Computer Network Interview Questions

1. What are the common models of computer network protocols? What does each layer do?

The common models are TCP/IP model and OSI model.

TCP/IP Model

- Physical Layer: Manages the hardware connections and raw bit transmission over a physical medium.
- Data Link Layer: Ensures reliable data transfer between directly connected nodes and provides error detection/correction.
- Network Layer: Handles packet forwarding, including routing, through different nodes in a network.
- Transport Layer: Provides end-to-end communication, reliability, and flow control. Includes protocols like TCP and UDP.
- Application Layer: Where high-level protocols operate, enabling services such as HTTP, FTP, and DNS.

Purpose: The TCP/IP model is simpler and widely used in teaching and practice for understanding basic networking principles, particularly in the context of the internet.

OSI Model

- Physical Layer: Similar to the TCP/IP model, it manages physical connections.
- Data Link Layer: Similar as well, focusing on node-to-node data transfer.
- Network Layer: Manages routing and forwarding.
- Transport Layer: Ensures reliable data transfer and flow control.
- Session Layer: Manages sessions or connections between applications. This includes establishing, maintaining, and terminating sessions.
- Presentation Layer: Ensures data is in a usable format, handling tasks like encryption, decryption, and data compression.
- Application Layer: Provides services and interfaces for end-user applications.

Purpose: The OSI model is used for conceptual clarity, providing a more granular breakdown of functions compared to the TCP/IP model. It is not as commonly used in practice but is foundational in theory and for troubleshooting complex issues.

2. What are the difference of TCP, UDP, and QUIC protocols? What are their respective pros and cons?

TCP (Transmission Control Protocol)

Features

- Connection-oriented: Establishes a connection between sender and receiver before data is transferred (three-way handshake).
- Reliable: Ensures data integrity by using acknowledgment packets (ACK) and retransmitting lost packets.
- Ordered delivery: Data is delivered in the order it was sent.
- Flow control: Regulates the amount of data sent to prevent congestion.

Pros

- Reliable data delivery: Guaranteed delivery of data (in order and without loss).
- Error correction: Ensures error-free data transfer by retransmitting lost or corrupted packets.
- Congestion control: Manages congestion in the network to avoid packet loss and delays.

Cons

- Overhead: The connection setup, acknowledgments, and retransmissions introduce latency and overhead.
- Slower for real-time applications: The reliability mechanisms, like retransmission, can introduce delays, making TCP unsuitable for time-sensitive applications.
- Not ideal for high-volume traffic: Requires more resources for maintaining the connection, making it less efficient for certain types of traffic.

Use Cases in CDN Systems

- Web Traffic (HTTP/HTTPS): Most CDN systems use TCP for delivering web content (like HTML, CSS, and JavaScript), as it ensures reliable and ordered delivery.
- File Transfers: TCP is used for transferring files like images, videos, and large documents, where reliable delivery is crucial.

UDP (User Datagram Protocol)

Features

- Connectionless: No connection is established before sending data.
- Unreliable: No guarantees of data delivery, order, or error correction.
- Low overhead: There are no acknowledgments or retransmissions, making it lightweight.

Pros

- Low latency: Because it does not require a connection setup or acknowledgment, UDP has very low latency, which is important for real-time applications.
- Efficient for streaming: With less overhead, UDP is well-suited for applications like live video streaming or online gaming.
- Broadcast and multicast: UDP supports sending data to multiple recipients at once, useful for broadcasting.

Cons

- Unreliable: Data loss is common, and there’s no way to detect or recover from it.
- No congestion control: UDP can contribute to congestion in the network since it doesn't handle flow control or retransmissions.
- Out-of-order packets: UDP does not guarantee the order of packets, which can be problematic for some applications.

Use Cases in CDN Systems

- Live Streaming: For live video or audio streaming, where low latency is critical, and minor packet loss is acceptable.
- Real-time Gaming: Multiplayer games use UDP to minimize latency and ensure real-time interactions between players.
- DNS Queries: UDP is used for DNS resolution because it's fast and doesn't require a reliable connection.

