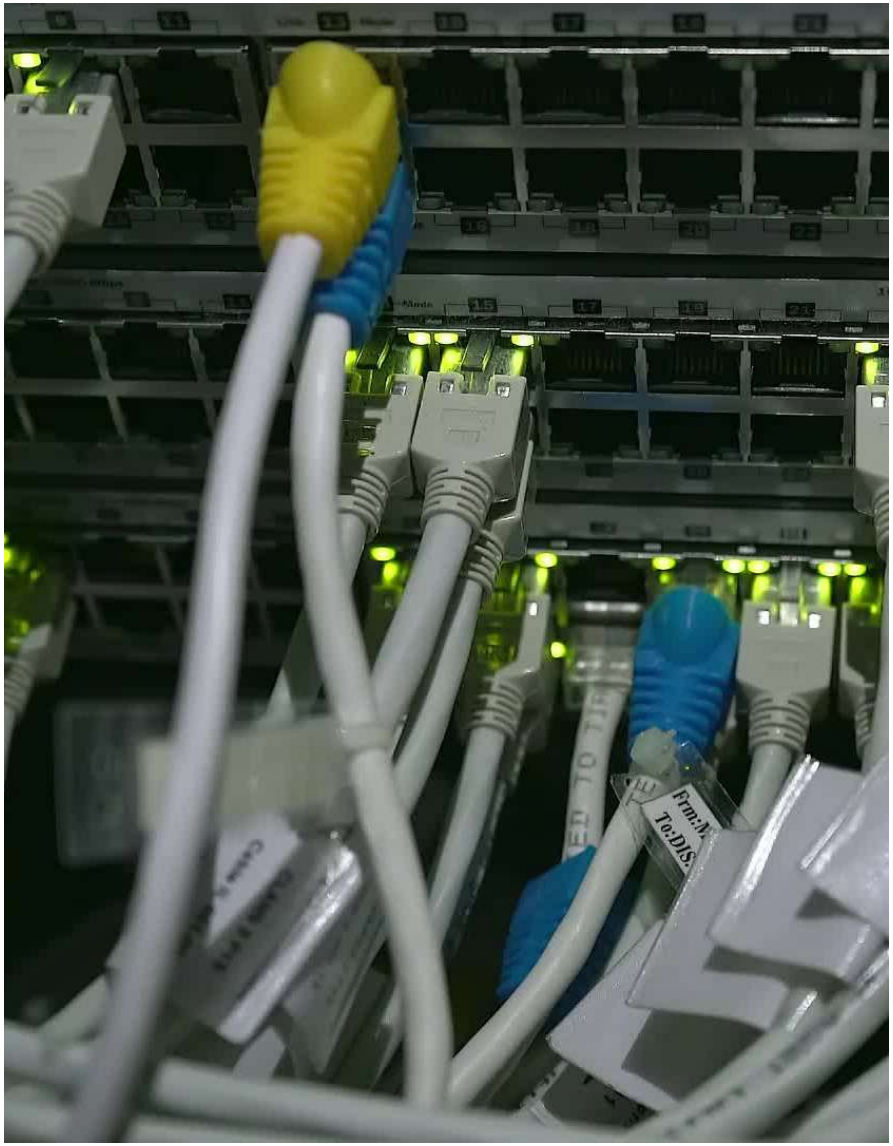
Networks
420-421-VA

Review Session

Sept 23, 2025

What Is a Network ?

- A computer network is a set of computers sharing resources located on or provided by network nodes.
- Computers use common communication protocols over digital interconnections to communicate with each other.
- These interconnections are made up of telecommunication network technologies based on physically wired, optical, and wireless radio frequency methods that may be arranged in a variety of network topologies.



Advanced Research Projects Agency

(ARPA)

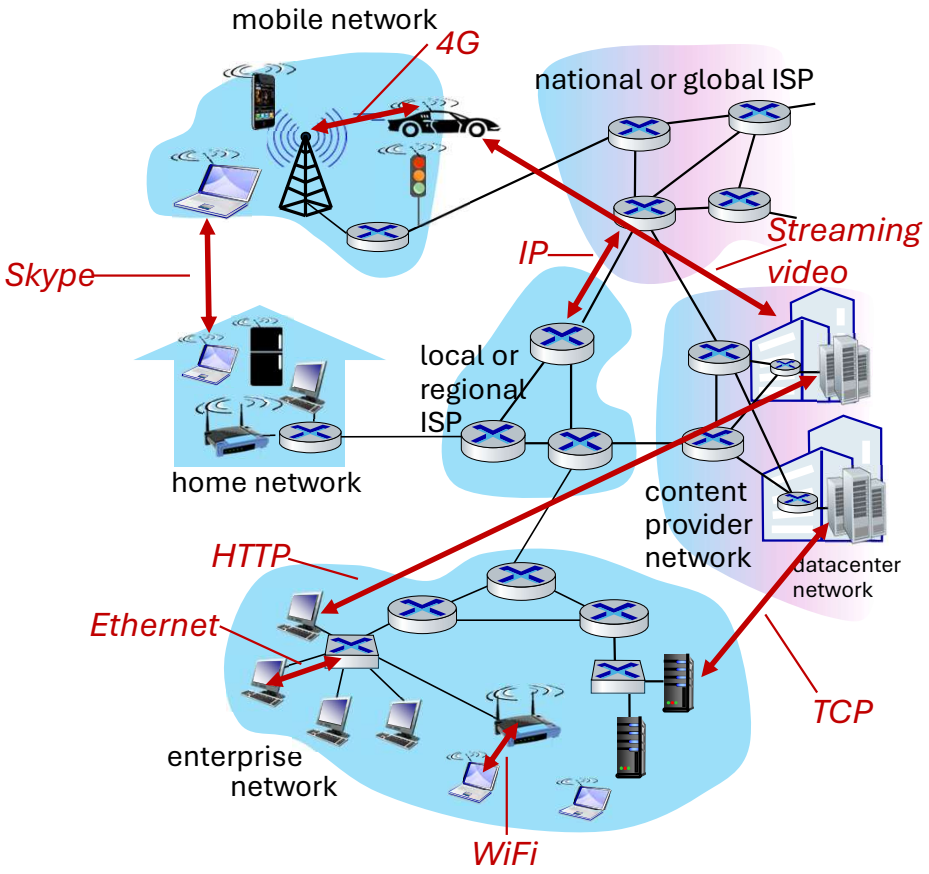
- In 1958, President Dwight Eisenhower created an agency called Advanced Research Projects Agency **(ARPA)**
- This was a response to the launch of Sputnik by the Russian in October 1957
- The goal of ARPA was to ensure that the USA has technological advantage over its enemies
- ARPA facilitated research which in 1969 lead to the establishment of the first packet switching network known as ARPANET

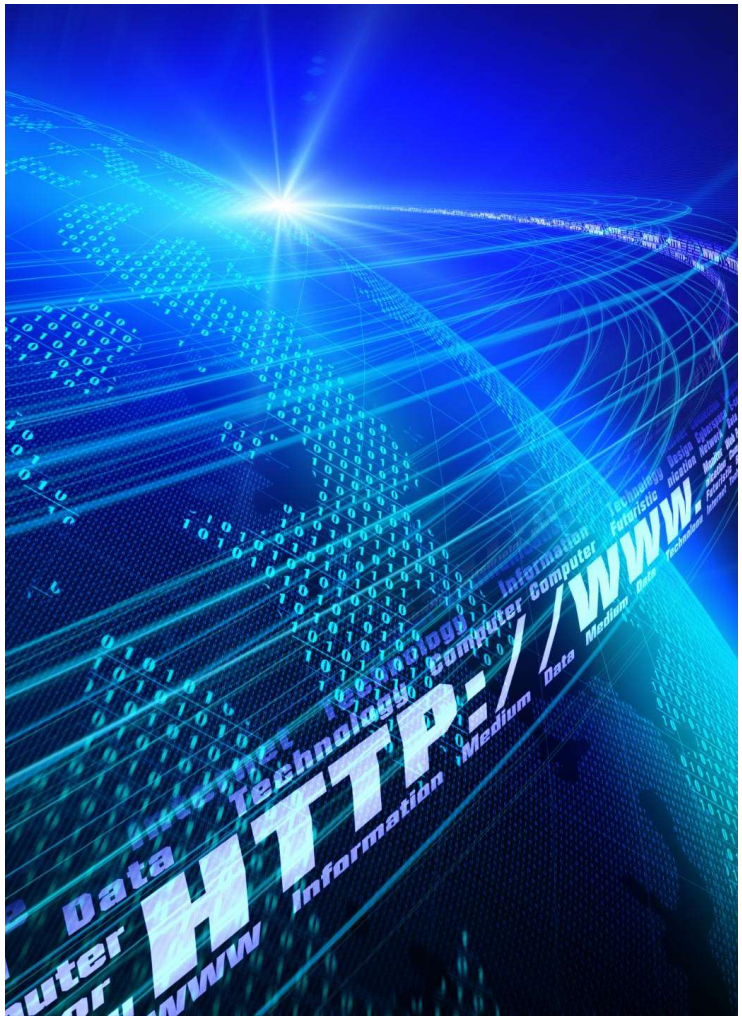
ARPANET

- The first link of ARPANET was established between University of California at Los Angeles (UCLA) and Stanford Institute of Research (SRI), in November 1969
- By December 1969, ARPANET expanded to four nodes

The Internet: a "nuts and bolts" view

- *Internet*: "network of networks"
  - Interconnected ISPs
- *protocols* are *everywhere*
  - control sending, receiving of messages
  - e.g., HTTP (Web), streaming video, Skype, TCP, IP, WiFi, 4G, Ethernet
- *Internet standards*
  - RFC: Request for Comments
  - IETF: Internet Engineering Task Force





Protocols and Standards

Role of Protocols

Protocols establish the rules and conventions for communication between network devices, enabling efficient data transmission.

