Transport Layer

Transport Layer

- Provides end to end connectivity between applications
- Functions provided include:
  1. Segmenting upper layer applications
  2. Establishing end to end operations (a logical connection)
  3. Sends segments from one end host to another end host
  4. Ensure data reliability (optional)

Segmenting upper layer applications is achieved by **multiplexing using ports**:

- ③ multiple applications share a transport connection.
- ③ transport is achieved segment by segment – each segment is autonomous
- ③ segments are processed first-come, first-served and may go to one or many destination hosts

How it works:

Software in the source machine sets a port number in for each application before transmission. This information is contained in the *header* and includes messages type, originating program, protocol used, etc.

Port numbers are used to multiplex from the transport layer to the application layer (see RFC 1700)

1 – 1023	well known ports, reserved
1024 – 65, 535	can be registered, can have local significance, are not assigned exclusively to one protocol

Moving the Data Reliably:

To use reliable transport services, 2 hosts must establish a connection-oriented session (a call is placed and accepted).

Both the sending and receiving applications inform their o/s that a connection will be initiated:

- Messages are sent to synchronize connection parameters, verify that transfer is authorized, and acknowledge that both sides are ready
- After synchronization, data transfer begins
- Communication during transfer verifies that data is received correctly

How it Works:

1. Synchronization: The Three Way Handshake
    - Host A requests synchronization (SYN)
    - Host B acknowledges request (ACK) and synchronizes (SYN)
    - Host A acknowledges synchronization (ACK)

2. Reliable Delivery: Positive Acknowledgement
    - Sender keeps a record of each segment sent and waits for ACK before sending the next segment
    - Timers cause retransmit if it expires

3. Ending the Connection: The Three Way Handshake
    - Host A requests finish (FIN)
    - Host B acknowledges request (ACK) and finishes (FIN)
    - Host A acknowledges synchronization (ACK)