🌐 Protocols in Peer-to-Peer (P2P) Systems

Unlike the **Client–Server model**, where communication is mostly **request** → **response**, P2P protocols are designed for **discovery, sharing, and coordination among equals**.

🔹 1. Application Layer Protocols (Main P2P Protocols)

- **BitTorrent Protocol**
  - Most popular P2P file-sharing protocol.
  - Splits files into chunks → peers download and upload chunks simultaneously.
  - Uses **trackers** or **DHT (Distributed Hash Table)** for peer discovery.
- **Gnutella Protocol**
  - Early decentralized file-sharing system.
  - Query flooding: a peer broadcasts search requests to neighbors.
- **eDonkey / eMule Protocol**
  - Uses both central servers (for indexing) + P2P sharing.
- **Kademlia (DHT Protocol)**
  - Used in BitTorrent, Ethereum, IPFS.
  - Peers store key–value pairs, enabling efficient **peer and resource lookup**.
- **IPFS (InterPlanetary File System)**
  - Content-addressed protocol → files identified by their cryptographic hash.

    - Built on top of DHT (Kademlia).

- **Blockchain P2P Protocols (Bitcoin, Ethereum, etc.)**
    - Each node shares blocks/transactions with others.
    - Gossip protocol → Information spreads like word of mouth.

🔹 2. Transport Layer Protocols

- **TCP (Transmission Control Protocol)**
    - Reliable, ordered communication.
    - Used by BitTorrent, IPFS, Bitcoin, etc.
- **UDP (User Datagram Protocol)**
    - Lightweight, faster but less reliable.
    - Used for peer discovery (DHT queries, trackers) and real-time P2P apps (VoIP, gaming).
- **QUIC (Quick UDP Internet Connections)**
    - Used in some modern P2P apps → combines UDP speed + TLS security.

🔹 3. Discovery & Routing Protocols

- **DHT (Distributed Hash Table)**
    - Peers organize into a virtual ring/mesh.
    - Each peer stores part of the keyspace → efficient lookup of files/peers.
    - Examples: Kademlia, Chord, Pastry, CAN.

- **Gossip Protocols**
  - Each peer randomly shares information with a few others.
  - Ensures eventual consistency across the network.
  - Used in blockchain, distributed databases (Cassandra).

🔹 4. Security Protocols

- **TLS/SSL (Transport Layer Security)**
  - Adds encryption between peers (used in modern P2P apps).
- **End-to-End Encryption**
  - Messaging apps (Signal, WhatsApp P2P before server relay) use it.
- **Public Key Cryptography**
  - Used in blockchain (digital signatures, wallet addresses).
  - Ensures trust without central authority.

🔹 5. Example: BitTorrent Workflow

1. Peer wants a file → Downloads a `.torrent` file (metadata).
2. Uses **tracker protocol (HTTP/UDP)** to find peers OR queries **DHT**.
3. Establishes **TCP/UDP connections** with peers.
4. Exchanges file chunks using **BitTorrent protocol**.
5. Security may use **TLS** if encrypted connections are enabled.

🔹 6. Example: Blockchain (Bitcoin) Workflow

1. New transaction is created and signed with **public/private key cryptography**.
2. Shared using **gossip protocol**.
3. Peers validate and propagate transactions.
4. Blocks are distributed using **TCP over P2P protocol**.
5. Consensus algorithms (Proof of Work, Proof of Stake) ensure integrity.

✅ Summary

In **P2P systems**, multiple protocols work together:

- **Application Layer** → Defines *how peers share/discover resources* (BitTorrent, IPFS, Gnutella, Blockchain protocols).
- **Transport Layer** → Defines *how data is sent reliably or quickly* (TCP, UDP, QUIC).
- **Discovery & Routing** → Defines *how peers find each other* (DHT, Gossip).
- **Security Layer** → Ensures *trust and privacy without central server* (TLS, PKI, encryption).