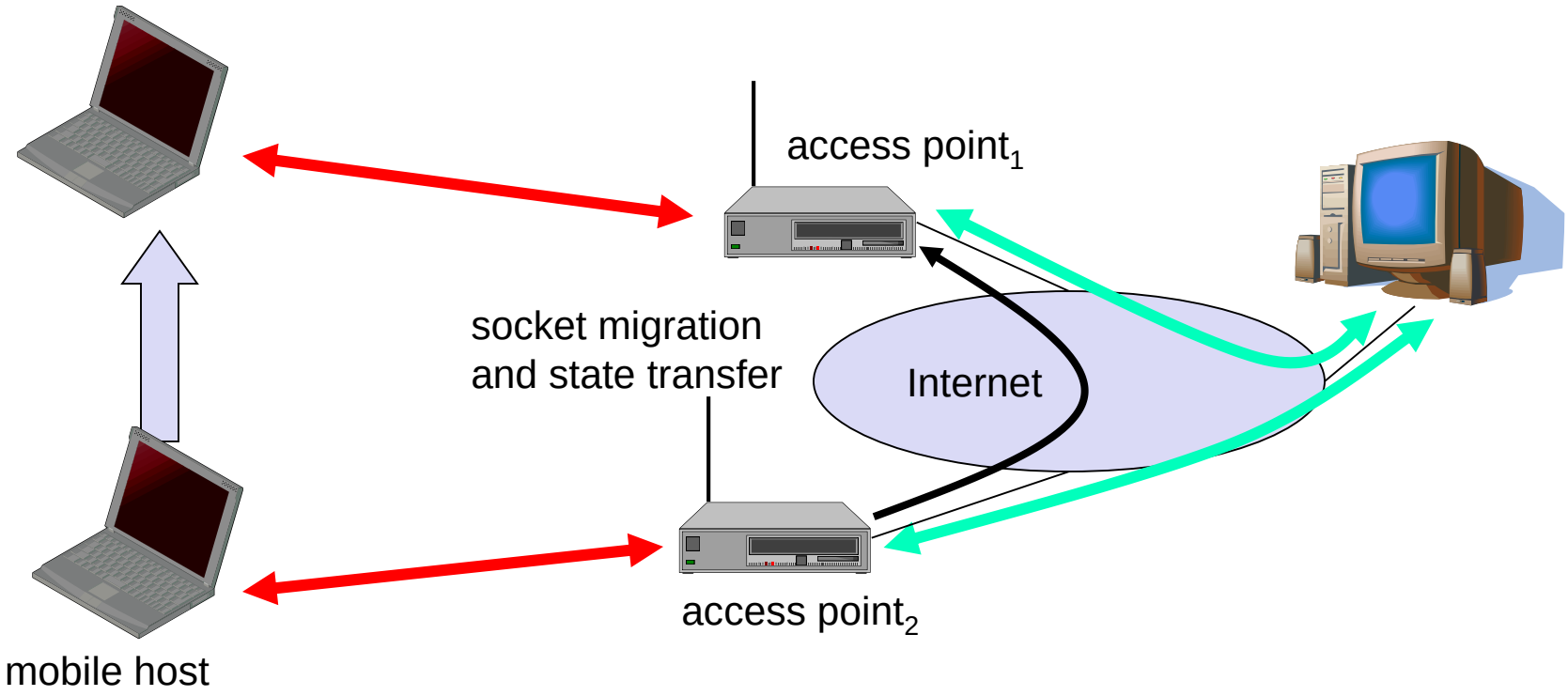

Figure 2: Socket and state migration after handover of a mobile host

Advantages

- I-TCP does not require any changes in the TCP protocol as used by the hosts in the fixed network or other hosts in a wireless network that do not use this optimization.
  - All current optimizations for TCP still work between the foreign agent and the corresponding host
- Due to strict partitioning into two connections, transmission errors on the wireless link, i.e., lost packets, cannot propagate into the fixed network
- Simple to control, wireless 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

