NETWORK AND SECURITY

Unit III — Transport and Application Layer Protocols

Topics Covered in this Unit

Transport Layer Fundamentals: Process-to-Process Delivery, Multiplexing & Port Addressing
Transmission Control Protocol (TCP): Features, Header Structure, Handshakes & Teardown
TCP Flow Control (Sliding Window & rwnd) & Congestion Control (Slow Start, AIMD & Fast Recovery)
User Datagram Protocol (UDP): Characteristics, Header Format & Exhaustive TCP vs. UDP Comparison
Port Categories (Well-Known, Registered, Dynamic) & Socket Programming (Client-Server APIs)
Application Layer Fundamentals: Core Functions, Client-Server Architectures & Service Models
Hypertext Transfer Protocol (HTTP & HTTPS): Web Basics, HTTP/1.1 vs 2 vs 3 & TLS Security
Domain Name System (DNS): Hierarchical Namespace, Recursive vs. Iterative Queries & Record Types
File Transfer Protocol (FTP): Dual-Port Architecture (Control vs. Data), Active vs. Passive Modes
Electronic Mail Architecture: SMTP (Push), POP3 (Download-and-Delete) & IMAP (Server Sync)
High-Yield University Exam Question Bank with Model Answers & Unit Summary

Compiled in a structured, easy-to-understand, textbook style format
Specially curated for B.Tech / B.E. / BCA / MCA / B.Sc Computer Science & IT Students

© Copyright — abhyasmitra.in — All Rights Reserved

COURSE SYLLABUS & MAPPING

[UNIT III SYLLABUS — PRESCRIBED UNIVERSITY CURRICULUM]

This section contains the official syllabus for Unit III. Verify with your university curriculum mapping.

Unit III Syllabus Breakdown:

Transport Layer: Fundamentals, Functions of Transport Layer, Process-to-process communication and port addressing, Segmentation and reassembly, Multiplexing and de-multiplexing; Protocols - TCP - Features, TCP segment structure, connection establishment (three-way handshake), flow control, congestion control; UDP - Characteristics, UDP segment structure, comparison of TCP and UDP; Port Numbers and Socket Programming Concepts: Well-known, registered and dynamic port numbers, Socket concept and client-server communication model; Application Layer: Fundamentals, Functions of Application Layer, Client-server architecture and service models; Protocols: HTTP and HTTPS – web communication basics, DNS – domain name resolution process, FTP – file transfer mechanisms, SMTP, POP3, IMAP – electronic mail protocols overview.

• Course Code: _____ • Semester / Year: _____

TABLE OF CONTENTS

3.1 Transport Layer Fundamentals & Functions

Definition: Transport Layer (Layer 4)

The Transport Layer is the central, core layer of the OSI and TCP/IP protocol architectures responsible for the true Process-to-Process (End-to-End) delivery of complete data messages between application programs running on different host machines across an internetwork.

Core Functions of the Transport Layer

- **1. Process-to-Process Delivery & Port Addressing:** While the Network Layer delivers packets from Host to Host (using IP addresses), the Transport Layer delivers messages from the specific transmitting application process to the exact receiving application process using 16-bit Port Numbers (0 to 65535).
- **2. Segmentation and Reassembly:** At the sender, large application data messages are divided into smaller, manageable chunks called Segments (in TCP) or Datagrams (in UDP). Each segment is assigned a sequence number, enabling the receiving transport layer to correctly reassemble the data in sequential order and discard duplicates.
- **3. Multiplexing and De-multiplexing:**
  - Multiplexing (Sender Side): Collects data chunks from multiple application processes (e.g., Web browser, Mail client, SSH session), wraps them with transport headers containing source/dest port numbers, and passes them down to the single network layer.
  - De-multiplexing (Receiver Side): Examines incoming segments, inspects the Destination Port Number, and directs the payload to the correct application socket.
- **4. End-to-End Connection Management:** Establishes, maintains, and cleanly terminates logical connections (via 3-way handshakes in connection-oriented protocols like TCP).
- **5. End-to-End Flow Control:** Prevents a fast sender from overflowing the receive buffer of a slow receiving process using dynamically advertised sliding windows.
- **6. End-to-End Error Control:** Uses cumulative acknowledgments, checksums, and retransmission timers to guarantee that the entire message arrives without corruption, loss, or duplicate delivery.

