# CSC474/574 Computer Networking

## Homework 5: Chapter 5 Answers

**Name:** Junesh Gautam
**Student ID:** 101150362

**Q1. Elaborate on the main functions of the Transport layer.**

**Answer:**
The Transport layer is responsible for providing *end-to-end* communication services between processes running on different hosts. Its key functions include:

- **Segmentation and Reassembly:** Breaking application data into segments at the sender, and reassembling them at the receiver.
- **Connection Management:** Establishing, maintaining, and tearing down logical connections (e.g., TCP's three-way handshake).
- **Flow Control:** Regulating data transmission to prevent overwhelming slower or congested receivers (e.g., TCP's window mechanism).
- **Error Control:** Retransmitting lost segments and detecting corrupted segments (via sequence numbers, checksums, etc.).
- **Multiplexing and Demultiplexing:** Identifying data for different applications using port numbers, allowing multiple concurrent communications between hosts.

**Q2. Consider a simple application-level protocol built on top of UDP that allows a client to retrieve a file from a remote server residing at a well-known address. The client first sends a request with a file name, and the server responds with a sequence of data packets containing different parts of the requested file. To ensure reliability and sequenced delivery, client and server use a stop-and-wait protocol. Ignoring the obvious performance issue, do you see a problem with this protocol? Think carefully about the possibility of processes crashing.**

**Answer:**
Yes, there is a subtle *problem.* The protocol lacks a way to handle process crashes or restarts. If either the client or server crashes and restarts while the other is still sending segments:

- The newly started process may not recognize old sequence numbers or requests.
- Another service using the same port might receive old or duplicate packets, causing confusion and potential data corruption.

- Without a robust handshake or unique connection identifiers, out-of-context or duplicated segments can be misinterpreted after a crash.

Even though the stop-and-wait mechanism ensures basic reliability (ACKing each segment), it doesn't handle these corner cases of process lifecycles. This highlights why a connection-oriented protocol like TCP is generally more robust for file retrieval.

**Q3. For a 1-Gbps network operating over 4000 km, the delay is the limiting factor, not the bandwidth. Consider a MAN with the average source and destination 20 km apart. At what data rate does the round-trip delay due to the speed of light equal the transmission delay for a 1-KB packet?**

**Answer:**
We want the data rate $R$ at which *transmission time* of a 1-KB packet equals the *round-trip propagation delay* over 20 km (one-way). Assume:

$$\text{Distance} = 20\,\text{km (one way)} \implies 40\,\text{km (round trip)}$$

$$\text{Speed of signal (approx.)} \approx 2 \times 10^8\,\text{m/s}$$

$$\text{Propagation Time (round trip)} = \frac{40{,}000\,\text{m}}{2 \times 10^8\,\text{m/s}} = 2 \times 10^{-4}\,\text{s} = 0.0002\,\text{s}\,(0.2\,\text{ms})$$

A 1-KB packet is $\approx 1024 \times 8 = 8192$ bits. Let $\Delta t$ be $0.0002\,\text{s}$. We set:

$$\text{Transmission time} = \frac{8192\,\text{bits}}{R} = \Delta t = 0.0002\,\text{s}.$$

So:

$$R = \frac{8192}{0.0002}\,\text{bits/s} = 40{,}960{,}000\,\text{bits/s} \approx 40.96\,\text{Mbps}.$$

Hence, $\boxed{\text{about } 41\,\text{Mbps}}$ is the data rate where the packet's transmission delay equals the round-trip propagation delay.

**Q4. Datagram fragmentation and reassembly are handled by IP and are invisible to TCP. Does this mean that TCP does not have to worry about data arriving in the wrong order?**

**Answer:**
No, TCP must still handle potential reordering. Even though IP fragmentation and reassembly are transparent to TCP, packets (IP datagrams) can arrive out of order due to:

- Different routing paths and varying network conditions.
- Potential duplication or delay from intermediate network devices.

TCP uses sequence numbers in its headers to correctly reassemble and reorder segments. So the fact that IP does fragmentation "invisibly" does not negate the possibility of out-of-order arrival at the transport layer. TCP *always* expects and is prepared for out-of-sequence data segments.