- The loss of the end-to-end semantics of TCP might cause problems if the foreign agent partitioning the TCP connection crashes
- Increased handover latency may be much more problematic due to buffering of data within the foreign agent and forwarding to a new foreign agent
- The foreign agent must be a trusted entity because the TCP connections end at this point.
  - If users apply end-to-end encryption, the foreign agent has to be integrated into all security mechanisms

Snooping TCP (1)

- One of the drawbacks of I-TCP is the segmentation of the single TCP connection into two TCP connections.
  - This looses the original end-to-end semantic.
- Snooping TCP works transparently and leaves the TCP end-to-end connection intact.
- The main function of the enhancement is to buffer data close to the mobile host to perform fast local retransmission in case of packet loss.
- A good place for the enhancement of TCP could be the foreign agent in the mobile IP context (see Figure 3)
- In this approach, the foreign agent buffers all packets with *destination mobile host* and additionally "snoops" the packet flow in both directions to recognize acknowledgements.

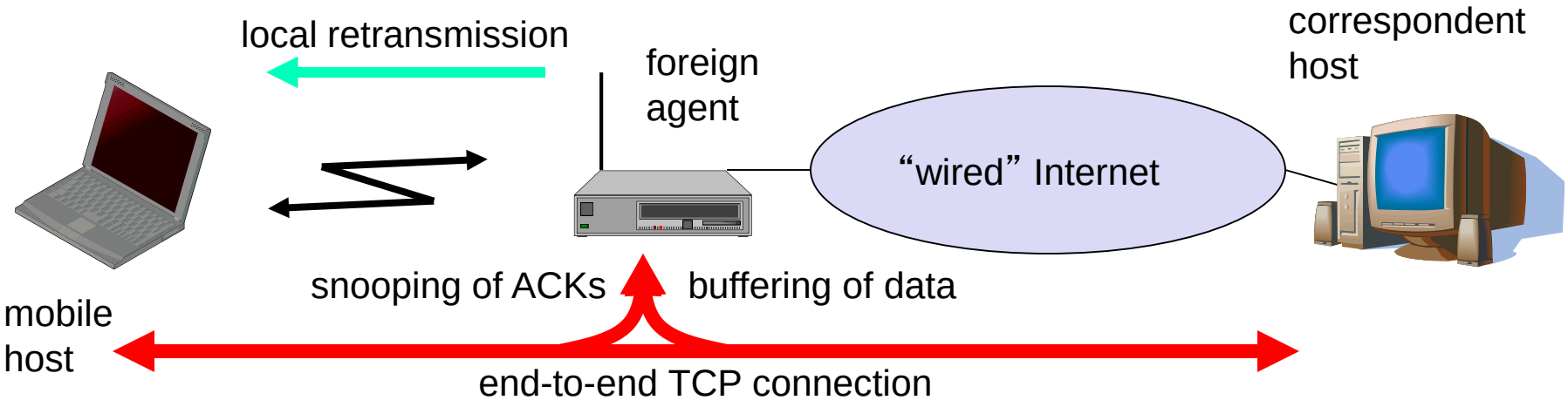

Figure 3: Snooping TCP as a transparent TCP extension

Snooping TCP (2)

- The reason for buffering packets toward the mobile node is to enable the foreign agent to perform a local retransmission in case of packet loss on a wireless link.
- The foreign agent buffers every packet until it receives an acknowledgement from the mobile host
- To remain transparent, the foreign agent must not acknowledge data to the correspondent host.
  - The correspondent host would believe that the mobile host had received data and would violate end-to-end semantic in case of a foreign agent failure.

Snooping TCP (3)

- Data transfer from the mobile host with *destination correspondent host* works as follows:
  - The foreign agent snoops into the packet stream to detect gaps in the sequence numbers of TCP
  - As soon as the foreign agent detects a missing packet, it returns a negative acknowledgement (NACK) to the mobile host. The mobile host can now retransmit the missing packet immediately

Advantages of extending the functions of a foreign agent with a “snooping” TCP

- The end-to-end TCP semantic is preserved.
  - If the foreign agent crashes, the correspondent host and mobile host will have a consistent view of the TCP connection.
  - The approach automatically falls back to standard TCP if the enhancements stop working
- The correspondent host does not need to be changed; most of the enhancements are in the foreign agent.
- It does not need a handover of state as soon as the mobile host moves to another foreign agent
- It does not matter if the next foreign agent uses the enhancement or not.
  - If not the approach falls back to standard solution

