☆ Application Layer (OSI Model)

The **Application Layer** is the **7th and top layer** of the OSI model.
It is the **layer closest to the user**, meaning when you use **websites, emails, file sharing**, etc., you are actually using the **Application Layer**. Acts as the interface between the user and the network.

Functions of Application Layer

1. Data Representation

- Converts data into a format the sender and receiver both understand.
- Makes sure text, images, files, videos can be sent over the network.
- Converts user data into standard formats before transmission.

Example:
Text → ASCII
Image → JPEG
Webpage → HTML

2.**Network Service Access**

  - Provides direct access to network services for users.
  - Example: Email (Outlook, Gmail), FTP for file transfer, remote login

3. Application Protocols

- Defines rules for communication between applications
- Handles message structure, data exchange, requests.

Example protocols:

- **HTTP** → web browsing
- **FTP** → file transfer
- **SMTP** → sending emails
- **DNS** → convert domain name to IP

4. Session Management

- Creates and manages sessions between two devices.
- Maintains connection until communication is complete.
- Closes it properly after use.

Example:
Login into remote server using **SSH/Telnet**.

How Application Layer Works

1. **Client sends a command** to server (like HTTP request).
2. Server listens → **assigns a port number** to the client.
3. Client sends a **connection request**.
4. Server sends **ACK (acknowledgement)** → connection established.
5. Now the client can:
    - Request files
    - Upload documents
    - Access server resources

After work is done → **session is closed**.

Protocol	Full Form	Use	Port
HTTP	HyperText Transfer Protocol	Web browsing	80
DNS	Domain Name System	Converts domain to IP	53
TELNET	Remote Terminal	Remote login	23

Protocol	Full Form	Use	Port
DHCP	Dynamic Host Configuration Protocol	Assign IP automatically	67, 68
FTP	File Transfer Protocol	File uploading/downloading	20 (data), 21 (control)
SMTP	Simple Mail Transfer Protocol	Sending emails	25, 587
NFS	Network File System	Remote file access	2049
SNMP	Simple Network Management Protocol	Manage network devices	161, 162

⭐ Presentation Layer (OSI Model)

Presentation Layer is the 6th layer of the OSI model.

It is called the Translation Layer or Syntax Layer, because this layer converts data into an “understandable” format.

The main functions of this layer:

👉 When data is sent ⟶ Format, encrypt, compress and send it

👉 When data arrives ⟶ Decrypt, decompress, translate and make it user-friendly.

Functions of Presentation Layer

1. Data Translation

The Presentation layer converts data from one format to another so both devices understand it.
Example: ASCII → EBCDIC, JPEG → BMP.

2. Data Compression

It reduces the size of data so it can travel faster through the network.
Example: ZIP files, MP3 compression.

3. Data Encryption & Decryption

It keeps data secure during transmission.

- Sender → encrypts the data
- Receiver → decrypts the data
  Example: SSL/TLS security.

4. Syntax and Semantics Management

It ensures the **structure, order, and meaning** of data are correct and understandable.

5. Transfer Syntax Negotiation

Both systems agree on **which format (syntax)** will be used to exchange data.

6. Interoperability

It allows different devices, operating systems, and applications to communicate smoothly.

Working of the Presentation Layer (Step-by-Step)

Sender Side:

1. Takes data from Application Layer.
2. Translates/Formats the data.
3. Encrypts it for security.

4. Compresses it to reduce size.
5. Sends it to Session Layer.

Receiver Side:

1. Receives data from Session Layer.
2. Decrypts it.
3. Decompresses it.
4. Translates/Formats it into readable form.
5. Passes it to Application Layer.

Presentation Layer Attacks

1. Man-in-the-Middle (MITM)

An attacker secretly sits between two devices and steals or changes the data.

Example:
You send your password to a website → attacker intercepts it before it reaches the server

2. SSL/TLS Downgrade Attack

The attacker forces the system to use weak or old encryption so the data becomes easier to hack.

Example:
Your browser supports strong HTTPS, but the attacker tricks it to use an older, weaker version → they can read your messages.

3. Certificate Spoofing

An attacker uses a fake certificate to look like a trusted website/server and tricks the user.