3.2 Transmission Control Protocol (TCP)

Key Features of TCP

- **Connection-Oriented:** Requires a formal logical connection establishment (3-Way Handshake) before data transfer begins, and a 4-way handshake to gracefully close.

- **Reliable Byte-Stream Service:** Treats data as a continuous, structured stream of bytes. Every transmitted byte is numbered sequentially, and unacknowledged bytes are automatically retransmitted upon timeout.
- **Full-Duplex Communication:** Data can flow simultaneously in both directions between communicating endpoints over a single TCP connection.
- **Flow and Congestion Control:** Dynamically regulates transmission rates based on receiver buffer capacity (`rwnd`) and network congestion state (`cwnd`).

TCP Segment Header Structure

The TCP header has a minimum size of 20 bytes (when no options are present) and can extend up to 60 bytes with optional fields.

TCP HEADER FORMAT (32-BIT WIDE WORDS)

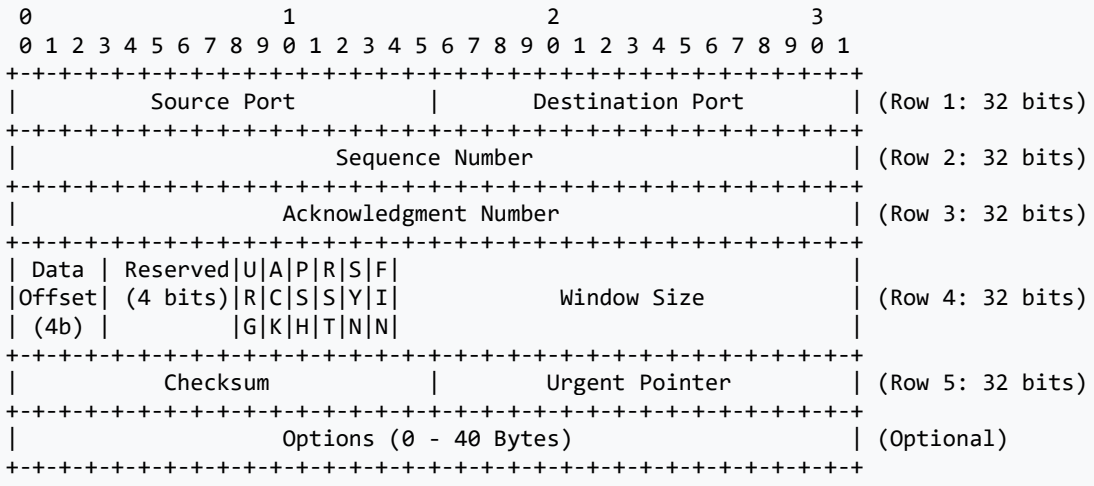

Figure: Standard TCP Header (20 to 60 Bytes)

- **Field Breakdown:**
  - Source Port & Destination Port (16 bits each): Identifies sending and receiving application processes.
  - Sequence Number (32 bits): Specifies the byte sequence number of the first data byte in this segment.
  - Acknowledgment Number (32 bits): Valid if ACK flag is set. Contains the next byte sequence number the receiver expects to receive (cumulative acknowledgment).
  - Data Offset / Header Length (4 bits): Number of 32-bit words in the TCP header (Value 5 = 5 x 4 = 20 bytes; Value 15 = 60 bytes).
  - Control Flags (6 bits): URG (Urgent pointer valid), ACK (Acknowledgment valid), PSH (Push data to app immediately), RST (Reset/abort connection), SYN (Synchronize sequence numbers during connection setup), FIN (Finish/close connection).

- Window Size (16 bits): Flow control receive window (`rwnd`), indicating the number of bytes the receiver can currently buffer.
- Checksum (16 bits): Mandatory 16-bit one's complement sum covering TCP header, data payload, and an IP pseudo-header.
- Urgent Pointer (16 bits): Points to the last byte of high-priority urgent data when URG flag is set.

