NAT (Network Address Translation) – Computer Networks Exam Notes

Definition:
NAT is a method used in routers to **translate private IP addresses** into a **public IP address** (and vice versa) to allow multiple devices on a private network to access the internet using a **single public IP**.

🔑 Why is NAT Used?

1. **IPv4 address conservation** – Public IPs are limited.
2. **Security** – Private IPs are hidden from the outside world.
3. **Multiple devices sharing one IP** – Like all devices in your home using one IP from your ISP.

🏠 Example:

You have three devices at home:

- PC: 192.168.0.2
- Phone: 192.168.0.3
- TV: 192.168.0.4

Your router has:

- Public IP: 103.25.43.7

When any device sends data to the internet:

- NAT replaces the private IP with the public IP (103.25.43.7) and keeps track of which private IP made the request.

🛡️ Pros:

- Saves IP addresses
- Adds a layer of security
- Allows internal IP structure flexibility

⚠️ Cons:

- Breaks end-to-end connectivity (bad for P2P apps, VoIP)
- Some protocols don't work well with NAT

ARP (Address Resolution Protocol) – Computer Networks Exam Notes

Definition:
ARP is a protocol used to **map an IP address to a MAC address** in a **local network** (LAN).

🔍 **Why is ARP Needed?**

- Devices communicate using **IP addresses**.
- Ethernet (or any data link layer) needs **MAC addresses** to send data.
- So, when a device knows another device's **IP** but not its **MAC**, it uses **ARP** to find out.

What is Encapsulation?

Encapsulation is the process of **adding headers (and sometimes trailers)** to data as it passes **down the layers** of the network model **from application to physical**.

🔁 What is Decapsulation?

Decapsulation is the **reverse process**, where headers and trailers are **removed** as the data goes **up the layers** at the receiver's end.

What is a Broadcast Address?

A **broadcast address** is used to **send data to all hosts** in a particular **network**.

□ IP Address Classes – Computer Networks Exam Notes

In **IPv4**, IP addresses are divided into **classes** to identify the **network and host** portions of the address. This system helps organize IP address allocation based on network size.

📌 **IPv4 Address Format:**

- Total: **32 bits** (e.g., 192.168.1.1)
- Written in **4 octets (8 bits each)**: xxx.xxx.xxx.xxx

□ IP Address Classes Table:

Class	Starting Bits	Range of First Octet	Number of Networks	Hosts per Network	Usage
A	0	1 – 126	128 (minus reserved)	~16 million	Large networks
B	10	128 – 191	16,384	~65,000	Medium networks
C	110	192 – 223	2 million+	254	Small networks
D	1110	224 – 239	N/A	N/A	Multicasting
E	1111	240 – 255	N/A	N/A	Reserved for research

Note: 127.x.x.x is reserved for **loopback** (e.g., 127.0.0.1).

📍 **Breakdown of Classes A, B, C:**

✅ **Class A (e.g., 10.0.0.1)**

- Network: First 8 bits
- Host: Remaining 24 bits
- Subnet mask: 255.0.0.0

✅ **Class B (e.g., 172.16.0.1)**

- Network: First 16 bits
- Host: Remaining 16 bits
- Subnet mask: 255.255.0.0

✅ **Class C (e.g., 192.168.1.1)**

- Network: First 24 bits
- Host: Last 8 bits
- Subnet mask: 255.255.255.0

🔒 **Private IP Address Ranges:**

Class Private IP Range

A 10.0.0.0 – 10.255.255.255

Class	Private IP Range
B	172.16.0.0 – 172.31.255.255
C	192.168.0.0 – 192.168.255.255

These IPs are **not routable on the internet**, used inside **LANs**.

□ Key Points:

- Classes are mostly **legacy** now (we use **CIDR** instead).
- Still useful for **understanding IP structure** and subnetting.
- Helps in identifying **network scale and capacity**.

✅ Advantages of Classful IP Addressing:

1. **Simple to Understand** – Easy to learn and implement.
2. **Fixed Structure** – Classes (A, B, C) clearly define network and host portions.
3. **Easy Routing** – Routers can identify network ID from the IP class.
4. **Standardized Ranges** – Useful for assigning IPs based on network size.