Key Standards

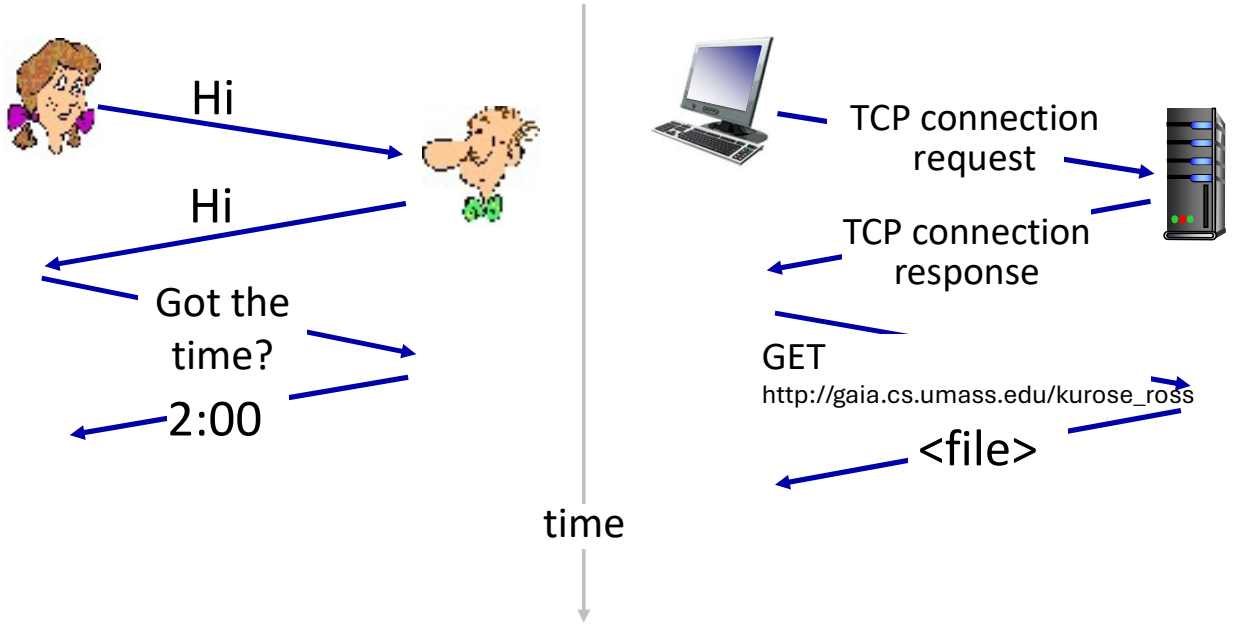
Standards like TCP/IP and HTTP provide frameworks for interoperability, allowing different systems to communicate effectively.

Interoperability Importance

Interoperability ensures that diverse technologies can work together seamlessly, enhancing overall network functionality.

What's a protocol?

A human protocol and a computer network protocol:



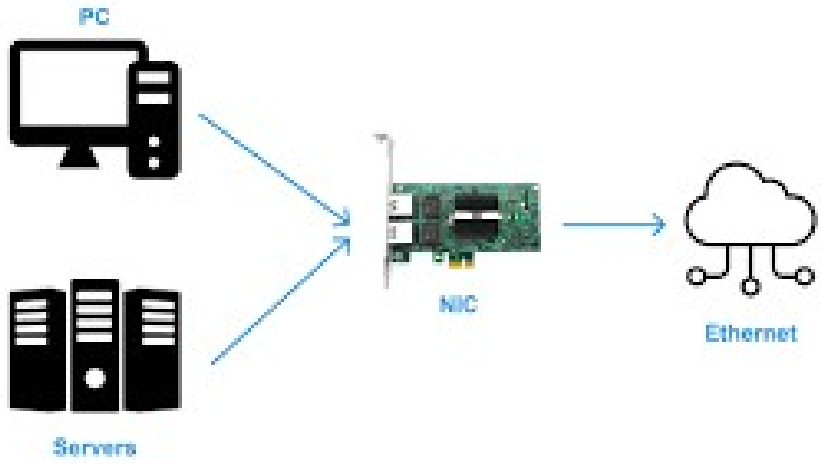
Q: other human protocols?

Communication Protocols

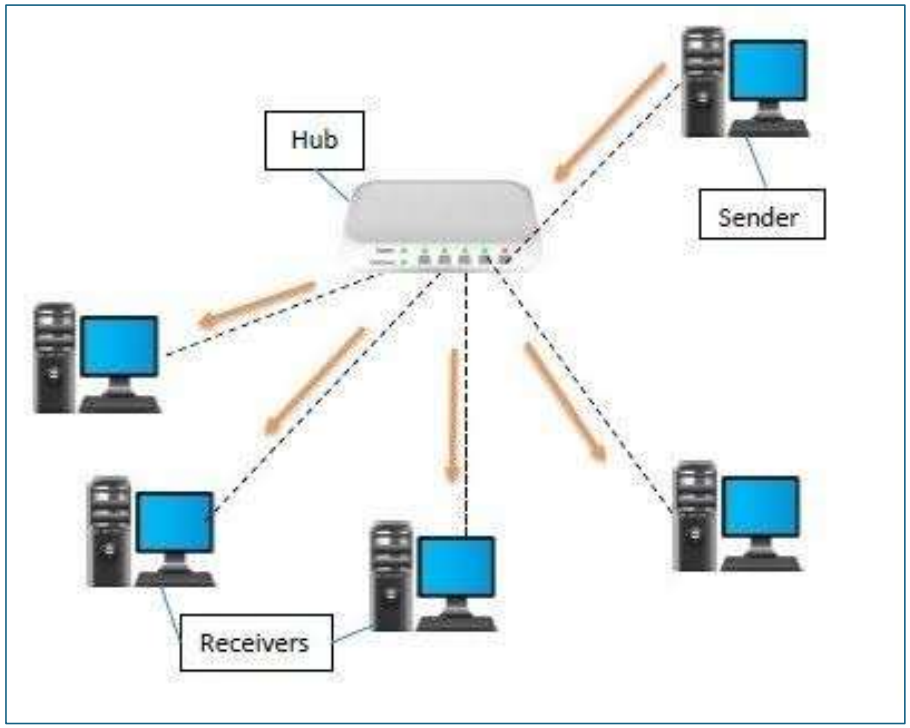
- When a computer wishes to transmit data to another computer the message is broken up into small pieces called packets
  - A packet is a block of computer data sent and received on a network
- The best way to visualize a packet is to think of it as a letter
- Like a letter a packet has a destination and a return or source address

Network Interface Card (NIC)

- The components of a network are
  - **Network Interface Card (NIC)**
  - Transport medium (cables or wireless medium)
  - Hubs
  - Bridges
  - Switches
  - Routers

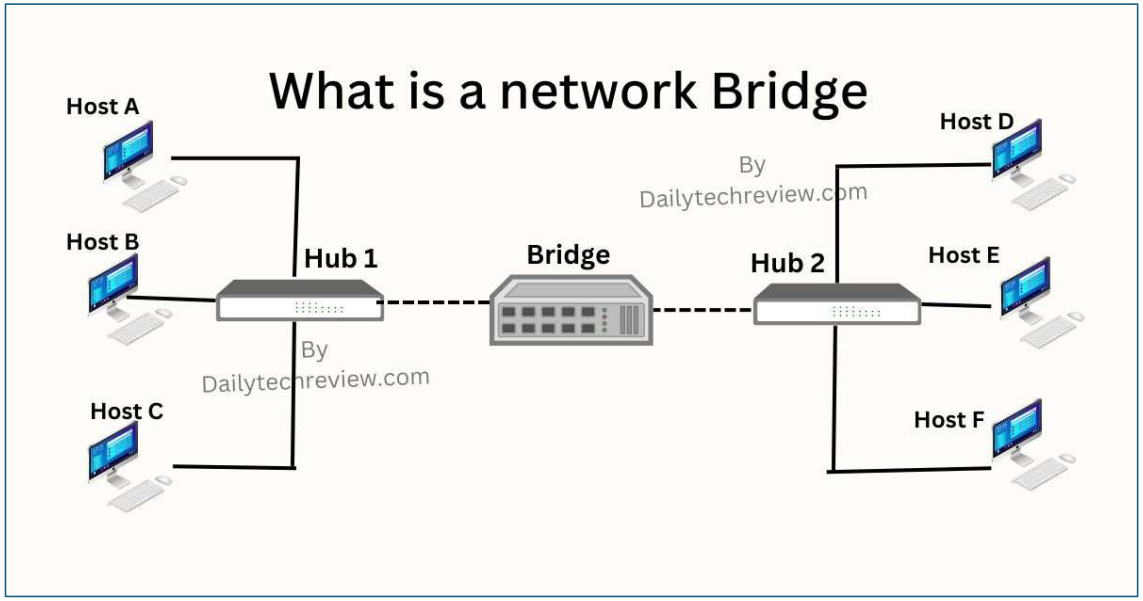


Hubs



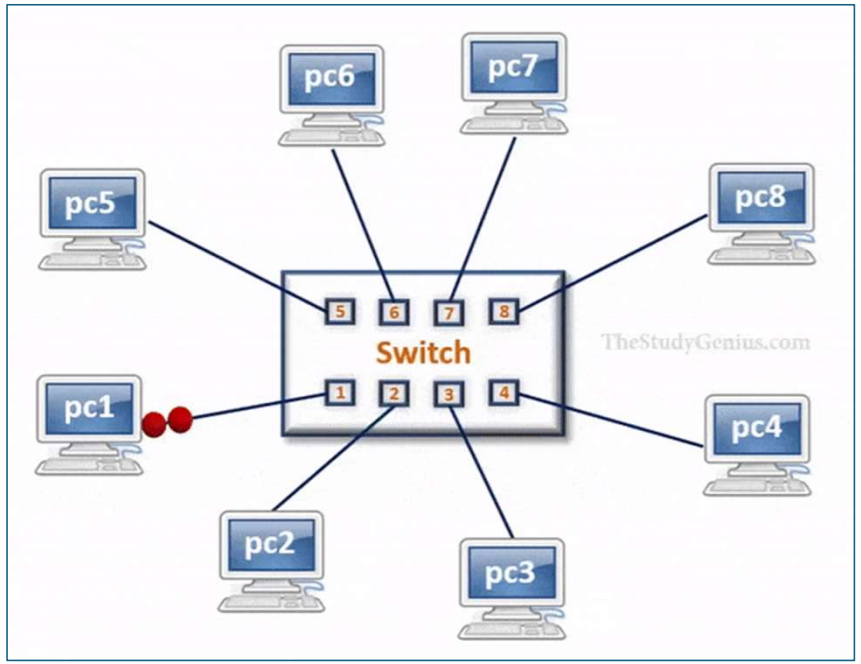
- A common connection point in a network
- Hub does not filter data, so it sends the packets it receive to all connected devices/computers
- Not suitable for large networks due to wasteful transmission which cause traffic congestion