TCP Connection Establishment: 3-Way Handshake

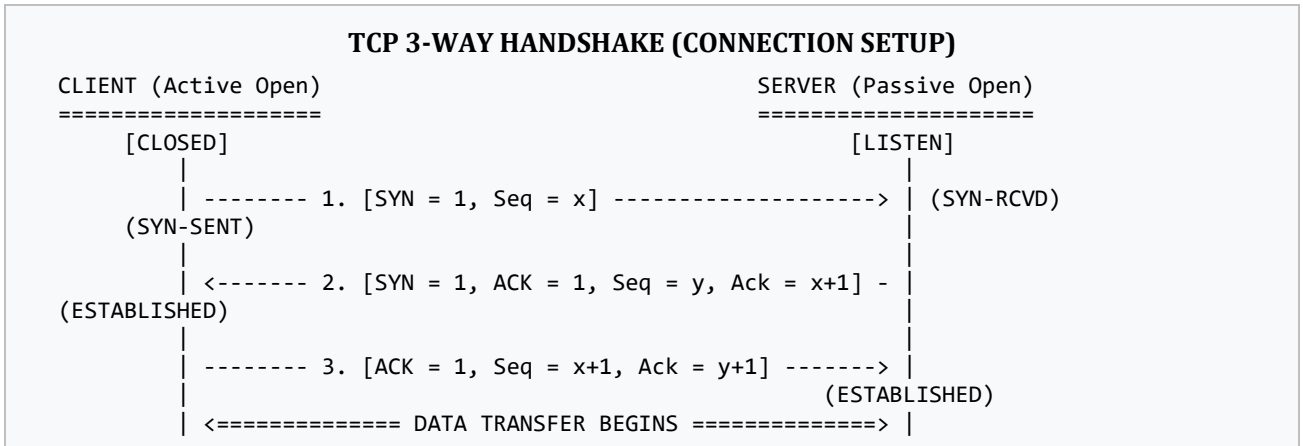

Figure: Three-Way Handshake establishes synchronized sequence numbers and connection state

TCP Connection Teardown: 4-Way Handshake

Because TCP is full-duplex, each simplex direction must be closed independently. Host A sends a FIN segment; Host B acknowledges with an ACK (closing direction A -> B). Host B continues sending remaining data, then transmits its own FIN; Host A sends an ACK and enters the TIME-WAIT state (waiting 2MSL = 2 * Maximum Segment Lifetime, typically 120s) to guarantee the final ACK reached the server before completely closing.

TCP Congestion Control Algorithms (Tahoe / Reno)

TCP employs four interlocking congestion control mechanisms to prevent network thrashing:

- **1. Slow Start:** Initializes Congestion Window `cwnd = 1 MSS` (Maximum Segment Size). For every ACK received, `cwnd` increases by 1 MSS, resulting in Exponential Growth (1 -> 2 -> 4 -> 8 MSS per RTT) until `cwnd` reaches the Slow Start Threshold (`ssthresh`).
- **2. Congestion Avoidance (AIMD):** When `cwnd >= ssthresh`, `cwnd` switches to Linear Growth (Additive Increase: `cwnd = cwnd + 1 MSS` per Round Trip Time RTT) to probe network capacity cautiously.

- **3. Fast Retransmit:** If 3 Duplicate ACKs arrive for a segment, TCP infers that packet loss occurred (without waiting for the slow retransmission timer to expire) and immediately retransmits the missing segment.
- **4. Fast Recovery (TCP Reno):** Sets `ssthresh = cwnd / 2`, sets `cwnd = ssthresh + 3 MSS`, and resumes linear congestion avoidance directly without dropping all the way back to `cwnd = 1`.

3.3 User Datagram Protocol (UDP)

Characteristics & Header Structure

UDP is a minimalist, connectionless, lightweight transport protocol defined in RFC 768. It provides a simple 'fire-and-forget' datagram service with zero connection overhead.

- **UDP Header Format (Only 8 Bytes Fixed Overhead):**
  - Source Port (16 bits): Port number of transmitting process (optional).
  - Destination Port (16 bits): Port number of destination process (mandatory).
  - Length (16 bits): Total length of the UDP header plus data payload in bytes (minimum value = 8).
  - Checksum (16 bits): Optional in IPv4, mandatory in IPv6. Covers header, data, and IP pseudo-header.