❌ Disadvantages of Classful IP Addressing:

1. **Wastes IP Addresses** – Fixed sizes don't fit actual needs (e.g., 300 hosts need Class B).
2. **No Flexibility** – Can't create custom-sized subnets.
3. **IPv4 Exhaustion** – IPs used up quickly due to inefficient allocation.
4. **Outdated** – Replaced by **CIDR**, which is more efficient and scalable.

CIDR (Classless Inter-Domain Routing) – Computer Networks

CIDR is a method used to allocate and route IP addresses more efficiently than the **classful system**.

✅ What is CIDR?

CIDR is a flexible, modern way to assign and route IP addresses. It **replaces** the old classful addressing system and allows for **variable-length subnet masks (VLSM)**.

- **CIDR Notation**: <IP Address>/<Prefix Length>
    - **Example**: 192.168.1.0/24

- The **/24** means the first **24 bits** are used for the **network portion**, and the remaining 8 bits are for hosts.

✅ **Advantages of CIDR:**

1. **Efficient IP Addressing**: Reduces waste by allowing flexible subnetting.
2. **More Flexible Subnets**: Networks can be created with any number of hosts.
3. **IPv4 Conservation**: Helps prevent exhaustion by optimizing IP allocation.
4. **Simpler Routing**: CIDR aggregates multiple IP blocks into a single route, improving routing table efficiency.

❌ **Disadvantages of CIDR:**

1. **Complexity**: More complex to calculate compared to classful addressing.
2. **Routing Table Size**: Though CIDR reduces some routing table entries, large networks still lead to **BGP routing table growth**.

CIDR vs. Classful Addressing:

Feature	Classful Addressing	CIDR
Subnet Size	Fixed (A/B/C)	Variable Length
Efficiency	Wastes IPs	More efficient
IP Allocation	Rigid, non-flexible	Flexible, custom
Routing	More routing entries	Aggregates routes

🔒 **Example of CIDR in Use:**

1. **Class A Network**: 10.0.0.0/8 (first 8 bits are network, remaining 24 are hosts)
2. **Class C Network**: 192.168.1.0/24 (first 24 bits are network, remaining 8 are hosts)

Subnetting in Computer Networks

Subnetting is the process of dividing a large network into smaller, more manageable sub-networks (subnets). It helps **optimize IP address usage** and enhances **network security** and **performance**.

Subnetting Steps (General Approach):

1. **Identify the IP Class** (A, B, C).
2. **Choose the Subnet Mask**: Decide how many bits to borrow from the host portion.
3. **Calculate Subnets**: Use the borrowed bits to determine the number of subnets.
4. **Calculate Hosts per Subnet**: The remaining bits are used for host addresses.
5. **Determine Network and Broadcast Addresses**: Calculate for each subnet.

□ Subnetting Example:

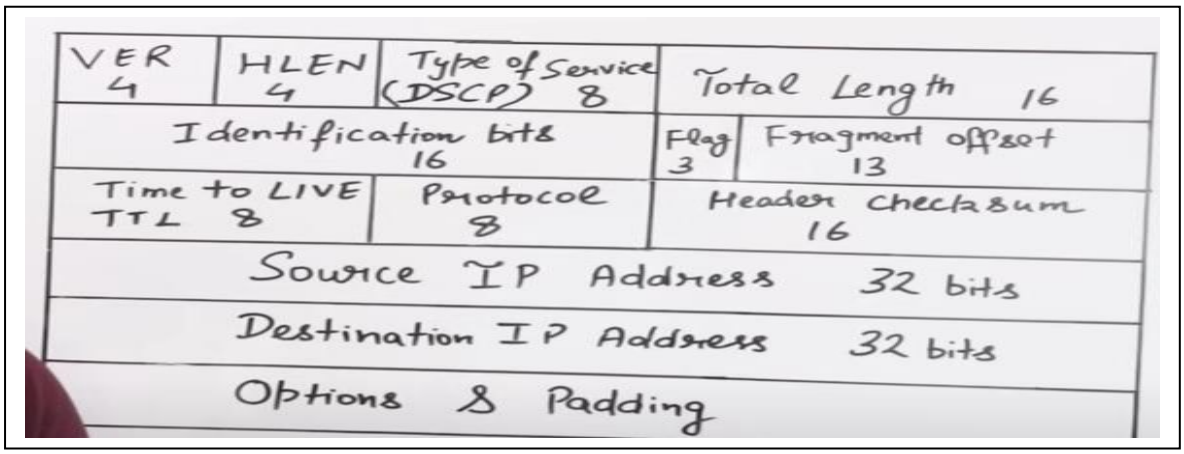
Class C Example (192.168.1.0/24):

We want to divide the 192.168.1.0/24 network into **4 subnets**.