Bridge



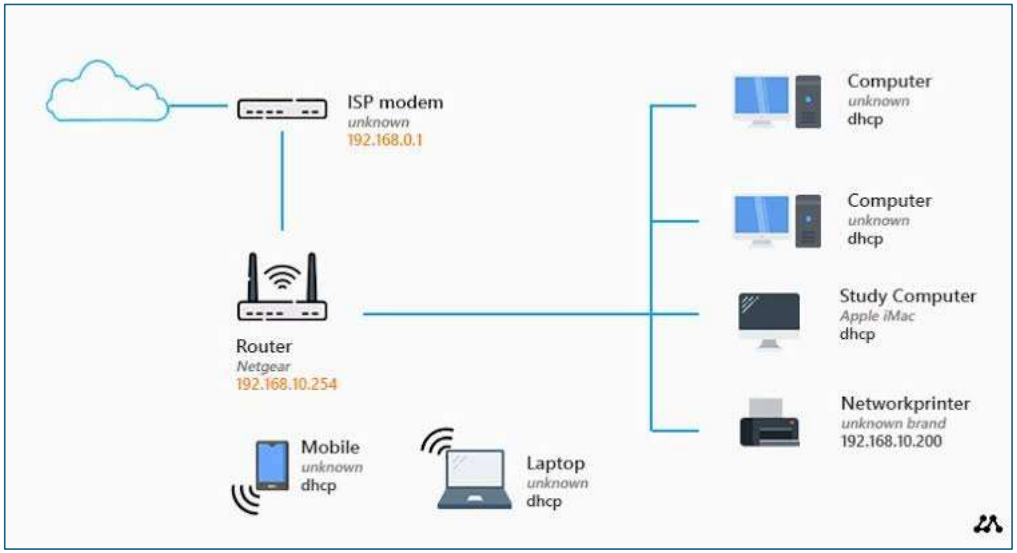
- Connects a local area network (LAN) to another LAN
- Unlike a hub, bridge looks at the destination of a packet before forwarding it, and it will send a packet to a destination only if it the packet was meant for the given destination
- Also, unlike hubs, bridges normally have only two ports, one for receiving data and the other for sending data
- Bridges are typically used to connect two networks which have different architectures (ex. An Ethernet network to a Token ring segment)

Switch



- Similarly to a hub, a switch serves as a central connection point for the devices on a network
- Unlike a hub, a switch keeps a record of the address of all the devices that are connected to it; therefore, when it receives a packet, it only forwards it the intended destination
- Unlike a bridge, a switch has multiple port
- Unlike hubs and bridges, switches also check packets for error before forwarding them

Router



- Differs from hub, bridge and switch, it that its primary function is to route packets from a source to a destination
- Routers are located at networks gateways (point where two are more networks are connected)
- Current routers often have switch and router functionality

Key Hardware Components

Routers

Routers are crucial for directing data packets between networks, ensuring efficient traffic management and connectivity.

Switches

Switches connect devices within a network, facilitating communication and data transfer between them.

Hubs

Hubs serve as central points of connection for devices in a network, allowing for data sharing among them.

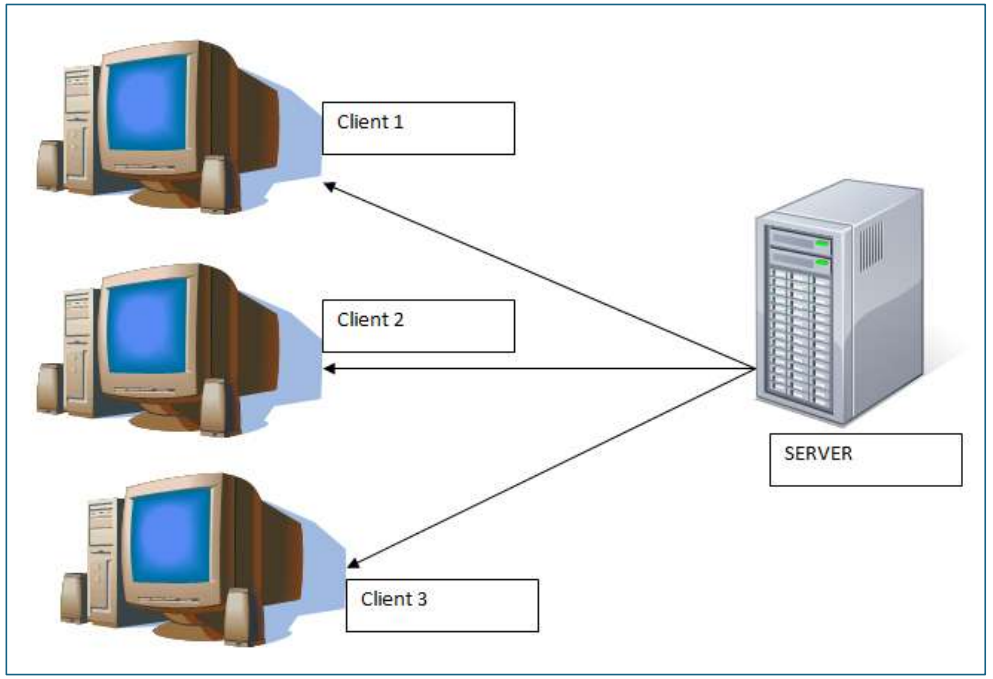
Access Points

Access points enable wireless devices to connect to a wired network, enhancing mobility and connectivity options.

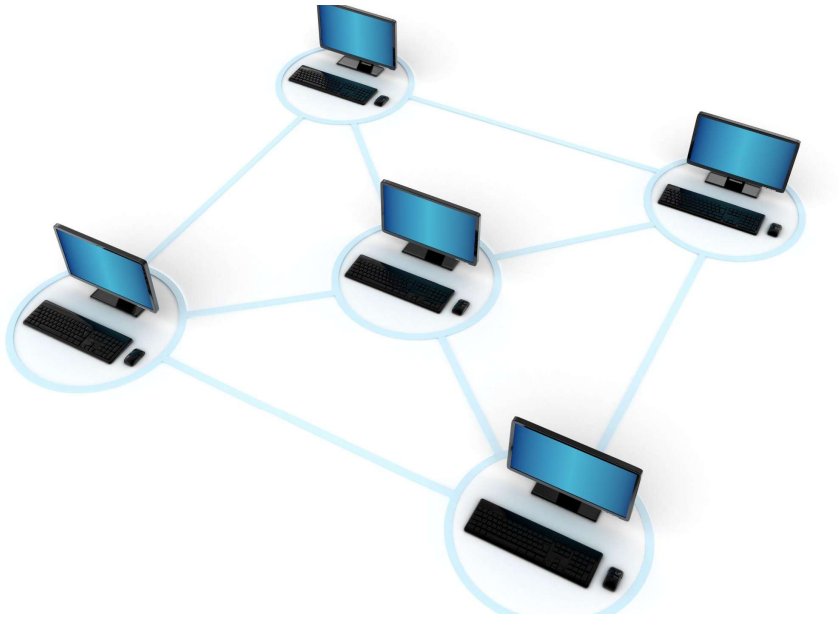
Types of network – Client Server Network

A **client-server network** is a type of network architecture where resources, services, or data are provided by one or more **servers**, and users (or devices) that request these resources or services are known as **clients**.

The client and server roles are separate and distinct, each performing different tasks within the network



Peer-to-Peer Networks



Decentralized Resource Sharing

Peer-to-peer networks enable users to share resources directly, eliminating the need for a centralized server.

File Sharing Capabilities

This network model is popular for file sharing, allowing users to exchange data easily and efficiently.

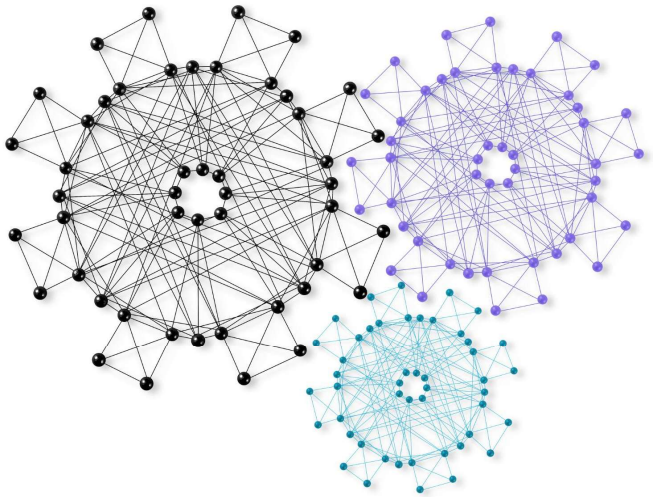
Decentralized Applications

Peer-to-peer networks support decentralized applications, promoting innovation and collaboration among users.

Increased Resilience

The absence of a central server in peer-to-peer networks enhances resilience against failures and attacks.

Basic Network Topologies



Star Topology

Star topology connects all devices to a central hub, providing easy management and fault isolation.

Ring Topology

In a ring topology, each device is connected to two others, forming a circular data path, allowing for orderly data transmission.