Example:
You think you're opening **https://yourbank.com**, but the attacker shows a fake certificate and you enter your bank details on a fake site.

4. Code Injection

The attacker inserts harmful code during data formatting or processing.

Example:
When an app formats your data, the attacker injects a harmful script that gives them access to your system.

Core Services of the Presentation Layer

1. Data Translation

- Converts data from one format to another so the receiver can understand it.
- **Example:** ASCII → EBCDIC.
    - Sender uses ASCII, but receiver supports EBCDIC → presentation layer translates it.

2. Data Compression

- Reduces data size for **faster transmission** and **better bandwidth use**.
- **Types:**
    - **Lossless:** No data lost (e.g., ZIP for text)
    - **Lossy:** Some data lost for smaller size (e.g., JPEG for images)
- **Example:** Large image is compressed before sending and decompressed by the receiver.

3. Data Encryption & Decryption

- Keeps data **secure** during transmission.
- **Encryption:** Turns readable data into secret code (ciphertext).
- **Decryption:** Converts it back to readable form at the receiver.
- **Example:** SSL/TLS protects passwords and banking data.

4. Syntax & Semantics Management

- Makes sure data is **structured correctly** (syntax) and **meaningful** (semantics).
- **Example:** Converting XML → JSON so apps understand the data correctly.

Limitations of the Presentation Layer

- **Limited error handling:** Relies on Transport Layer.
- **Protocol dependency:** Fails if protocols or formats are incompatible.
- **Processing overhead:** Extra work due to translation, encryption, and compression.
- **No direct control of data transfer:** Depends on lower layers.
- **Protocol constraints:** Works only with supported standards like SSL/TLS, MIME, etc.

Session Layer (Layer 5)

The **Session Layer** is the **5th layer of the OSI model**.
It **manages communication sessions** between devices, ensuring smooth and organized data exchange. Think of it as the **manager of a conversation** between two computers.

Main Functions of the Session Layer

1. **Session Establishment**
    - Creates a "session" (logical connection) between two devices.
    - **Types of session mappings:**
        - **One-to-One:** 1 session → 1 transport connection
        - **Many-to-One:** Multiple sessions share 1 transport connection
        - **One-to-Many:** 1 session uses multiple transport connections
    - When the session ends, the transport connection also ends.
2. **Communication Synchronization**
    - Keeps the communication **organized and recoverable**.

- Uses **checkpoints**:
  - Marks places in data so if something fails, you don't have to start from the beginning.
- Essential for **long or complex data transfers**.

3. **Activity Management**
   - Splits continuous data flow into **logical tasks called activities**.
   - Each activity is independent → easier to manage different tasks in a single session.
4. **Dialog Management**
   - Controls **who sends data and when**.
   - Two modes:
     - **Half-duplex:** Only one device sends at a time (uses token system).
     - **Full-duplex:** Both devices send data at the same time.
   - Helps **prevent data collisions** and improves efficiency.
5. **Data Transfer**
   - Manages the actual **sending and receiving of data**.
   - Respects communication mode:
     - Half-duplex → one sends, other listens
     - Full-duplex → both send and receive simultaneously
   - Ensures **reliable and organized data exchange**.
6. **Resynchronization**
   - If a session fails, it can **restore the session without starting over**.
   - Methods:
     - **Set:** Start a new sync point

- **Abandon:** Discard the old sync point
- **Restart:** Use a new sync point after the last major acknowledged one

How the Session Layer Works

1.Session Parameters: Sets rules for who talks and how data is sent/received.

2.Token Control: Only the device with the token can send data.

3. Checkpoints: Allows resuming from the last checkpoint if something goes wrong.

4.Data Integrity: Ensures data is complete and not duplicated.

5.Session Termination: Properly closes the session after all data is exchanged.

Protocols at the Session Layer

1. ADSP (AppleTalk Data Stream Protocol) – For communication in Apple LAN networks.
2. RTCP (Real-time Transport Control Protocol) – Checks quality of multimedia streams (like video/audio).
3. PPTP (Point-to-Point Tunneling Protocol) – Helps create VPN connections over the internet.
4. PAP (Password Authentication Protocol) – Verifies user login via password.
5. RPC (Remote Procedure Call Protocol) – Lets a program run commands on another computer.
6. SDP (Sockets Direct Protocol) – Helps fast communication over special networks (RDMA).