1. **Subnet Mask**: Start with /24 (default Class C mask: 255.255.255.0).
2. **Borrow Bits**: To create 4 subnets, we borrow 2 bits (because 2^2 = 4).
    - New subnet mask: /26 (or 255.255.255.192).
3. **Number of Subnets**: We now have 4 subnets.
4. **Subnet Ranges**:
    - 192.168.1.0/26 → Network: 192.168.1.0, Broadcast: 192.168.1.63, Usable IP range: 192.168.1.1 – 192.168.1.62
    - 192.168.1.64/26 → Network: 192.168.1.64, Broadcast: 192.168.1.127, Usable IP range: 192.168.1.65 – 192.168.1.126
    - 192.168.1.128/26 → Network: 192.168.1.128, Broadcast: 192.168.1.191, Usable IP range: 192.168.1.129 – 192.168.1.190
    - 192.168.1.192/26 → Network: 192.168.1.192, Broadcast: 192.168.1.255, Usable IP range: 192.168.1.193 – 192.168.1.254

IPv4 Header Format – All Fields Explained

The **IPv4 header** contains information required for routing and delivering packets to the destination. The header is structured in a way that ensures each device in the network understands how to handle the packet. Let's break it down field by field.



- **Version (4 bits)**:
  - Specifies the version of IP used. For IPv4, this is always 4.
- **IHL (Internet Header Length, 4 bits)**:
  - Specifies the length of the IPv4 header in **32-bit words**.
  - Minimum value: 5 (for a 20-byte header, no options).
- **Type of Service (ToS, 8 bits)**:
  - Used to indicate the desired QoS (Quality of Service) treatment.
  - Nowadays, it's often used for **Differentiated Services** (DSCP).
- **Total Length (16 bits)**:
  - The total size of the IP packet (header + data).
  - Maximum value: 65535 bytes.
- **Identification (16 bits)**:
  - Unique identifier for each packet. Used for fragment reassembly.
- **Flags (3 bits)**:
  - **Reserved (0)**: Always 0.
  - **Don't Fragment (DF)**: A flag to prevent fragmentation.
  - **More Fragments (MF)**: Indicates more fragments follow.
- **Fragment Offset (13 bits)**:
  - Used during fragmentation to indicate the position of a fragment in the original packet.

 Time to Live (TTL, 8 bits):

- Specifies how many hops (routers) the packet can make before being discarded.
- Each router decreases the TTL by 1. If TTL reaches 0, the packet is discarded.

 Protocol (8 bits):

- Indicates the upper-layer protocol that the packet data corresponds to (e.g., TCP = 6, UDP = 17).

 Header Checksum (16 bits):

- A checksum calculated to detect errors in the header. If the checksum is incorrect, the packet is discarded.

 Source Address (32 bits):

- The **source IP address** of the packet (sender).

 Destination Address (32 bits):

- The **destination IP address** of the packet (receiver).

 Options (variable):