Bus Topology

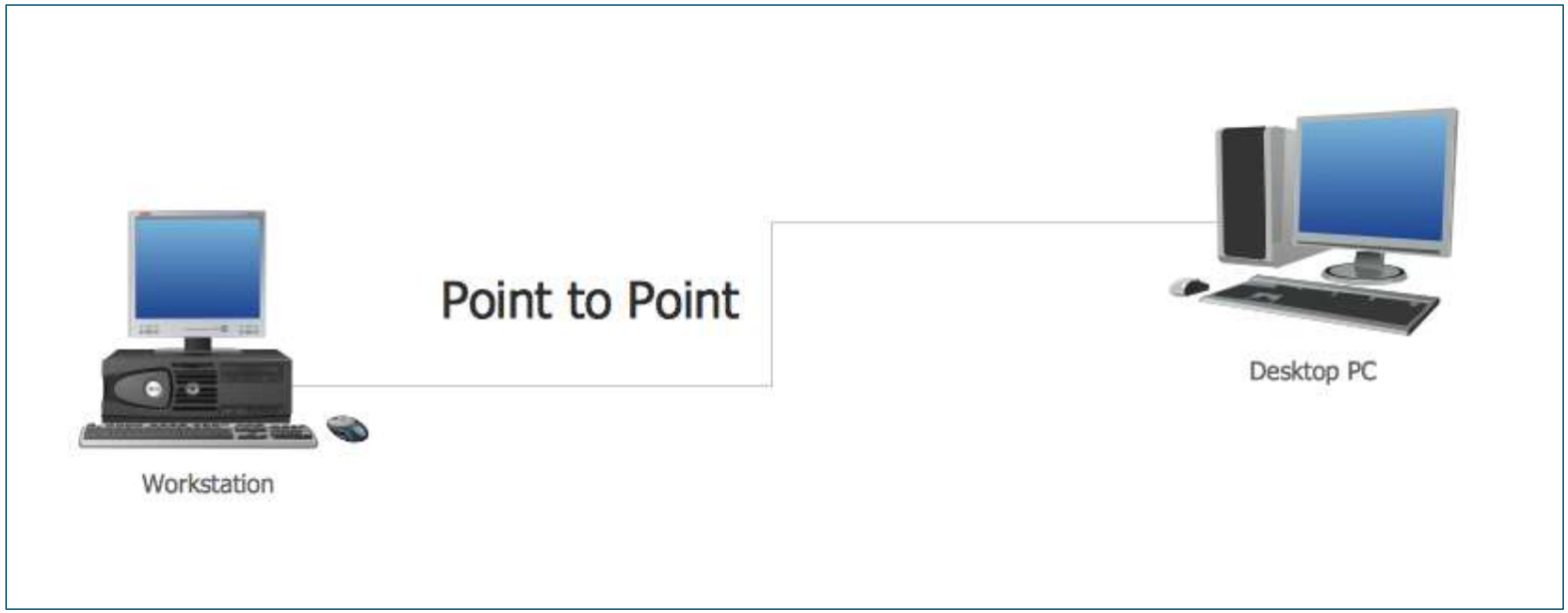
Bus topology uses a single central cable, with all devices connected to it, making it simple but prone to data collisions.

Mesh Topology

Mesh topology connects every device to every other device, enhancing redundancy and reliability but increasing complexity.

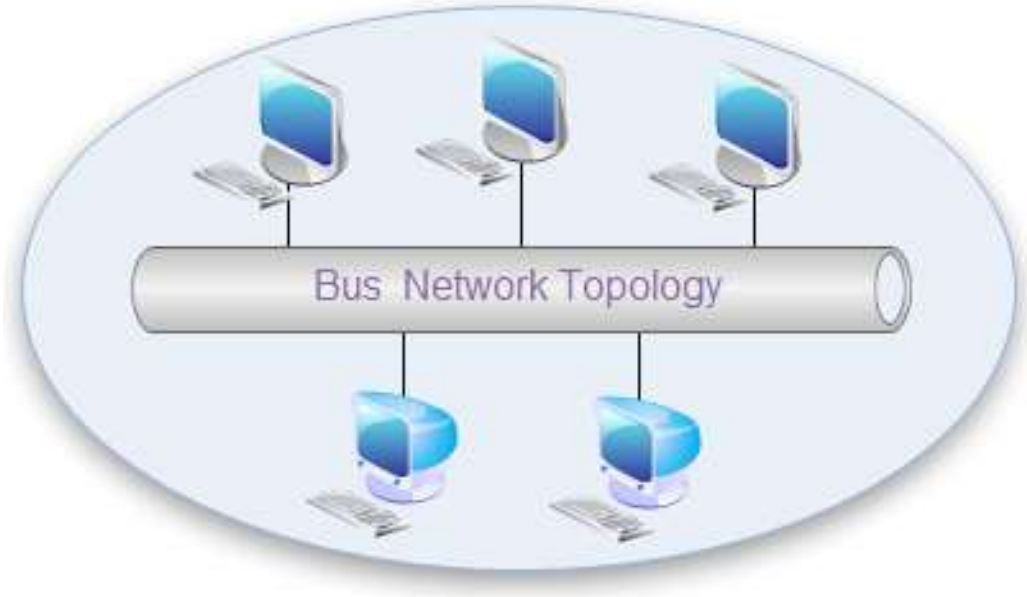
Point to point topology

- This is the simplest of all network topologies
- The network consist of a direct link between two computers



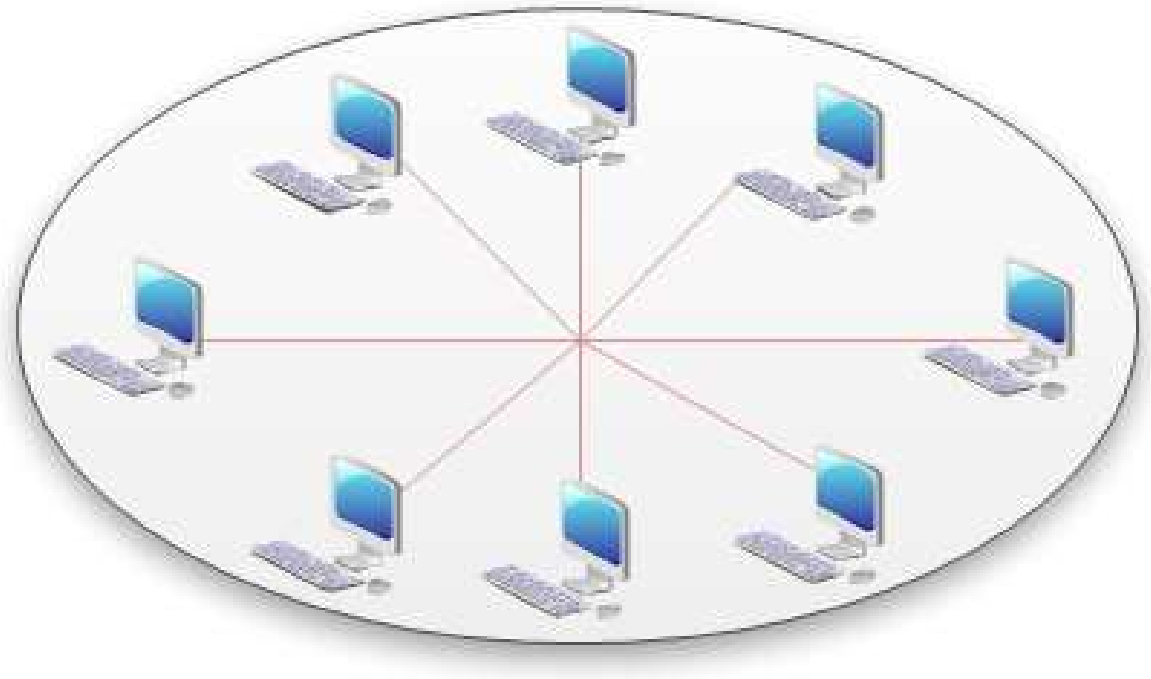
Bus topology

- All the nodes are connected to one main cable
- The main cable acts as the backbone for the network



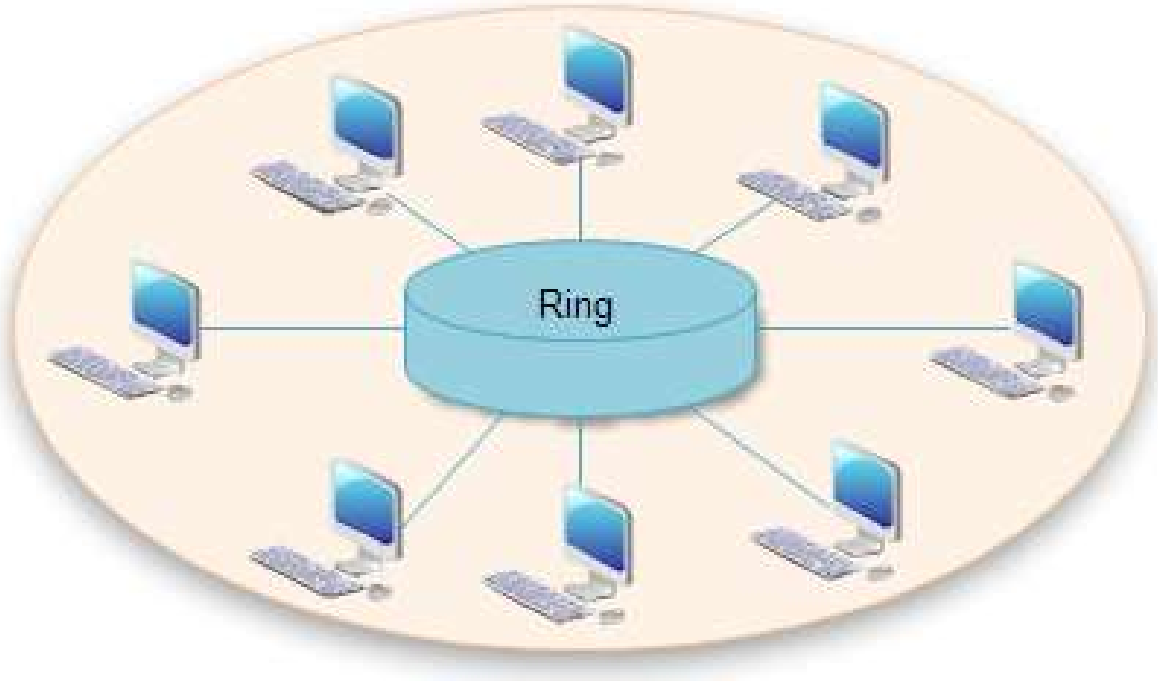
Star topology

- Star topology has a central connection point that can be a hub, switch or router
- Each device in the network is connect to the central point via a point-to- point connection