☆ Transport Layer (4)

The **Transport Layer** is the **4th layer** of the OSI model.
Its main goal is **end-to-end delivery of data between applications** on two different computers. Think of it like a **delivery service** that delivers parcels (data) safely, in order, and without loss.

Main Functions of Transport Layer

1. **End-to-end communication**
   Connects one application on a computer to another application on another computer.
2. **Process-to-process delivery**
   Uses port numbers (like address numbers) to send data to the correct application.
3. **Segmentation & Reassembly**
   Breaks big data into segments (TCP) or datagrams (UDP), and joins them again at receiving side.
4. **Error control**
   Detects errors, retransmits lost data, and ensures correct delivery.
5. **Flow control**
   Makes sure the sender does not send too fast for the receiver.
6. **Reliable or unreliable delivery**
   - TCP → reliable
   - UDP → fast but not reliable
7. **Sequencing**
   Ensures data arrives in the correct order.
8. **Multiplexing & Demultiplexing**
   Allows many applications to use the network at the same time.

How Transport Layer Works

- Works only in end devices, not routers.
- Uses port numbers to identify apps (Example: HTTP uses port 80).
- Divides data into segments/datagrams.
- Adds header (port numbers, sequence numbers, checksum).
- Sends it to the Network Layer for routing.
- At the receiving side, segments are checked, reordered, and given to the correct application.

TCP 3-Way Handshake Process

The TCP 3-Way Handshake is a process used by the Transmission Control Protocol (TCP) to establish a reliable connection between a client and a server before data transfer. It ensures that both sides are synchronized and ready to communicate.

TCP 3-Way Handshake Steps

Step 1: SYN

Client → Server
Client sends SYN to say:
“Hi, I want to start. Here is my starting sequence number.”

Step 2: SYN + ACK

Server → Client
Server replies with SYN + ACK:
“I received your SYN. Here is my starting sequence number.”

Step 3: ACK

Client → Server
Client sends ACK:
“I received your SYN. Connection ready!”

Now the **connection is established**, and **data transfer can begin**.

✓ Why 3 steps?

1. Both sides must confirm that the other side is ready.
2. Both sides need to exchange and synchronize sequence numbers.
3. In only 2 steps, both sides cannot be fully sure — so 3 steps are required.

#Transmission Control Protocol (TCP)

What is TCP?

- TCP is a connection-oriented protocol used for communication between devices over a network.
- Works at Transport Layer (Layer 4) of the OSI model.
- Ensures reliable, ordered, and error-free delivery of data.
- Works together with IP (Internet Protocol) which handles addressing and routing.

How TCP Works

1.Connection Establishment:

Uses 3-way handshake (SYN → SYN-ACK → ACK) to start communication.

2.Data Transmission:

Divides data into segments and sends them to the receiver.

3.Reassembly: Receiver reassembles the segments in correct order using sequence numbers.

4.Acknowledgment (ACK): Receiver confirms receipt of each segment.

5.Error & Flow Control:

- Checksum detects errors.
- Sliding window controls flow to avoid overwhelming the receiver.

6.Connection Termination: Uses 4-step handshake (FIN → ACK → FIN → ACK) to close connection properly.

Features of TCP (Transmission Control Protocol)

1. **Segment Numbering**
   - Every byte of data is numbered.
   - The receiver uses these numbers to reassemble data in order.
   - Acknowledgments (ACKs) confirm receipt of data.
2. **Connection-Oriented**
   - TCP establishes a connection before transmitting data (3-way handshake).
   - The connection stays active until all data is successfully transferred.
3. **Full Duplex Communication**
   - Data can flow in both directions simultaneously.
   - Both sender and receiver can send and receive at the same time.
4. **Flow Control**
   - Prevents the sender from overwhelming the receiver.
   - Uses mechanisms like sliding window to adjust transmission rate.
5. **Error Control**
   - Detects lost, corrupted, duplicated, or out-of-order segments.
   - Retransmits data when errors are detected to ensure reliability.