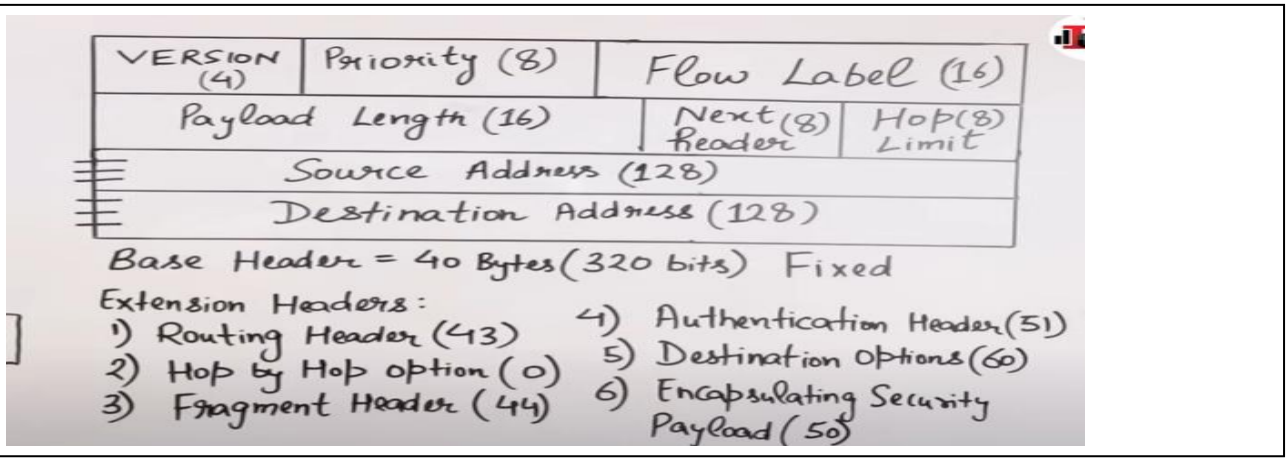
- Used for additional features like security, record route, timestamp, or source routing.
- Usually, this field is not used, and the header is just 20 bytes long.

 Padding:

- Padding bits are added to make the total header length a multiple of 32 bits when options are included.

IPv6 – Internet Protocol Version 6

IPv6 is the **new generation** of the **Internet Protocol** designed to solve the limitations of IPv4, especially regarding address exhaustion. It uses a **128-bit** address format, compared to the **32-bit** address format in IPv4, allowing for an **extremely large number of addresses**.



1. **Version (4 bits)**:
    - **What it is**: Specifies the IP version. For IPv6, this is always 6.
2. **Traffic Class (8 bits)**:
    - **What it is**: Similar to **Type of Service (ToS)** in IPv4. Used to differentiate traffic based on priority and QoS needs.
    - **Why it's important**: Helps manage and prioritize network traffic, like differentiating between video streaming and regular browsing.
3. **Flow Label (20 bits)**:
    - **What it is**: Used to identify a flow of packets that require special handling by routers. A flow is a stream of packets sent from a particular source to a specific destination with the same properties (e.g., quality of service).
    - **Why it's important**: This field helps ensure all packets belonging to the same flow are treated the same way across the network.
4. **Payload Length (16 bits)**:
    - **What it is**: The size of the data being carried by the packet. It specifies the length of the payload (data), excluding the header.
    - **Why it's important**: It helps routers understand how much data is inside the packet, so they can properly route it.
5. **Next Header (8 bits)**:
    - **What it is**: Identifies the protocol used in the payload of the packet (similar to the **Protocol** field in IPv4).
    - **Examples**:
        - **6**: TCP
        - **17**: UDP
        - **58**: ICMPv6
6. **Hop Limit (8 bits)**:
    - **What it is**: Similar to **TTL (Time to Live)** in IPv4, this field limits the number of hops (routers) the packet can pass through before being discarded.
    - **Why it's important**: Prevents packets from endlessly circulating the network in case of routing errors.
7. **Source Address (128 bits)**:
    - **What it is**: The **IPv6 address** of the sender (source). It's a 128-bit address, allowing for a significantly larger address space compared to IPv4.
    - **Why it's important**: Identifies the origin of the packet.
    -

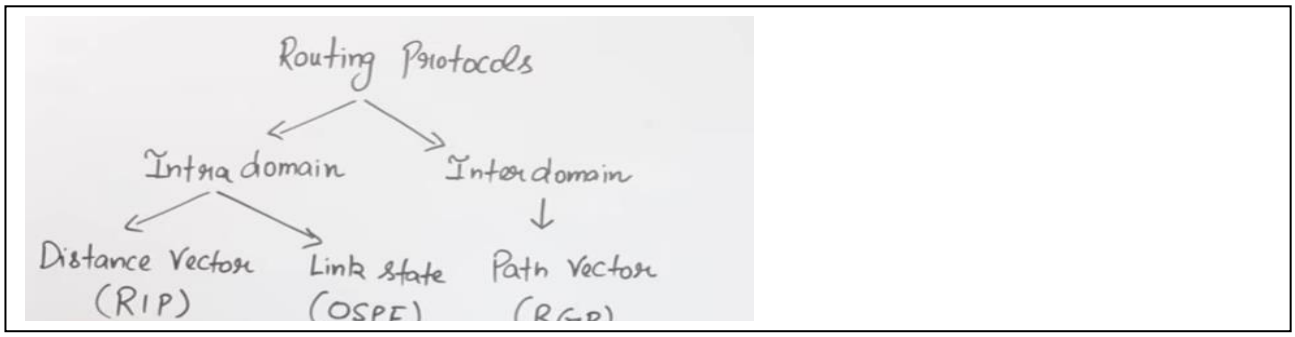
8. **Destination Address (128 bits)**:
    - **What it is**: The **IPv6 address** of the receiver (destination). Like the source address, it's also 128 bits.
    - **Why it's important**: Identifies where the packet should go.