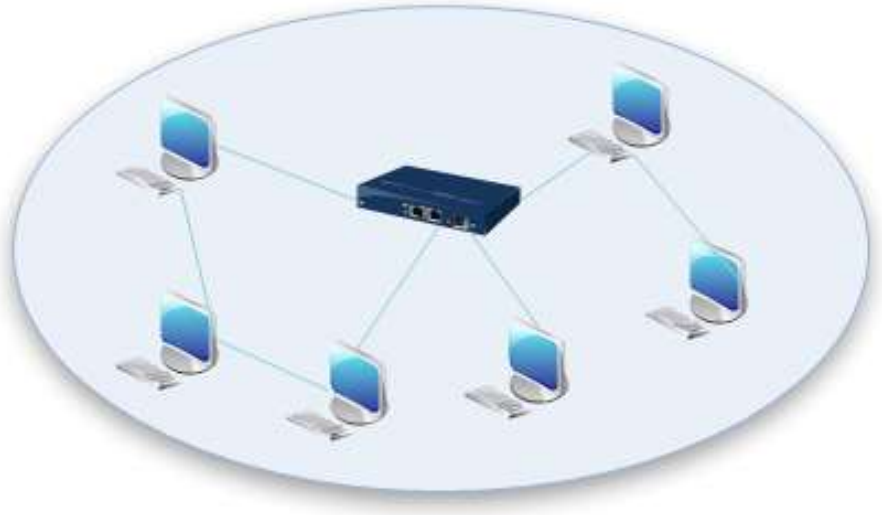
Ring

- The devices in a ring topology are connected in a circular fashion, and the data travel in one direction
- Each device is directly connected to the next device, forming a single path for data through the network



Mesh

- A mesh network is a decentralised network
- Unlike the previous topologies, messages send on a mesh network can take any of several possible paths from the source to the destination



Packets

- When a computer wishes to transmit data to another computer the message is broken up into small pieces called packets
  - A packet is a block of computer data sent and received on a network
- The best way to visualize a packet is to think of it as a letter
- Like a letter a packet has a destination and a return or source address

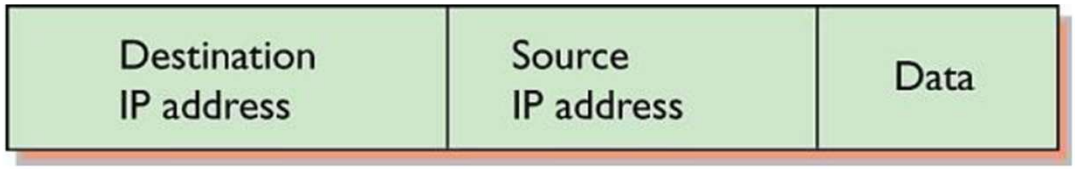
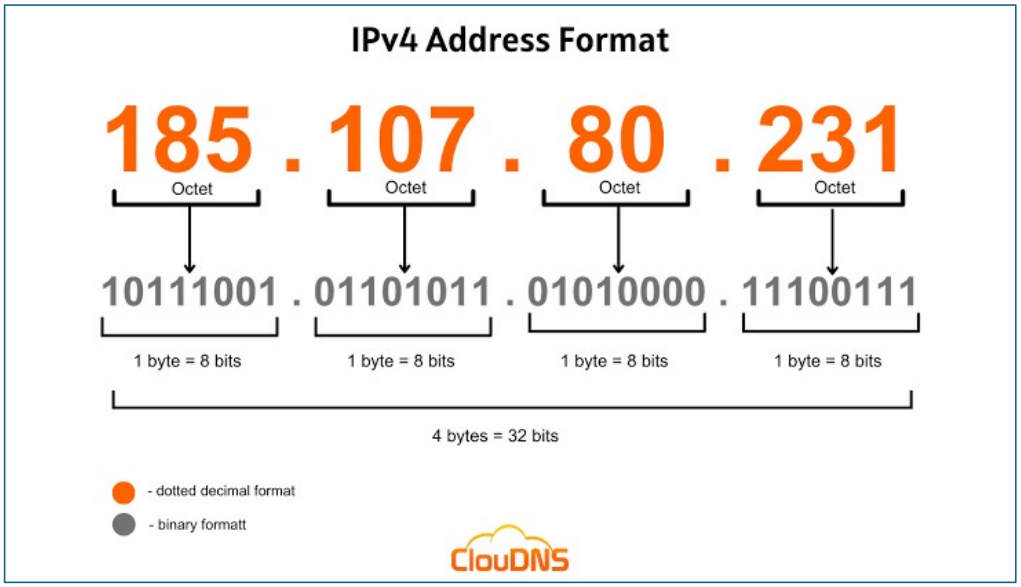


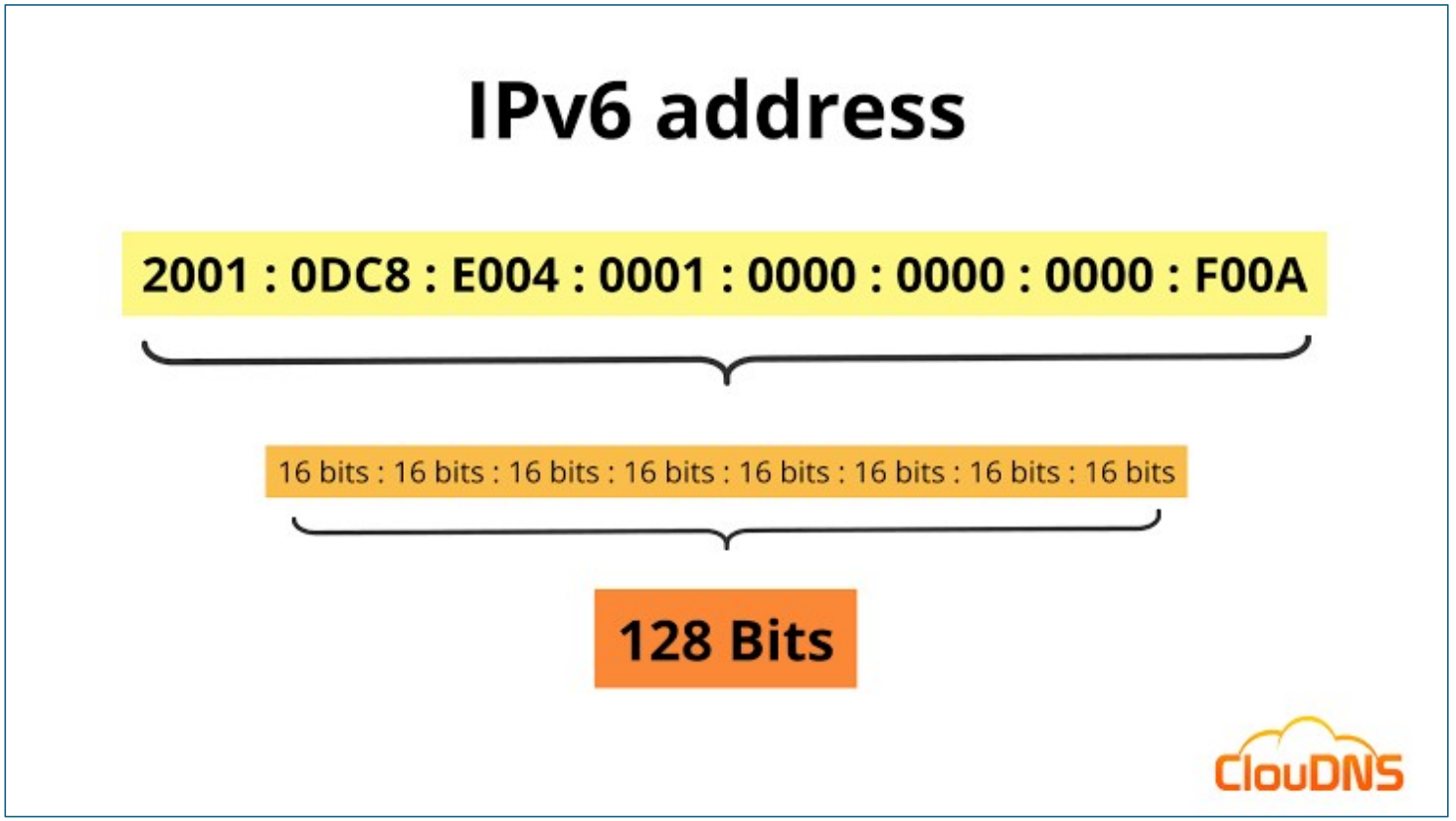
Figure 1.29 IP Packet

IP Addresses

- The packets that travel on a network must have a destination address and a source address
- There are two addressing schemes
  - IPv4: 2^{32} or 4,294,967,296 possible addresses
  - IPv6: 2^{128}, or about 3.403×10^{38} possible addresses



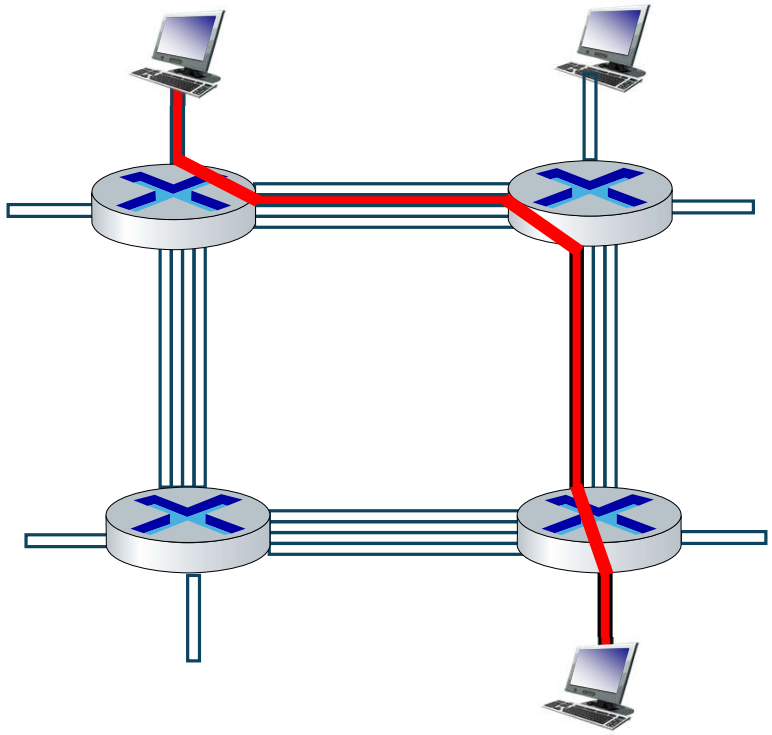
IPv6 address format



Alternative to packet switching vs. circuit switching

end-end resources allocated to, reserved for "call" between source and destination