6. **Congestion Control**
    - Monitors network congestion and adjusts the sending rate.
    - Prevents network overload by reducing transmission speed when necessary.

Advantages of TCP

- Reliable and error-checked delivery.
- Ensures data arrives in correct order.
- Widely used and standardized by IETF.
- Works with IP to connect devices across networks.

Disadvantages of TCP

- Adds extra overhead → can slow down small networks.
- Not suitable for non-TCP/IP protocols (e.g., Bluetooth).
- Complex protocol → consumes more resources.

Services Provided by TCP

1. **Process-to-Process Communication**
    - Uses 16-bit port numbers to identify sending and receiving processes.
    - Ensures data reaches the correct application on the destination host.
2. **Stream-Oriented**
    - Data is sent as a continuous stream of bytes.
    - Grouped into **segments** with TCP headers before being encapsulated in IP packets.
3. **Full-Duplex Service**

- Communication can occur **simultaneously in both directions** between sender and receiver.

4. **Connection-Oriented Service**
    - TCP establishes a **reliable connection** before data transfer.
    - **Phases of connection**:
        1. Connection establishment (3-way handshake)
        2. Data transfer
        3. Connection termination
5. **Reliability**
    - Ensures data integrity using:
        - Checksums
        - Sequence numbers
        - Acknowledgements (ACKs)
        - Retransmissions for lost or corrupted data
        - Congestion control
6. **Multiplexing**
    - Allows multiple logical connections over a single physical connection.
    - Achieved through port numbers at sender and receiver ends

TCP Connection Establishment (3-Way Handshake)

Purpose:

- Establish a reliable connection between sender and receiver before data transfer.
- Both sides synchronize sequence numbers and agree on parameters like window size and maximum segment size (MSS).

Steps of 3-Way Handshake

Step 1: Sender → Receiver (SYN)

- **Seq = 521** → initial sequence number of sender
- **SYN = 1** → request to synchronize sequence numbers
- **MSS = 1460 B** → maximum segment size sender can receive
- **Window = 14600 B** → sender's buffer size

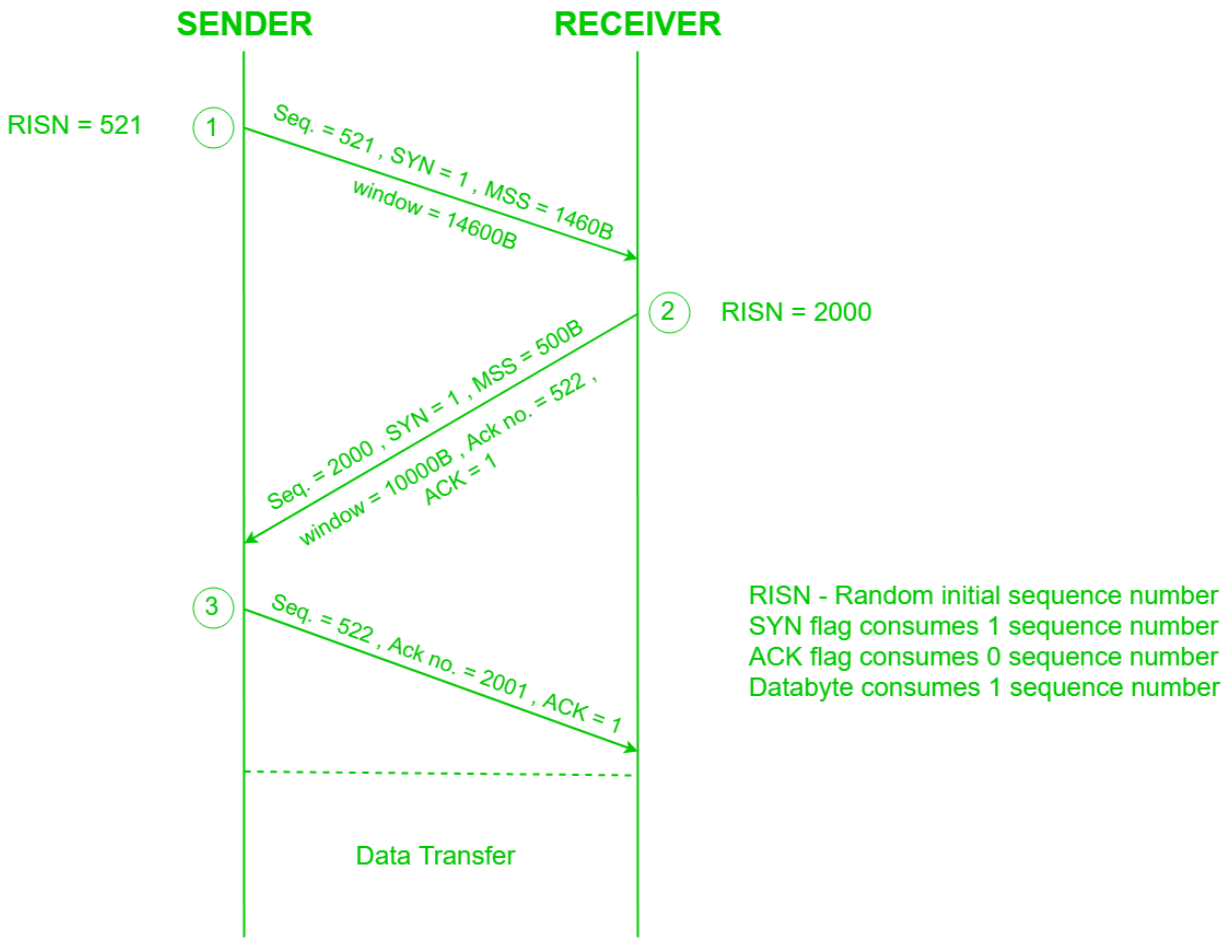
Step 2: Receiver → Sender (SYN + ACK)