Routing in Computer Networks

Routing is the process of forwarding data packets from the source to the destination across different networks. It involves determining the **best path** for the packets to travel, from the source to the destination, based on certain criteria.

Routing Protocols:

Routing protocols are used by routers to determine the best path for data packets



VIDEO DEKHLO BHAI LEC NUMBER:-58,59,60

Steps in Link State Routing Process:

1. **Initialization**:
    - Routers initialize their link-state information. They know about their own interfaces and neighbors.
2. **Flooding LSAs**:
    - Each router floods the LSAs across the network to inform all other routers of its links and their statuses.
3. **Building LSDB**:
    - Every router collects the LSAs and builds a database (LSDB) that holds the complete network topology.
4. **SPF Calculation**:
    - Using the LSDB, each router runs the **Dijkstra's SPF algorithm** to compute the best paths to every other router in the network.
5. **Routing Table Update**:
    - Based on the SPF calculation, the router updates its routing table with the best paths to all destinations.

6. **Handling Network Changes**:
    - If a link status changes (e.g., a router or link fails), the router sends an updated LSA, and the network recalculates the routes.

Feature	Distance Vector Routing	Link State Routing
Routing Information	Routers share full routing tables with neighbors.	Routers share only information about their own links.
Algorithm Used	Bellman-Ford algorithm (hop count metric).	SPF algorithm (Dijkstra's algorithm).
Convergence Time	Slower convergence due to periodic updates.	Faster convergence due to faster propagation of link changes.
Routing Table Information	Contains destinations, next hop, and distance.	Contains complete network topology (Link-State Database).
Scalability	Less scalable, struggles with large networks.	More scalable, handles large networks efficiently.
Overhead	Lower overhead in memory and CPU, but high due to full table exchanges.	Higher overhead due to LSDB maintenance and flooding LSAs.
Examples	RIP, IGRP	OSPF, IS-IS
Advantages	Simple to configure, requires less memory.	Fast convergence, more accurate network view, scalable.
Disadvantages	Slow convergence, prone to routing loops (count-to-infinity).	High overhead, requires more resources (memory, CPU).

Transport Layer

The **Transport Layer** (Layer 4) in the OSI model is responsible for ensuring reliable communication between devices over a network. It sits above the Network Layer (Layer 3) and below the Session Layer (Layer 5)

Here's a shorter version of the **Transport Layer Responsibilities**:

1. **Segmentation & Reassembly**: Breaks data into smaller segments for transmission and reassembles it at the receiver.
2. **End-to-End Communication**: Ensures data is transferred reliably between source and destination.
3. **Flow Control**: Regulates the rate of data transfer to prevent congestion.
4. **Error Detection & Correction**: Detects and corrects errors in data transmission.
5. **Reliable Data Transfer**: Ensures that data is delivered accurately and in the correct order (e.g., TCP).
6. **Connection Establishment & Termination**: Handles the setup and teardown of connections (e.g., TCP handshake).
7. **Multiplexing**: Allows multiple communication sessions using different port numbers.
8. **Protocol Support**: Supports TCP (reliable) and UDP (unreliable) protocols for different needs.

A socket address is a combination of an IP address and a port number, which uniquely identifies a communication endpoint in a network. It is used in the Transport Layer to enable communication between devices over a network.

TCP (Transmission Control Protocol)

TCP is a **connection-oriented** protocol in the **Transport Layer** of the OSI model. It is one of the most commonly used protocols for reliable communication over a network. TCP ensures that data is transmitted in the correct order, without errors, and guarantees delivery.