- in diagram, each link has four circuits.
  - call gets 2nd circuit in top link and 1st circuit in right link.
- dedicated resources: no sharing
  - circuit-like (guaranteed) performance
- circuit segment idle if not used by call (no sharing)
- commonly used in traditional telephone networks



Packet switching versus circuit switching

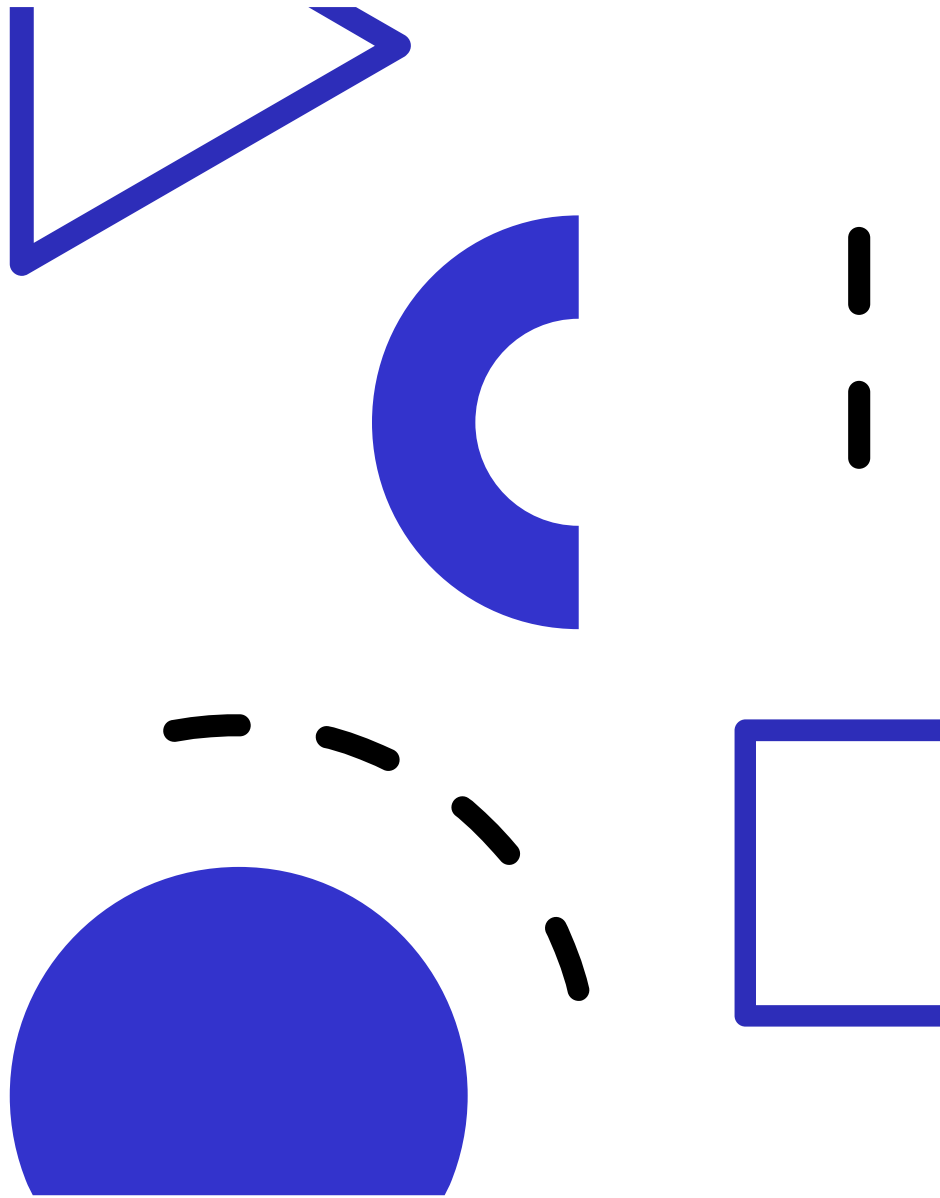
Is packet switching a “slam dunk winner”?

- great for “bursty” data – sometimes has data to send, but at other times not
  - resource sharing
  - simpler, no call setup
- excessive congestion possible: packet delay and loss due to buffer overflow
  - protocols needed for reliable data transfer, congestion control
- *Q:* How to provide circuit-like behavior?
  - bandwidth guarantees traditionally used for audio/video applications

Internet structure: a “network of networks”

- Hosts connect to Internet via access Internet Service Providers (ISPs)
  - residential, enterprise (company, university, commercial) ISPs
- Access ISPs in turn must be interconnected
  - so that any two hosts can send packets to each other
- Resulting network of networks is very complex
  - evolution was driven by economics and national policies
- Let’s take a stepwise approach to describe current Internet structure

The OSI Model



Understanding the OSI Model

- **Open Systems Interconnection (OSI)** model
  - Presented in 1984 by the **International Organization for Standardization (ISO)**
  - Based on examination of existing protocols, ISO recommended a seven-layer network model
  - Allows vendors to implement networks that permit communication among the wide variety of network implementations
- The OSI model is not an absolute standard for computer networks
  - Used as a reference model

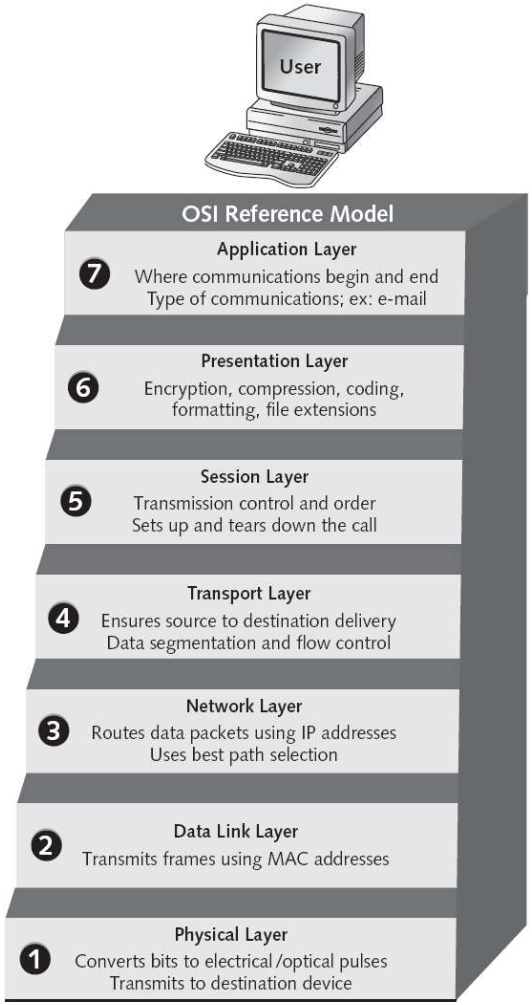


Figure 1-2 Layer Functions

Reasons for Layering

- Advantages
  - Simplifies the networking model
  - Enables programmers to specialize in a particular level or layer
  - Provides design modularity
  - Encourages interoperability
  - Allows networking vendors to produce standardized interfaces

Peer OSI Communication

- **Peer communication**
  - Each layer will only talk to its peer on the opposite side of the communications process
  - Each layer is unaware of the activities of all other layers of the model
  - Allows error checking to occur on two separate layers simultaneously
- Each layer does provide services to the layer above it and receives services from the layer below it
  - Layers do not acknowledge these services in any way

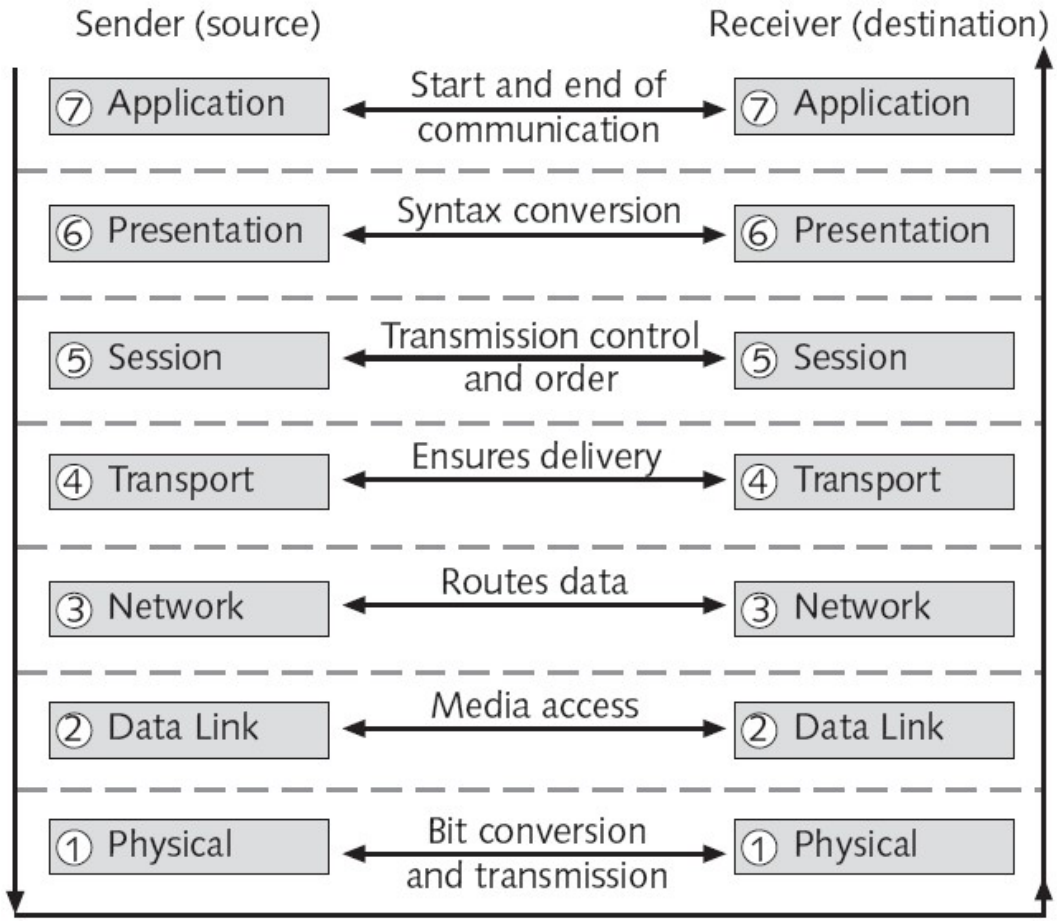


Figure 1-1 OSI reference model

Layer Functions

- The OSI model was developed as an industry standard
  - For companies to use when developing network hardware and software to ensure complete compatibility
- Each layer in the OSI model performs a specific function in the transmission process
- Most modern networks do not implement the OSI model exactly as it is defined