- **Seq = 2000** → initial sequence number of receiver
- **SYN = 1** → request to synchronize sequence numbers
- **ACK = 1, Ack no = 522** → acknowledges sender's SYN (522 = 521+1)
- **MSS = 500 B** → receiver's maximum segment size
- **Window = 10000 B** → receiver's buffer size

Step 3: Sender → Receiver (ACK)

- **Seq = 522** → next sequence number
- **ACK = 1, Ack no = 2001** → acknowledges receiver's SYN (2001 = 2000+1)

Result: Connection established and ready for data transfer.



#TCP Connection Termination

TCP closes a connection **in an orderly way** using a **four-step handshake** to make sure both sides finish sending and receiving data.

Types of Connection Release

1. **Abrupt Release**
    - Connection is closed immediately using **RST (Reset)** flag.
    - Happens if:
        - Segment received for a non-existing connection.
        - Invalid header detected.
        - Lack of resources or remote host unreachable.

2. **Graceful Release**
    - Connection is closed **step by step** using **FIN flag**.
    - Each side closes its connection individually.

TCP 4-Way Handshake (Connection Termination)

When a TCP connection ends, it uses **4 steps** to close reliably.

Steps

1. **FIN (Client → Server)**
    - Client says: “I’m done sending data.”
    - FIN = Finish sending.
2. **ACK (Server → Client)**
    - Server replies: “I got your FIN.”
    - Confirms client’s request to close.
3. **FIN (Server → Client)**
    - Server says: “I’m also done sending data.”
    - Server finishes its part of communication.
4. **ACK (Client → Server)**
    - Client replies: “I got your FIN. Connection closed.”
    - Now both sides close connection safely.

Why 4 Steps?

- TCP is **full-duplex**, data flows both ways.
- Each side needs to **finish sending separately**.
- Ensures **no data is lost** before closing the connection.

#Congestion Control in Computer Networks

#What is Congestion?

- Congestion happens when **too much data** flows through the network at the same time.
- Similar to a **traffic jam**: everything slows down, packets may be lost, or the network can collapse.

Effects of Congestion Control

- **Improved stability:** Network runs smoothly.
- **Reduced latency & packet loss:** Data delivered faster.
- **Enhanced throughput:** More data transferred successfully.
- **Fairness:** Bandwidth shared evenly among users.
- **Better user experience:** Faster access to apps and websites.
- **Prevents collapse:** Avoids severe network breakdowns.