Key Features of TCP:

1. **Connection-Oriented**:
    - Before data transfer begins, a connection is established between the sender and receiver. This process is known as the **three-way handshake**.
2. **Reliable Data Delivery**:
    - Guarantees that data reaches its destination without errors. If any data is lost or corrupted, TCP will retransmit it.

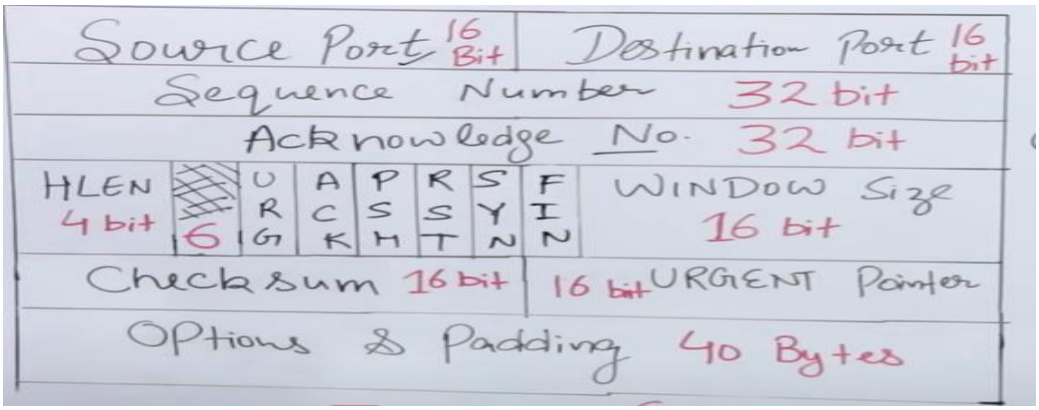
3. **Flow Control**:
    - Uses **windowing** to control the rate of data transmission, preventing the sender from overwhelming the receiver with too much data at once.
4. **Error Detection and Correction**:
    - Every TCP segment has a **checksum** for error detection. If a segment is corrupted, TCP requests a retransmission.
5. **Ordered Data Delivery**:
    - TCP ensures that data is received in the correct order. If segments are received out of order, TCP will reorder them before passing to the application.
6. **Full Duplex Communication**:
    - Allows both sender and receiver to transmit data simultaneously, enabling two-way communication.
7. **Congestion Control**:
    - TCP adjusts the data transmission rate based on the network's current traffic load to avoid congestion.

How TCP Works:

1. **Connection Establishment** (Three-Way Handshake):
    - **Step 1**: The client sends a SYN (synchronize) message to the server.
    - **Step 2**: The server responds with a SYN-ACK (synchronize-acknowledge) message.
    - **Step 3**: The client sends an ACK (acknowledge) message, completing the handshake.
2. **Data Transfer**:
    - Data is sent in **segments**. Each segment contains a sequence number to ensure correct ordering and error checking.
3. **Connection Termination**:
    - After data transfer is complete, both sides send FIN (finish) messages to gracefully terminate the connection.

Applications of TCP:

- **Web browsing (HTTP/HTTPS)**: Ensures reliable transmission of web pages.
- **File Transfer (FTP)**: Guarantees complete and accurate file transfer.
- **Email (SMTP, IMAP)**: Ensures reliable transmission of emails.