Exhaustive Master Comparison: TCP vs. UDP

Comparison Parameter	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Connection Mode	Connection-Oriented (Requires 3-way handshake)	Connectionless (No handshake; fire-and-forget)
Reliability	100% Reliable (Guarantees in-order delivery & ACKs)	Unreliable / Best-Effort (Packets may be lost/reordered)
Header Size	20 to 60 Bytes (Heavy overhead)	8 Bytes Fixed (Minimal overhead)
Transmission Speed	Slower (Due to handshakes, ACKs, flow control)	Extremely Fast (Minimal latency and zero state)
Flow & Congestion Control	Yes (Sliding window, Slow start, AIMD)	No (Transmits at application rate)
Data Transmission Unit	Continuous Byte Stream	Discrete Datagram Packets

Comparison Parameter	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Communication Modes	Unicast Point-to-Point only	Unicast, Multicast, and Broadcast
Typical Real-World Protocols	HTTP, HTTPS, FTP, SSH, SMTP, MySQL	DNS, DHCP, VoIP (RTP), SNMP, Live Streaming, Gaming

3.4 Port Numbers & Socket Programming Concepts

1. Classification of Port Numbers (IANA Standard)

- **1. Well-Known Ports (0 to 1023):** Reserved for standardized system and administrative core protocols. Requires root/admin privileges to bind. Examples: HTTP (80), HTTPS (443), SSH (22), Telnet (23), SMTP (25), DNS (53), DHCP (67/68), TFTP (69), FTP (20/21).
- **2. Registered Ports (1024 to 49151):** Assigned by IANA upon request for specific corporate software products and user processes. Examples: MySQL (3306), Microsoft RDP (3389), PostgreSQL (5432), Redis (6379), Apache Tomcat (8080).
- **3. Dynamic / Private / Ephemeral Ports (49152 to 65535):** Assigned automatically by the client operating system kernel for temporary outbound client connections during communication sessions.

2. The Socket Concept & Client-Server APIs

Definition: Network Socket

A Network Socket is an abstract software endpoint of an end-to-end bidirectional communication channel. It is mathematically represented as the combination of an IP Address and a Port Number: Socket = (IP Address : Port Number). For example: (192.168.1.100 : 443).

TCP Socket Workflow (Stream-Based Connection)

- **Server Workflow:** `socket()` (create socket) -> `bind()` (attach IP & port) -> `listen()` (enter passive listening mode) -> `accept()` (blocks until client connects, creates dedicated connection socket) -> `recv()` / `send()` (exchange data) -> `close()`.
- **Client Workflow:** `socket()` (create socket) -> `connect()` (initiates 3-way handshake) -> `send()` / `recv()` (exchange data) -> `close()`.

3.5 Application Layer Protocols (HTTP, DNS, FTP, SMTP)

1. HTTP and HTTPS (Hypertext Transfer Protocol)

HTTP is the foundational request-response protocol of the World Wide Web. Clients (browsers) send HTTP Requests, and servers respond with HTTP Responses containing resources (HTML, JSON, media).

- **HTTP Evolution:**
  - HTTP/1.1: Introduced Persistent Connections (`Connection: keep-alive`) and Pipelining. Suffers from Head-of-Line (HoL) blocking.
  - HTTP/2: Binary protocol with Multiplexing (streams multiple requests/responses in parallel over a single TCP connection) and HPACK header compression.
  - HTTP/3: Operates over UDP using QUIC transport, eliminating TCP handshake latency and transport Head-of-Line blocking.
- **HTTPS (HTTP Secure - Port 443):** Encapsulates standard HTTP traffic within a cryptographic TLS/SSL (Transport Layer Security) tunnel, guaranteeing Confidentiality (AES encryption), Integrity (HMAC/SHA), and Server Authentication (X.509 Digital Certificates).

2. DNS (Domain Name System - Port 53)

DNS is a globally distributed, hierarchical database that resolves human-readable domain names (e.g., `www.abhyasmitra.in`) into machine-routable IP addresses (e.g., `104.21.45.89`).

- **Hierarchical DNS Tree:** Root Servers ('.') -> Top-Level Domain (TLD) Servers ('.in', '.com', '.edu') -> Authoritative Name Servers (e.g., `ns1.abhyasmitra.in`).
- **Resolution Queries:**
  - Recursive Query: Client requests local DNS resolver to perform the entire resolution and return the final IP answer.
  - Iterative Query: DNS resolver iteratively queries Root -> TLD -> Authoritative server, receiving referral pointers at each step.
- **Common DNS Record Types:**
  - A Record: Maps domain name to 32-bit IPv4 address.
  - AAAA Record: Maps domain name to 128-bit IPv6 address.
  - CNAME (Canonical Name): Alias pointing one domain to another domain name.
  - MX (Mail Exchanger): Specifies the mail server responsible for receiving emails for the domain.
  - PTR (Pointer): Reverse DNS lookup (maps IP address to domain name).