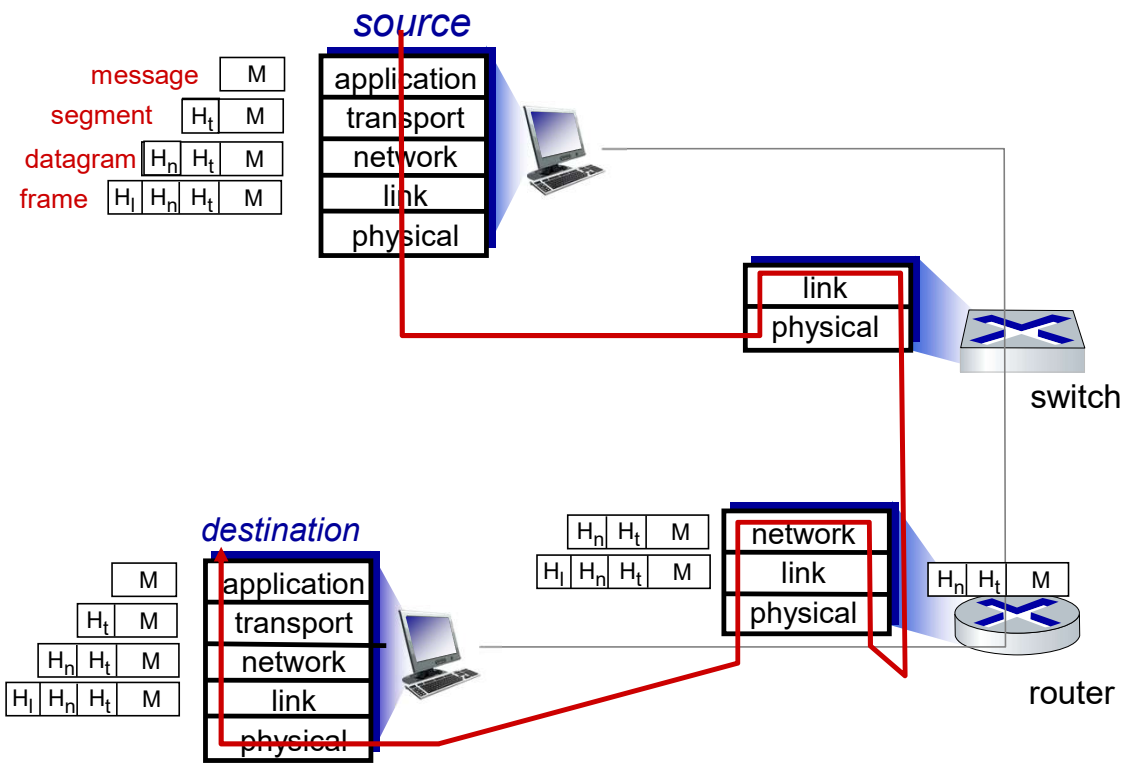
Summary (OSI Model continued)

- The Physical layer handles the physical transmission of data across the network
- The Data Link layer, the second layer of the OSI model, interacts with the networking hardware
- The Network layer supports logical addressing and routing of data packets
- The Transport layer segments data that is to be sent out on the network into MTUs
- The Session layer, the fifth layer, establishes and maintains connections between computers during data transfers
- The Presentation layer, the sixth layer, handles data translation, encryption, and formatting for transmission on the network or for interpretation by the Application layer
- The Application layer, the seventh and highest layer, handles the interface between the network and the user
- When the network user sends data to the network, it goes through a five-step data encapsulation process

Introducing Networks

The TCP/IP Model

Encapsulation

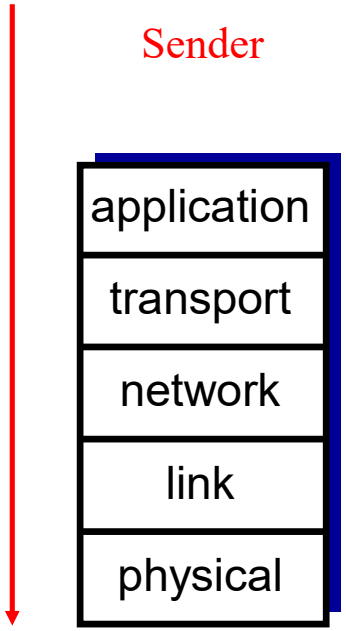


Encapsulation and De-Encapsulation

- Encapsulation: the entire process of preparing data to go onto a network
  - All the steps from the application to the Application, Presentation, Session, Transport, Network, and Data Link layers
- De-encapsulation: the reverse process of encapsulation
  - Stripping all the extra header information out as the data goes up the stack

Internet protocol stack

- *application:* supporting network applications
  - IMAP, SMTP, HTTP (Data)
- *transport:* process-process data transfer
  - TCP, UDP (Segment)
- *network:* routing of datagrams from source to destination
  - IP, routing protocols (Packet)
- *link:* data transfer between neighboring network elements (Frame)
  - Ethernet, 802.11 (Wi-Fi), PPP
- *physical:* bits “on the wire” (Bits)



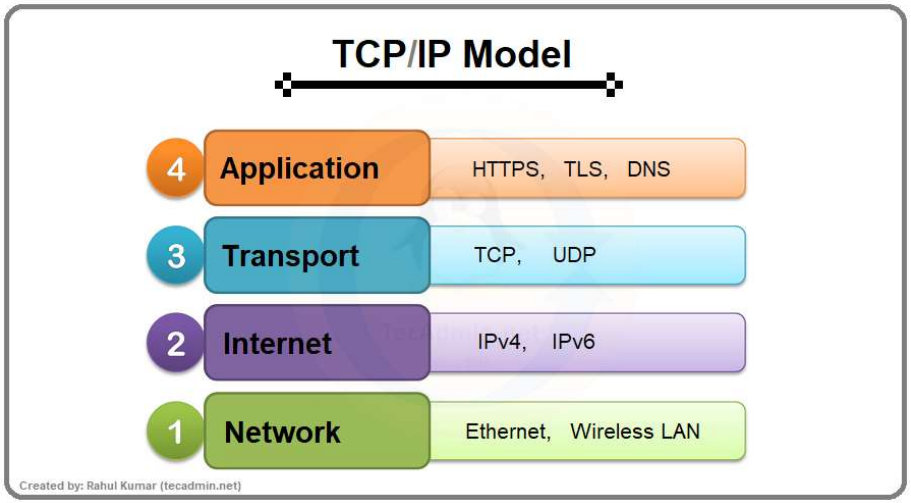
Summary (TCP/IP Model)

Application Layer

- **Purpose**: Provides services and protocols that allow user applications to communicate over the network.
- **Responsibilities**: Data formatting, encryption, session management, end-user interaction

Transport Layer

- **Purpose**: Ensures correct delivery of data between applications.
- **Responsibilities**:
  - End-to-end communication.
  - Port numbers to identify applications.
  - Reliability (acknowledgments, retransmission, sequencing).
- **Protocols**:
  - **TCP (Transmission Control Protocol)** → reliable, connection-oriented.
  - **UDP (User Datagram Protocol)** → fast, connectionless.



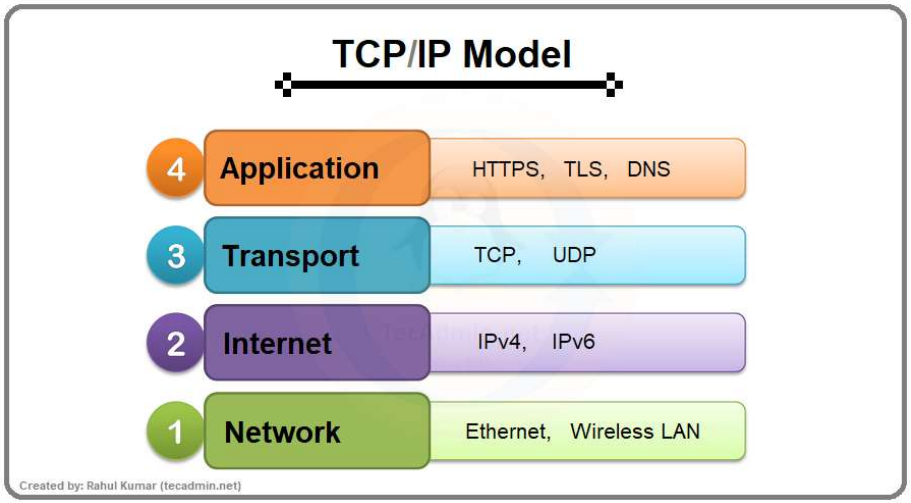
Summary (TCP/IP Model)