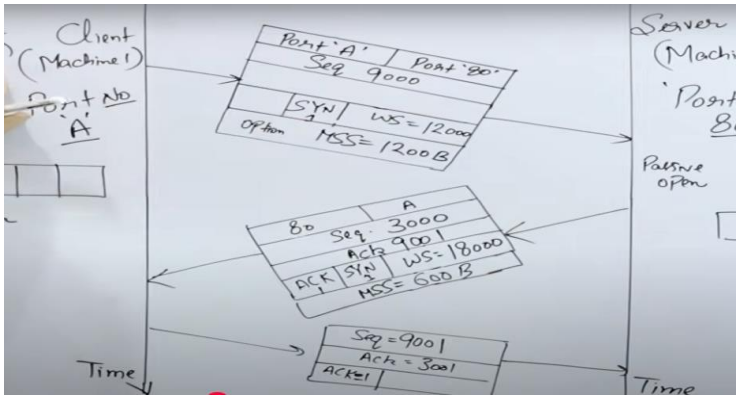


Field	Size	Description
Source Port	16 bits	Port number of the sender.
Destination Port	16 bits	Port number of the receiver.
Sequence Number	32 bits	Indicates the sequence number of the first byte in this segment.
Acknowledgment Number	32 bits	Used to acknowledge received data (next expected byte).
Data Offset (Header Length)	4 bits	Length of the TCP header in 32-bit words.
Reserved	3 bits	Reserved for future use (always 0).
Flags (Control Bits)	9 bits	Control information (e.g., SYN, ACK, FIN, RST, PSH, URG).
Window Size	16 bits	Size of the receiver's window (for flow control).
Checksum	16 bits	Error-checking for header and data.
Urgent Pointer	16 bits	Used if the URG flag is set (points to urgent data).
Options (if any)	Variable	Used for additional features like maximum segment size (MSS).
Padding	Variable	Added to ensure header length is a multiple of 32 bits.

Control Flags (Key Ones):

- **SYN** – Initiate connection.

- **ACK** – Acknowledges received data.
- **FIN** – Graceful connection termination.
- **RST** – Reset the connection.
- **PSH** – Push the data immediately to the application.
- **URG** – Urgent pointer field is significant.



Pure Acknowledgement

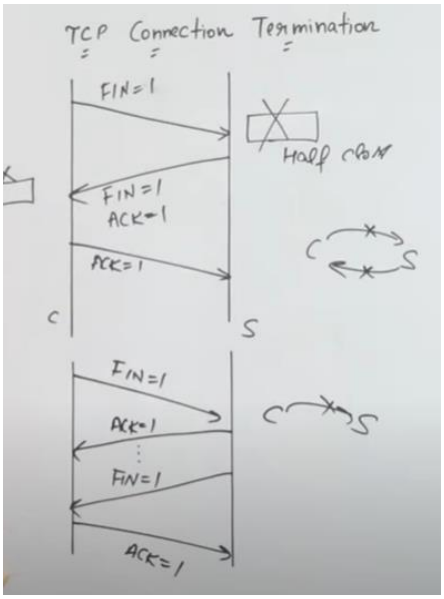
- The receiver **sends an ACK** packet **without** any data.
- Used **only to confirm** that data was received.
- Common when the receiver has **no data to send back**.

Piggybacking

- The receiver **delays the ACK** slightly to **include it with outgoing data**.
- So the ACK **"rides along"** with the data going in the reverse direction.

Feature	**Pure Acknowledgement**	**Piggybacking**
ACK sent with data?	❌ No	✅ Yes
Bandwidth usage	Higher (more packets)	Lower (fewer packets)
Delay	None (immediate ACK)	Slight delay possible
Efficiency	Less efficient	More efficient

Connection Termination in TCP



UDP (User Datagram Protocol)

UDP is a **connectionless** and **lightweight** transport layer protocol used for fast communication where **speed is more important than reliability**.

🔑 **Key Features of UDP:**

1. **Connectionless**
    - No need to establish or maintain a connection.
    - No handshake like TCP.
2. **Fast & Lightweight**
    - Minimal overhead.
    - Suitable for real-time applications.
3. **No Reliability**
    - No guarantee of delivery, ordering, or error checking beyond a basic checksum.
4. **Message-Oriented**
    - Sends data as independent packets (datagrams).
    - Each packet is handled separately.

5. **No Flow or Congestion Control**
    - Sends as much data as the application wants, as fast as it wants.

📥 When to Use UDP?

- **Live Streaming (audio/video)**
- **Online Gaming**
- **VoIP (Voice over IP)**
- **DNS (Domain Name System)**
- **TFTP (Trivial File Transfer Protocol)**

✉️ UDP Header Format (8 Bytes Only):

Field	Size	Description
Source Port	16 bits	Sending port
Destination Port	16 bits	Receiving port
Length	16 bits	Length of UDP header + data
Checksum	16 bits	Basic error checking (optional)

✅ Advantages of UDP:

- Faster than TCP
- Less overhead
- Suitable for time-sensitive applications

❌ Disadvantages of UDP:

- No error recovery
- No data order guarantee
- Packets may be lost or duplicated

