Chapter 2 - Notes - CS 356

Chapter 2: Application Layer

2.1 Principles of Network Applications

- **Architectures**:
  - **Client-Server**:
    - Central server, permanent IP, always-on.
    - Examples: Web, email.
  - **Peer-to-Peer (P2P)**:
    - End systems (peers) act as both client and server.
    - Self-scalable but complex.
    - Example: BitTorrent.
- **Process Communication**:
  - Client process = initiates.
  - Server process = waits.
  - **Socket** = interface for sending/receiving messages.
- **Addressing**:
  - IP address + port number (e.g., HTTP → port 80, SMTP → port 25).

2.2 Transport Services for Applications

- **Requirements**:
  - **Reliability**: File transfer requires 100% reliability.
  - **Timing**: Real-time voice/video requires low latency.
  - **Throughput**: Streaming apps require guaranteed minimum.
  - **Security**: TLS for encryption and integrity.

- **Protocols**:
    - **TCP**:
        - Connection-oriented, reliable.
        - Flow control + congestion control.
    - **UDP**:
        - Connectionless, unreliable, lightweight.
        - Used in VoIP, streaming.

2.3 HTTP (HyperText Transfer Protocol)

- **Overview**:
    - Application-layer protocol for the web.
    - Client-server model.
    - Uses TCP (port 80).
    - Stateless protocol.
- **Connection types**:
    - **Non-persistent**: one TCP connection per object.
    - **Persistent (HTTP/1.1)**: multiple objects over one connection.
- **Message Types**:
    - **Request**: GET, POST, HEAD, PUT.
    - **Response**: Status codes (200 OK, 301 Moved, 404 Not Found).
- **Cookies**:
    - Maintain state between client-server.
    - Stored on client + backend database.
- **Web Caching (Proxy servers)**:
    - Reduce delay + bandwidth usage.
- **Modern HTTP**:

  - **HTTP/2**: Multiplexing, prioritization, reduced HOL blocking.
  - **HTTP/3**: Runs over UDP with QUIC, adds encryption + congestion control.

2.4 Email

- **Components**:
  - **User agent (UA)**: mail client (e.g., Outlook).
  - **Mail servers**: store & forward.
  - **SMTP**: Simple Mail Transfer Protocol for sending.
- **Protocols**:
  - **SMTP**: TCP port 25, ASCII commands.
  - **IMAP/POP3**: retrieving mail.
- **Example flow**:
  - Alice → her mail server → Bob's mail server → Bob's UA.

2.5 DNS (Domain Name System)

- **Purpose**: Maps domain names to IP addresses.
- **Structure**:
  - Hierarchical:
    - **Root servers**.
    - **TLD servers** (.com, .edu).
    - **Authoritative servers**.
- **Resource Records (RR)**:
  - `A`: maps hostname → IP.
  - `NS`: name server.
  - `MX`: mail exchange.
- **Caching**:

  - Local DNS servers cache responses (can lead to stale data).

2.6 P2P Applications

- **Characteristics**:
  - Decentralized.
  - Peers provide and request services.
  - Self-scalable.
- **Challenges**:
  - Peer churn.
  - Search and lookup.
  - Security concerns.

2.7 Video Streaming & CDNs

- **Streaming**:
  - Requires continuous delivery.
  - Uses **buffering** + adaptive bitrate streaming.
- **DASH (Dynamic Adaptive Streaming over HTTP)**:
  - Video divided into chunks.
  - Client requests chunks adaptively based on bandwidth.
- **CDNs (Content Delivery Networks)**:
  - Distributed caching servers.
  - Reduce latency, improve reliability.
