Network Protocol Basics

1. Client-Server Model

Definition: A centralized model where the server hosts services/resources, and clients request them.

Flow: Client → Request → Server → Response → Client

Examples:

- Web applications (HTTP)
- Email servers (SMTP)

Pros: Easy to manage, secure, centralized control.

Cons: Single point of failure, server load can be high.

2. Peer-to-Peer (P2P) Model

Definition: Each node (peer) can act as both a client and a server.

Flow: Peer ↔ Peer (Direct Communication)

Examples:

- File-sharing networks (BitTorrent)
- WebRTC video calls

Pros: Scalable, no central server bottleneck.

Cons: Security and coordination challenges.

3. WebSockets

Purpose: Real-time, **bidirectional** communication between client and server.

How it works:

- Client sends **HTTP handshake** → Server upgrades connection → **Full-duplex** channel created.

Use cases: Chat apps, notifications, live dashboards.

Difference from HTTP:

- HTTP: Request/Response → One-time connection.

- WebSocket: Persistent connection → Continuous data flow.

4. HTTP vs TCP vs UDP

Feature	HTTP	TCP	UDP
Type	Application Layer	Transport Layer	Transport Layer
Connection	Request-Response	Connection-oriented	Connectionless
Reliability	Depends on TCP	Reliable, Ordered	Unreliable, Fast
Use Cases	Web pages, APIs	File transfer, APIs	Live streaming, Gaming

HTTP uses TCP underneath → Reliable communication.

UDP used where speed > reliability (e.g., Video Calls).

5. FTP (File Transfer Protocol)

Used for transferring files between systems.

Operates over **TCP**.

Two Modes: **Active** and **Passive**.

Secure alternative: **SFTP (SSH File Transfer Protocol)**.

6. SMTP, POP, and IMAP

SMTP (Simple Mail Transfer Protocol) → For **sending emails** (Client → Server → Server → Client).

POP3 (Post Office Protocol) → Downloads email to local client, then deletes from server.

IMAP (Internet Message Access Protocol) → Keeps emails on the server, allows sync across devices.

7. Key HLD Takeaways

- **HTTP + HTTPS** → Web communication & APIs.
- **WebSocket** → Real-time chat & notifications.
- **WebRTC + UDP** → Video/Audio calls (P2P).
- **SMTP + IMAP/POP3** → Email systems.
- **TCP vs UDP** → Trade-off: Reliability vs Speed.

- **FTP/SFTP** → File sharing in enterprises.