QUIC (Quick UDP Internet Connections)

Features

- Built on UDP: QUIC uses UDP as its transport protocol but adds features from TCP and TLS for reliability and security.
- Connection multiplexing: QUIC supports multiplexing multiple streams over a single connection, avoiding head-of-line blocking (a problem in TCP).
- Zero round-trip time (0-RTT) connection establishment: Allows faster connection establishment by resuming sessions from previous connections.
- Encryption: QUIC encrypts the entire transport layer, providing better security.

Pros

- Low latency: QUIC can establish connections faster than TCP and reduce latency with features like 0-RTT connection establishment.

- Improved reliability: Combines the best aspects of TCP and UDP, offering reliable delivery while being faster and more efficient.
- Multiplexing: QUIC supports multiplexing, allowing multiple streams over a single connection without the head-of-line blocking problem.
- Built-in encryption: QUIC encrypts the entire connection, improving security compared to traditional protocols like TCP.

Cons

- Newer protocol: QUIC is relatively new and might not be fully supported in all environments or across older network infrastructures.
- Higher complexity: QUIC's multiplexing and encryption can introduce complexity for both implementation and debugging.
- Not widely adopted (yet): Although it's gaining traction (especially with HTTP/3), QUIC is not as universally deployed as TCP or UDP.

Use Cases in CDN Systems

- Web Traffic (HTTP/3): QUIC is the basis for HTTP/3, which is being adopted for faster, more efficient web browsing with lower latency. CDNs are beginning to use QUIC to improve performance for HTTPS-based web traffic.
- Improved Streaming: QUIC is used for streaming applications that require fast, reliable delivery with low latency, such as video conferencing or real-time media streaming.
- Mobile Applications: QUIC can significantly reduce connection setup time, making it ideal for mobile applications that require low latency and efficient resource usage.

Comparison of Use Cases in CDN Systems

Protocol Best for Use Case in CDN
TCP Reliable data transfer Web traffic (HTTP/HTTPS), file downloads, secure transactions
UDP Low latency, real-time applications Live streaming, online gaming, DNS queries, real-time voice/video
QUIC Low latency, secure, multiplexing HTTP/3 for web traffic, improved video streaming, mobile app performance

3. Describe the process of three-way handshake and four-way handwave in TCP

The Three-Way Handshake is a process used in TCP (Transmission Control Protocol) to establish a reliable connection between a client and a server. This process ensures that both parties are synchronized before data transfer begins.

Three-Way Handshake (TCP Connection Establishment)

The Three-Way Handshake ensures that both the client and server are ready to communicate and have synchronized their sequence numbers for reliable data transmission.

1. SYN (Synchronize)
   The client sends a SYN packet to the server to initiate the connection. This packet includes an initial sequence number (ISN) that represents the starting point of the sequence of data that will be transmitted.
2. SYN-ACK (Synchronize-Acknowledge)
   The server responds to the client with a SYN-ACK packet. This packet acknowledges the client's SYN (by setting the ACK flag) and includes the server's own initial sequence number.
3. ACK (Acknowledge)
   The client sends an ACK packet to the server, acknowledging the server's SYN-ACK. At this point, the connection is fully established, and data transfer can begin.

Visual Representation

Client -> [SYN] -> Server
Client <- [SYN, ACK] <- Client
Client -> [ACK] -> Server

Four-Way Wavehand (TCP Connection Termination)

Four-Way Wavehand is the process used to terminate a TCP connection. It ensures that both sides of the connection are gracefully closed.

1. FIN (Finish)
   The client sends a FIN packet to the server, indicating that the client has finished sending data. This tells the server that it should stop accepting data from the client.
2. ACK (Acknowledge)
   The server responds to the client's FIN with an ACK packet, acknowledging the receipt of the FIN.
3. FIN (Finish)
   The server then sends its own FIN packet to the client, indicating that it has finished sending data and is ready to close the connection.
4. ACK (Acknowledge)
   The client sends an ACK packet to the server, acknowledging the server's FIN. At this point, the connection is fully terminated, and both sides can release resources.

Visual Representation

Client -> [FIN] -> Server
Client <- [ACK] <- Server

Client <- [FIN] <- Server
Client -> [ACK] -> Server

Summary:
• The Three-Way Handshake establishes a connection between the client and server using SYN and ACK packets.
• The Four-Way Handshake (or Wavehand) terminates the connection by exchanging FIN and ACK packets to ensure both sides gracefully close the connection without losing any data.