// #Congestion Control Algorithms

1. Leaky Bucket Algorithm

- Sends packets at a **constant rate** regardless of bursts.
- Extra packets **discarded** if bucket overflows.

Steps:

1. Packets arrive → added to bucket.
2. Bucket leaks packets at constant rate.
3. Bursty traffic becomes smooth.

Limitation: Rigid, wastes bandwidth if traffic is bursty.

---

2. Token Bucket Algorithm

- More flexible; **tokens generated** at a fixed rate.
- **Packet sent only if token available**.

Steps:

1. Tokens added at regular intervals.
2. Each token allows sending 1 packet.
3. If tokens exist → send immediately.
4. If no tokens → packet waits.

Advantage: Handles bursty traffic efficiently without losing data.

User Datagram Protocol (UDP)

What is UDP?

- UDP is a Transport Layer protocol.
- It is connectionless and fast.
- No guarantee of delivery, order, or error checking.
- Used for real-time or time-sensitive applications, such as:
    - Video streaming
    - DNS
    - VoIP

UDP Header

- **8 bytes long**, followed by data.
- Important fields:
    1. **Source Port (16 bits):** Port of sending application.
    2. **Destination Port (16 bits):** Port of receiving application.
    3. **Length (16 bits):** UDP header + data size.
    4. **Checksum (16 bits):** For error detection (optional in IPv4, mandatory in IPv6).

Note:

- UDP provides **no error or flow control**.
- Port numbers differentiate **different applications**.

Applications of UDP

1. **DNS (Domain Name System)**
    - Fast query and response lookups.
2. **DHCP (Dynamic Host Configuration Protocol)**
    - Assigns IP addresses dynamically to devices.
3. **VoIP (Voice over IP)**
    - Real-time voice communication; tolerates some packet loss.
4. **RIP (Routing Information Protocol)**
    - Sends periodic routing updates between routers.
5. **NTP (Network Time Protocol)**
    - Synchronizes system clocks with minimal overhead.

#UDP in DDoS Attacks

UDP Flood Attack:

- Attacker sends **lots of UDP packets** to target computers/ports.

How it works:

1. Fake (spoofed) IP packets sent to random ports.
2. Target replies with ICMP “Destination Unreachable.”
3. Target’s system gets overloaded → network becomes slow.

How to stop it:

- Watch for unusual UDP traffic.
- Use **rate limiting**, firewalls, or IPS.
- Use **DDoS protection services**.

#How UDP Works with IP

1. Application provides data + destination → UDP.
2. UDP attaches its header.
3. Passed to IP layer for addressing & routing.
4. IP adds its header and forwards to destination.
5. Receiver removes UDP header and delivers data to application.

TCP vs UDP

Feature	TCP	UDP
Type	Connection-oriented (needs to establish a connection first)	Connectionless (sends data directly)
Reliability	Reliable: guarantees delivery, retransmits lost packets	Not reliable: may lose packets, no retransmission
Data Order	Maintains sequence; data arrives in order	No sequencing; order not guaranteed
Speed	Slower because of reliability checks	Faster because it skips checks
Error Checking	Yes, with checksum, ACK, and flow control	Basic checksum only, no flow control
Header Size	20–60 bytes (variable)	8 bytes (fixed)
Handshaking	3-way handshake (SYN, SYN-ACK, ACK)	None (no handshake)
Broadcast	Not supported	Supports broadcast and multicast
Stream Type	Byte stream	Message stream (datagram)
Overhead	Higher due to reliability and control	Low, lightweight

Feature	TCP	UDP
Common Applications	Web browsing, email, FTP, HTTPS, Telnet	DNS, DHCP, VoIP, video streaming, online games

🌐 Network Layer (OSI Model)

The Network Layer is Layer 3 of the OSI Model.
Its main job is to deliver packets from the source host to the destination host across different networks**.**

🔑 Key Function

1. Logical Addressing (IP Addressing)

- Assigns unique IP addresses to devices.
- Helps identify devices across multiple networks.

2. Packetization

- Converts transport layer segments into **packets**.

3. Host-to-Host Delivery

- Ensures data reaches from one host to another, even if they are on different networks.