✅ TCP vs UDP

Feature	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Connection	Connection-oriented	Connectionless
Reliability	Reliable (acknowledgment, retransmission)	Unreliable (no guarantee of delivery)
Speed	Slower due to overhead	Faster, minimal overhead
Ordering	Ensures ordered delivery	No guarantee of order
Error Checking	Yes (with error recovery)	Yes (basic checksum, no recovery)
Header Size	20–60 bytes	8 bytes
Flow Control	Yes	No
Congestion Control	Yes	No
Use Case	File transfer, emails, web (HTTP, FTP)	Live video/audio, DNS, online gaming
Data Transmission	Stream-based	Message-based (Datagrams)

Definition of some terms:-

1. Hub

- A **basic networking device** that connects multiple computers in a network.
- It **broadcasts** data to **all** devices connected, regardless of the destination.
- **No intelligence**, works at **Physical Layer (Layer 1)**.
- ⚠️ **Inefficient** and rarely used today.

🔁 2. Repeater

- A device that **regenerates and amplifies** weak signals in a network.
- Used to **extend** the range of a network.
- Operates at the **Physical Layer (Layer 1)**.

🌉 3. Bridge

- Connects **two LAN segments** and **filters** traffic.
- Uses **MAC addresses** to decide whether to forward or block data.

- Operates at the **Data Link Layer (Layer 2)**.

🔄 **4. Switch**

- A smarter hub that **forwards data only to the intended device** using **MAC addresses**.
- Reduces network traffic and collisions.
- Works at the **Data Link Layer (Layer 2)**.

🌐 **5. Router**

- Connects **different networks** (e.g., LAN to Internet).
- Uses **IP addresses** to **route data** between networks.
- Operates at the **Network Layer (Layer 3)**.
- Can perform **NAT**, **firewalling**, and **routing**.

P2P vs Client/Server

Feature	Peer-to-Peer (P2P)	Client/Server
Architecture	All devices (peers) act as both client & server	Separate clients and a central server
Control	Decentralized	Centralized
Resource Sharing	Each peer shares its resources	Server provides resources/services to clients
Scalability	Hard to manage with many peers	Easier to scale with proper server setup
Security	Less secure (no central control)	More secure (centralized policies)
Speed	Can be slower due to distributed nature	Generally faster and more efficient
Examples	Torrenting (BitTorrent), Skype (older)	Web browsing, Email, Online banking

OSI Model (Open Systems Interconnection)

The **OSI model** is a **7-layer framework** that standardizes how different network devices communicate over a network. Each layer performs a specific function in the process of data transmission.

🔢 7 Layers of OSI Model (Top to Bottom):

Layer No.	Layer Name	Function
7️⃣	**Application Layer**	User interface, network services (e.g., email, web)
6️⃣	**Presentation Layer**	Data translation, encryption, compression
5️⃣	**Session Layer**	Session management (start, maintain, end communication)
4️⃣	**Transport Layer**	Reliable delivery (TCP/UDP), flow control, error handling
3️⃣	**Network Layer**	Routing, logical addressing (IP)
2️⃣	**Data Link Layer**	The data link layer is responsible for moving frames from one hop (node) to the next. MAC address, error detection/correction, frame formatting
1️⃣	**Physical Layer**	Transmission of raw bits over the medium (cables, signals)

Data Link Layer (Layer 2 of OSI Model)

The **Data Link Layer** is responsible for **node-to-node delivery** — it ensures that data is transferred **safely and correctly between two directly connected devices** on the same network.

🔑 **Main Functions:**

Function	Description
Framing	Breaks data into **frames** before transmission.
Error Detection & Handling	Uses checksums like **CRC** to detect errors in frames.
Flow Control	Manages data rate so the receiver is not overwhelmed.
MAC Addressing	Uses **MAC addresses** to identify devices on the same network.
Access Control	Decides **which device can use the medium** (e.g., CSMA/CD in Ethernet

DATAGRAM ME HEADER AOR FOOTER LAGANE SE FRAMES BANTA HAI , HEADER AOR FOOTER LAGANE KO ENCAPSULATION KEHTE HAI

How Checksum Works (Simple Steps):

1. **Sender Side**:
    - Break the data into small fixed-size segments (e.g., 16-bit words).
    - Add all the segments using binary addition.
    - **Invert the result** (1's complement) → this is the **checksum**.
    - Send **data + checksum** to the receiver.
2. **Receiver Side**:
    - Add all received segments **including checksum**.
    - If the final sum is **all 1s**, the data is **assumed correct**.
    - If not, **error is detected**.