# COMPUTER NETWORKS

*Unit IV — Transport Layer Protocols*

**Topics Covered (09 Hours)**

Process-to-Process Delivery & Transport Services
Socket Programming Concepts (IP & Ports)
Transport Protocol Addressing & TSAPs
Connection Establishment (Three-Way Handshake)
Connection Release & The Two-Army Problem
Flow Control, Buffering, and Multiplexing
TCP Header, Connection States & Congestion Control
UDP Protocol, SCTP & RTP Overview
Transport Protocols for Wireless Networks

*Compiled in a simple, easy-to-understand, book style format for students of Computer Science and Engineering*

# TABLE OF CONTENTS

4.1 Process-to-Process Delivery & Socket Programming ..... 1
Socket Programming ..... 1
Detailed Socket Programming Lifecycle ..... 1
4.2 Elements of Transport Layer Protocols ..... 1
Addressing (TSAPs & NSAPs) ..... 1
Connection Establishment (Three-Way Handshake) ..... 1
Connection Release ..... 1
Flow Control and Buffering ..... 1
4.3 Transport Layer Protocols: TCP, UDP, SCTP, RTP ..... 1
SCTP and RTP Protocols ..... 1
4.4 TCP Congestion Control & Wireless Networks ..... 1
TCP Congestion Control Algorithms ..... 1
TCP over Wireless Networks ..... 1
Unit Summary ..... 1

# 4.1 Process-to-Process Delivery & Socket Programming

While the Network Layer is responsible for host-to-host delivery (moving packets between computers), the Transport Layer is responsible for process-to-process delivery (moving messages between specific running software processes on those computers).

## Socket Programming

Processes read from and write to the network using Sockets. A socket is the logical interface between application programs and the network protocol stack. Sockets are identified by combining the IP address (identifying the host machine) and the Port Number (identifying the specific process on that machine, such as port 80 for HTTP or port 443 for HTTPS).

> **Socket**
> *A socket is an endpoint for communication, defined by the binding of an IP address and a Port Number.*

### *Detailed Socket Programming Lifecycle*

The network application interface relies on standard operating system API calls to control socket state:

- **socket():** Creates a raw, unassigned socket file descriptor in memory, specifying the protocol family (IPv4/IPv6) and type (TCP/UDP).
- **bind():** Associates the created socket with a specific local IP address and Port Number. Necessary for servers so clients know where to connect.
- **listen():** An exclusive server system call. It transitions the socket into a passive state, allocating a queue to hold incoming client connection requests.
- **accept():** Called by the server. It blocks execution until a connection request arrives. It then returns a new socket descriptor dedicated to communicating with that specific client, freeing the original socket to continue listening.
- **connect():** Called by the client. It initiates the three-way handshake connection sequence to the specified server IP and port.
- **read() / write() or recv() / send():** Used by both peers to exchange actual data bytes across the established channel.
- **close():** Deallocates the socket descriptor, releasing port numbers and terminating the connection.

# 4.2 Elements of Transport Layer Protocols

Transport protocols must run over unreliable network layers (like IP), meaning they must handle connection setup, flow control, and data integrity over a packet-dropping subnet.

## Addressing (TSAPs & NSAPs)

To establish a transport connection, an application must specify which Transport Service Access Point (TSAP) it wants to connect to. TSAPs are mapped to Network Service Access Points (NSAPs) at the network layer. For example, a TSAP is a port number, and an NSAP is an IP address.

## Connection Establishment (Three-Way Handshake)

The main challenge of connection setup is handling delayed duplicate packets from old, closed connections. To prevent this, TCP uses a Three-Way Handshake:

- **1. SYN:** The client sends a SYN packet with a random sequence number x to the server.
- **2. SYN-ACK:** The server responds with a SYN-ACK packet, acknowledging x (ACK = x+1) and choosing its own sequence number y.
- **3. ACK:** The client acknowledges y (ACK = y+1) and sends the first data bits.

If an old, delayed SYN packet arrives at the server, the server responds with a SYN-ACK containing the old sequence number. When the client receives this, it recognizes the sequence mismatch and rejects the connection by sending a Reset (RST) packet, preventing accidental duplicate connections.