4. Forwarding

- A router takes an incoming packet and forwards it to the correct outgoing interface.

5. Routing

- Chooses the **best path** for packets using routing algorithms and protocols.

6. Fragmentation & Reassembly

- Breaks large packets into smaller fragments when MTU is small.
- Reassembles them at the destination.

7. Subnetting

- Divides one big network into smaller networks for better management.

8. NAT (Network Address Translation)

- Converts private IPs to public IPs.
- Saves IP address space and improves security.

🛠️ How the Network Layer Works (Step by Step)

1. Each device has a unique IP address.
2. Transport layer data is turned into packets.
3. Source and destination IP addresses are added.
4. Routers analyze the destination IP.
5. Packets travel **hop-by-hop** through routers.
6. If packet is too large, it is fragmented.
7. Destination device reassembles fragments.
8. If any error occurs, ICMP sends an error message back.

⭐ Advantages

- Supports end-to-end communication across different networks.
- Scalable due to hierarchical addressing and subnetting.
- Efficient packet routing.
- Connects different types of networks (internetworking).

⚠ Limitations

- No flow control → may cause congestion.
- Limited error control (depends on upper layers).

Routing	**Flooding**
Requires routing table	No routing table needed
May find shortest path	Always finds shortest path
Less reliable	More reliable
Less traffic	More traffic
No duplicate packets	Many duplicate packets

- Routers can drop packets during heavy load.
- Fragmentation increases overhead

Structure of Classful Addressing

CLASS	LEADING BITS	NET ID BITS	HOST ID BITS	NO. OF NETWORKS	ADDRESSES PER NETWORK	START ADDRESS	END ADDRESS
CLASS A	0	8	24	2^7 (128)	2^{24} (16,777,216)	0.0.0.0	127.255.255.255
CLASS B	10	16	16	2^{14} (16,384)	2^{16} (65,536)	128.0.0.0	191.255.255.255
CLASS C	110	24	8	2^{21} (2,097,152)	2^8 (256)	192.0.0.0	223.255.255.255
CLASS D	1110	NOT DEFINED	NOT DEFINED	NOT DEFINED	NOT DEFINED	224.0.0.0	239.255.255.255
CLASS E	1111	NOT DEFINED	NOT DEFINED	NOT DEFINED	NOT DEFINED	240.0.0.0	255.255.255.255

Feature	IPv4	IPv6
Address Length	32-bit	128-bit
Address Format	Decimal (e.g., 192.168.0.1)	Hexadecimal (e.g., 2001:0db8::1)
Configuration	Manual / DHCP	Auto-configuration supported
Connection Integrity	End-to-end integrity difficult	End-to-end integrity achievable
Security	External tools like IPSec	IPSec built-in
Fragmentation	Sender and routers can fragment	Only sender fragments
Flow Identification	Not available	Flow Label field in header
Checksum Field	Present	Not present
Transmission Scheme	Broadcast supported	Multicast & anycast only
Header Size	Variable (20–60 bytes)	Fixed (40 bytes)
Conversion	Can convert to IPv6	Not all IPv6 can convert to IPv4
Field Structure	4 fields separated by dot	8 fields separated by colon

Feature	IPv4	IPv6
Address Classes	A, B, C, D, E	No classes
VLSM Support	Supported	Not supported
Example	66.94.29.13	2001:0000:3238:DFE1:0063:0000:0000:FE

Feature	Private IP	Public IP
Scope / Use	Local network only	Internet / global network
Communication	Used within a LAN/network	Used to communicate over the Internet
Address Uniqueness	Unique within network, can be reused in other networks	Globally unique, cannot be reused
LAN / Internet	Works only on LAN	Used for Internet access
Control / Availability	Free, controlled by the user	Controlled by ISP, not free
Security	More secure (inside LAN)	Less secure, can be attacked
NAT Required	Yes, to communicate with public IPs	No NAT needed
How to Check	ipconfig command	Search “what is my IP” on Google

Feature	Private IP	Public IP
Range / Example	10.0.0.0 – 10.255.255.255172.16.0.0 – 172.31.255.255192.168.0.0 – 192.168.255.255Example: 192.168.1.10	All other IPs except private rangesExample: 17.5.7.8