Disadvantages of the scheme

- Snooping TCP does not isolate the behavior of the wireless link the way I-TCP does.
- Using negative acknowledgements between the foreign agent and mobile host assumes additional mechanisms on the mobile host.
  - This approach is no longer transparent for arbitrary mobile hosts
- All efforts for snooping and buffering data may be useless if certain encryption schemes are applied end-to-end between the correspondent host and mobile host

Mobile TCP (1)

- Dropping packets due to handover or higher bit error rates is not the only phenomenon of wireless links and mobility
  - the occurrence of lengthy and/or frequent disconnections is another problem
- Quite often mobile users cannot connect at all.
- One example is islands of wireless LANs inside buildings but no coverage of the whole campus.
- What happens to standard TCP in the case of disconnection?
- A TCP sender tries to retransmit data controlled by a retransmission timer that doubles with each unsuccessful retransmission attempt, up to a maximum of one minute.
  - This means the sender tries to retransmit an unacknowledged packet every minute and will give up after 12 retransmissions.

Mobile TCP (2)

- What happens if connectivity is back earlier than this?
  - No data is successfully transmitted for a period of one minute!
- The retransmission time-out is still valid and the sender has to wait.
  - The sender also goes into slow start because it assumes congestion
- What happens in the case of I-TCP if the mobile host is disconnected?
- The proxy buffers more and more data, so the longer the period of disconnection, the more buffer is needed.
- If a handover follows a disconnection, even more state has to be transferred to the new proxy.
- The snooping approach also suffers from being disconnected.

Mobile TCP (3)

- The mobile host will not be able to send ACKs and thus snooping cannot help in this situation.
- The M-TCP (mobile TCP) approach has the same goals as I-TCP and snooping TCP
  - To prevent the sender window from shrinking if bit errors or disconnection but not congestion cause current problems
  - M-TCP wants to improve overall throughput, to lower the delay, to maintain end-to-end semantics of TCP, and to provide a more efficient handover
  - M-TCP is especially adapted to problems arising from lengthy or frequent disconnections
- M-TCP splits the TCP connection into two parts as I-TCP does.

Mobile TCP (4)

- An unmodified TCP is used on the standard host-*supervisory host* (SH) connection, while an optimized TCP is used on the SH-MH connection.
- The supervisory host is responsible for exchanging data between both parts similar to a proxy in I-TCP
- The M-TCP approach assumes a relatively low bit error rate on the wireless link.
- Thus it does not perform caching/retransmission of data via the SH.
- If a packet is lost on the wireless link, it has to be retransmitted by the original sender.
- This maintains the TCP end-to-end semantics.

Mobile TCP (5)

- The SH monitors all packets sent to the MH and ACKs returned from the MH.
- If the SH does not receive an ACK for some time, it assumes that the MH is disconnected.
- It chokes the sender by setting the senders's window size to 0.
- This forces the sender to go into *persistent mode and* the sender will not try to retransmit data.
- As soon as the SH detects connectivity again, it reopens the window of the sender to the old value.
- The sender can continue sending at full speed
- The wireless side uses an adapted TCP that can recover from packet loss much faster.

Mobile TCP (6)

- This modified TCP does not use slow start, thus M-TCP needs a *bandwidth manager* to implement fair sharing over the wireless link.

Advantages of M-TCP are the following

- It maintains the TCP end-to-end semantics.
  - The SH does not send any ACK itself but forwards the ACKs from the MH.
- If the MH is disconnected, it avoids useless retransmissions, slow starts or breaking connections by simply shrinking the sender's window to 0.
- Since it does not buffer data in the SH as I-TCP does, it is not necessary to forward buffers to a new SH.
  - Lost packets will be automatically retransmitted to the new SH

Disadvantages

- As the SH does not act as proxy like in I-TCP, packet loss on the wireless link due to bit errors is propagated to the sender.
- M-TCP assumes low bit error rates, which is not always a valid assumption
- A modified TCP on wireless link not only requires modifications to the MH protocol software but also new network elements like the bandwidth manager.