3. FTP (File Transfer Protocol - Ports 20 & 21)

FTP is an interactive client-server protocol designed for transferring bulk files between hosts. FTP is unique because it uses TWO separate TCP connections simultaneously:

- **Control Connection (Port 21):** Carries commands (USER, PASS, RETR, STOR) and status codes. Remains open for the entire session.
- **Data Connection (Port 20):** Opened dynamically strictly to transfer file data or directory listings, and closed immediately after each file transfer completes.
- **Active vs Passive Mode:** In Active Mode, the server connects back to the client's data port (often blocked by client firewalls). In Passive Mode (PASV), the client initiates the data connection to a random server port, bypassing firewall blocks.

4. Electronic Mail Protocols (SMTP, POP3, IMAP)

Email Protocol	Default Port	Type / Direction	Operational Behavior & Characteristics
SMTP (Simple Mail Transfer Protocol)	Port 25 (Server-to-Server) / Port 587 (Submission)	Push Protocol (Sending)	Transfers email from sending client to mail server, and relays mail between intermediate mail servers.
POP3 (Post Office Protocol v3)	Port 110 (Plain) / Port 995 (SSL/TLS)	Pull Protocol (Retrieval)	Downloads emails to local device and deletes them from server by default. Designed for single-device offline access.
IMAP (Internet Message Access Protocol)	Port 143 (Plain) / Port 993 (SSL/TLS)	Pull Protocol (Retrieval)	Keeps all emails and folder directories synchronized on the server. Allows seamless multi-device access (Phone + Laptop).

3.6 High-Yield University Exam Question Bank

Part A: Short Answer Questions (2 to 5 Marks)

Q1. Explain the significance of Port Numbers and list three well-known ports. [3 Marks]

- **Answer:** Port numbers (16 bits, 0-65535) provide process-to-process addressing at the Transport layer, identifying specific software applications. Well-known ports: HTTP (80), HTTPS (443), DNS (53).

Q2. Why does FTP use two separate connections? [3 Marks]

- **Answer:** FTP separates control commands from actual data payload (Out-of-Band control). Control Connection (Port 21) transmits user authentication and commands throughout the session; Data

Connection (Port 20) opens dynamically exclusively to transfer file bytes and closes immediately upon completion.

Q3. Differentiate between POP3 and IMAP. [4 Marks]

- **Answer:** POP3 downloads mail to a local client and removes it from the server (designed for single device offline use). IMAP synchronizes folders and maintains all messages on the server, supporting seamless multi-device access.

Part B: Long Answer Questions (10 to 15 Marks Outline Guides)

- **Q4. Explain the TCP 3-Way Handshake, TCP Segment Header Format, and TCP Connection Teardown in detail with diagrams. [15 Marks]**
- **Q5. Compare TCP and UDP in detail with respect to connection state, reliability, header fields, flow control, and suitable real-world applications. [10 Marks]**
- **Q6. Discuss TCP Congestion Control mechanisms: Slow Start, Congestion Avoidance (AIMD), Fast Retransmit, and Fast Recovery with cwnd progression graphs. [12 Marks]**
- **Q7. Explain the Domain Name System (DNS) architecture, DNS record types, and the complete step-by-step domain name resolution process (Iterative vs Recursive queries). [10 Marks]**

3.7 Unit Summary & Key Takeaways

- Transport layer provides process-to-process delivery, port addressing (0-65535), segmentation, multiplexing, and flow/error control.
- TCP is connection-oriented, reliable (3-way handshake, sequence numbers, checksums), flow-controlled (rwnd), and congestion-controlled (Slow start, AIMD).
- UDP is connectionless, unreliable, lightweight (8-byte header), and prioritized for real-time applications (VoIP, DNS, gaming).
- A Socket is the endpoint of communication: Socket = (IP Address : Port Number).
- Application protocols: HTTP/HTTPS (Web, TLS/SSL), DNS (Name resolution, A/MX records), FTP (Dual ports 20/21), SMTP/POP3/IMAP (Email push & pull).

© Copyright — abhyasmitra.in — All Rights Reserved