## Connection Release

Releasing a connection can be Asymmetric (one side abruptly shuts down, which can cause data loss) or Symmetric (each side closes its direction independently). Symmetric release requires solving the Two-Army Problem, where two generals must agree on a coordinate attack time over an unreliable communication channel. No protocol can guarantee absolute agreement, so timers are used to force disconnects.

## Flow Control and Buffering

Like the data link layer, the transport layer manages flow control, but across the entire internet. It uses sliding windows where the receiver specifies how much buffer space it has left (Window Advertisement). Buffers can be allocated dynamically at the sender, receiver, or both.

# 4.3 Transport Layer Protocols: TCP, UDP, SCTP, RTP

The internet protocol suite contains two primary transport protocols: TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

| Feature | TCP | UDP |
|---|---|---|
| Connection Style | Connection-Oriented. Requires handshake. | Connectionless. Packets sent instantly. |
| Reliability | Reliable. Guarantees in-order delivery via ACKs. | Unreliable. Packets can be lost/out-of-order. |
| Header Size | 20 to 60 bytes. | Fixed 8 bytes. Very lightweight. |
| Flow & Congestion | Yes. Implements sliding window and TCP congestion algorithms. | No. It sends data as fast as the application generates it. |
| Data Unit | Byte Stream (handles segments). | Message / Datagram boundaries preserved. |

### *SCTP and RTP Protocols*

- **SCTP (Stream Control Transmission Protocol):** A modern transport protocol combining features of TCP and UDP. It is message-oriented (like UDP) but guarantees reliable, in-order transport (like TCP). It also supports multi-homing (using multiple IP interfaces simultaneously).
- **RTP (Real-time Transport Protocol):** Runs on top of UDP. It appends sequence numbers and timestamps to payloads, allowing media players to reorder packets and synchronize audio/video playback (used in VoIP, streaming).

# 4.4 TCP Congestion Control & Wireless Networks

TCP plays a critical role in preventing global internet collapse by dynamically adjusting its sending rate based on network congestion.

### *TCP Congestion Control Algorithms*

- **Slow Start:** TCP starts by sending a small number of segments (Congestion Window CWND = 1). For every ACK received, it doubles CWND, increasing the rate exponentially until it hits the slow start threshold (ssthresh).
- **Congestion Avoidance:** Once ssthresh is reached, TCP increases CWND linearly (by 1 segment per round-trip time) to carefully probe network limits.
- **Fast Retransmit & Recovery:** If the sender receives three duplicate ACKs, it assumes a packet was lost, immediately retransmits it without waiting for a timeout, and cuts CWND in half (Multiplicative Decrease).

## TCP over Wireless Networks

Standard TCP assumes that any packet loss is caused by congestion. In wireless networks, however, packet loss is frequently caused by transmission noise, handoffs, or signal fading. When standard TCP runs on wireless links, it interprets this noise loss as congestion, immediately drops its window size (slowing transmission), and ruins performance. Solutions include:

- **Indirect TCP (I-TCP):** Splits the TCP connection into two distinct parts: a standard wired connection from the sender to the wireless base station, and a modified wireless connection from the base station to the mobile host. This isolates wireless packet loss from the wired network.
- **Snooping TCP:** The base station monitors (snoops on) the TCP segments passing through. If it detects a packet was lost on the wireless link (by seeing duplicate ACKs or missing sequence numbers), it locally retransmits the packet immediately, preventing the wired sender from entering congestion control.

# Unit Summary

This unit explored the services and protocols of the Transport Layer.

- The Transport Layer provides process-to-process delivery using logical endpoints called Sockets.
- Sockets bind an IP address and a Port Number to identify running application processes.
- Connection establishment relies on a Three-Way Handshake (SYN, SYN-ACK, ACK) to prevent delayed duplicate errors.
- Connection release must handle the symmetric Two-Army Problem using timers and independent closures.
- TCP is connection-oriented, reliable, and byte-stream-based, while UDP is connectionless, fast, and lightweight.
- SCTP offers message-oriented reliable transfer and multi-homing. RTP adds timestamps for real-time streaming.
- TCP congestion control implements Slow Start, Congestion Avoidance, and Fast Retransmit.
- Wireless TCP requires modifications because standard TCP incorrectly interprets radio packet losses as network congestion.