Internet Layer

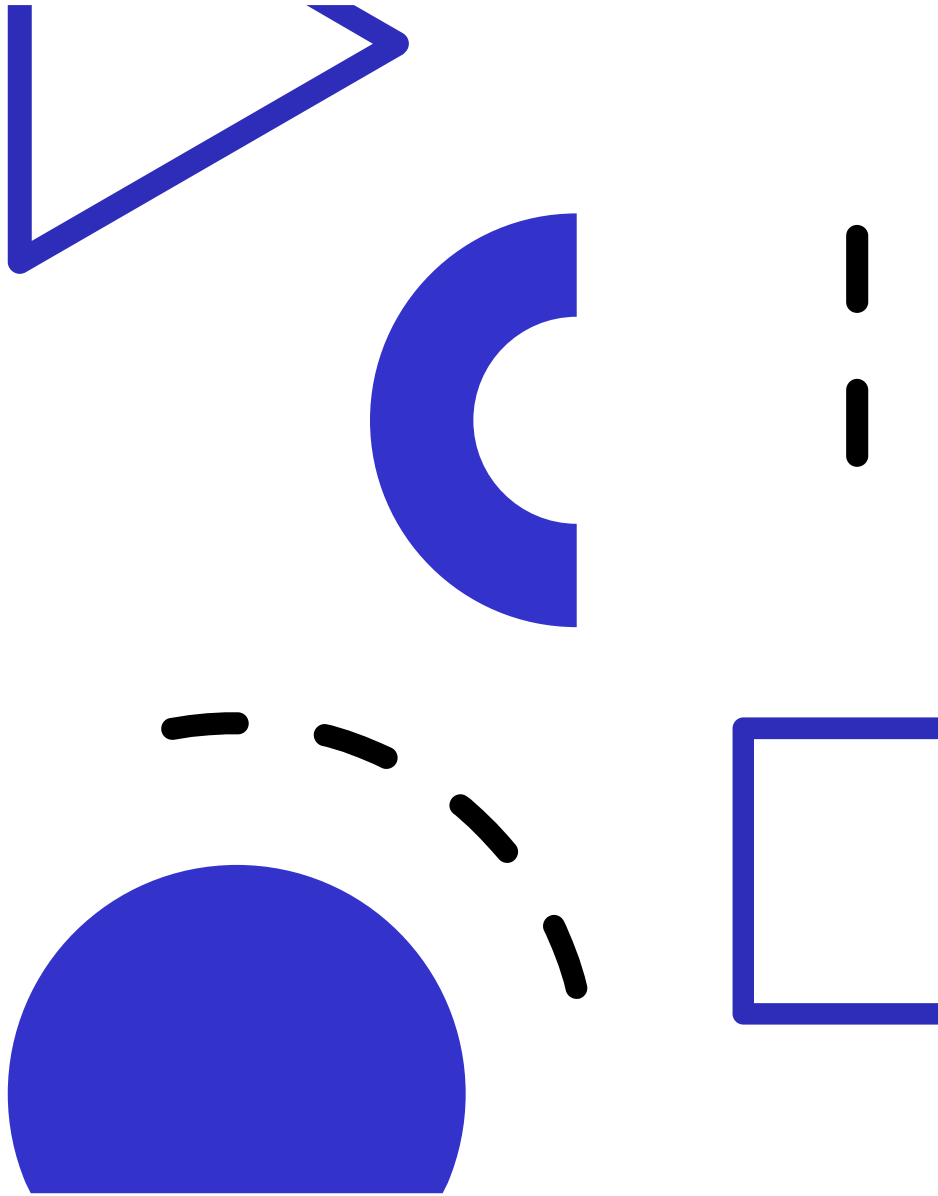
- **Purpose**: Handles logical addressing and routing of data between networks.
- **Responsibilities**:
  - Identifying devices using **IP addresses**.
  - Routing data from source to destination across multiple networks.
- **Protocols**:
  - **IP (IPv4, IPv6), ICMP** (error & diagnostic messages, e.g., ping)
  - **ARP** (maps IP to MAC address), **IGMP** (manages multicast groups)

Network Access Layer (Link Layer)

- **Purpose**: Moves data across the physical network (local delivery).
- **Responsibilities**:
  - Framing data with **MAC addresses**.
  - Error detection at the link level.
  - Physical transmission of bits over copper, fiber, or wireless.
- **Examples**: Ethernet, Wi-Fi, PPP, DSL.



Application & Presentation Layers



Application layer: overview

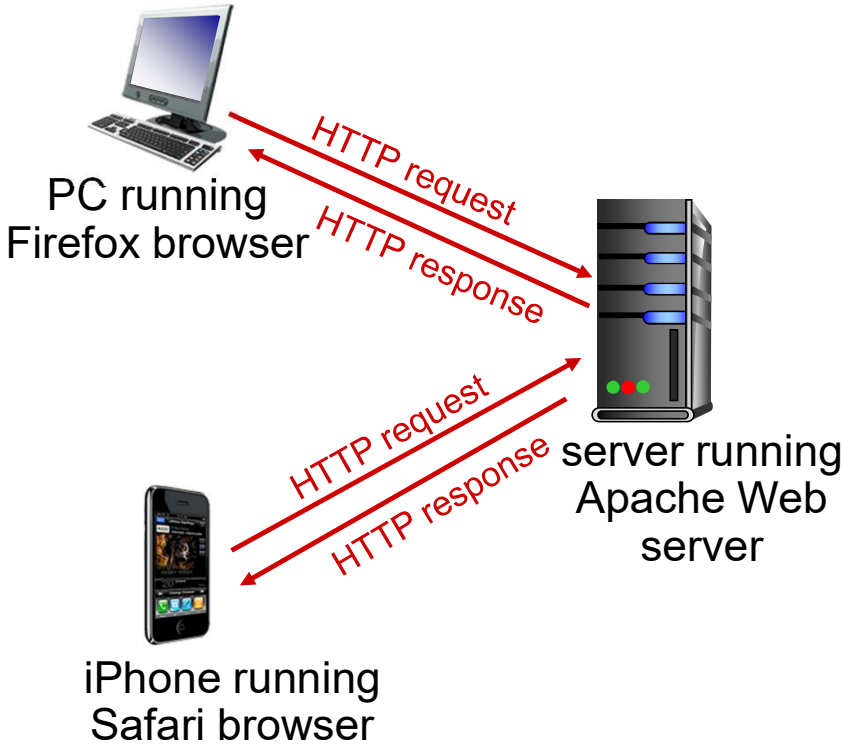
- Principles of network applications
- Web and HTTP
- E-mail, SMTP, IMAP
- The Domain Name System (DNS)
  - Dynamic Host Configuration Protocol (DHCP)
- Network Security
- Encryption
- Tunneling
- Virtual Private Networks

Web Services & HTTP

HTTP overview

HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server model:
  - *client:* browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - *server:* Web server sends (using HTTP protocol) objects in response to requests



HTTP connections: two types

Non-persistent HTTP

1. TCP connection opened
2. at most one object sent over TCP connection
3. TCP connection closed

downloading multiple objects required multiple connections

Persistent HTTP

- TCP connection opened to a server
- multiple objects can be sent over *single* TCP connection between client, and that server
- TCP connection closed

Conditional GET

Goal: don't send object if cache has up-to-date cached version

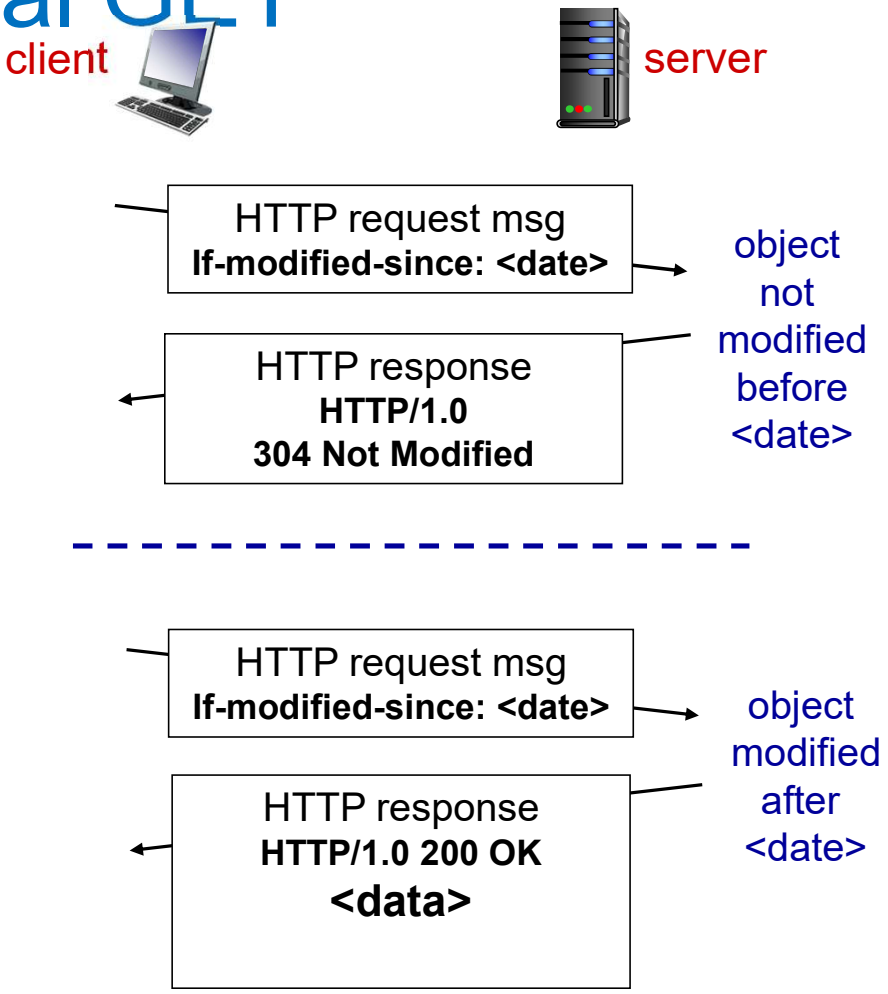
- no object transmission delay
- lower link utilization

- *cache:* specify date of cached copy in HTTP request

  If-modified-since: <date>

- *server:* response contains no object if cached copy is up-to-date:

  HTTP/1.0 304 Not Modified



E-mail, SMTP, IMAP

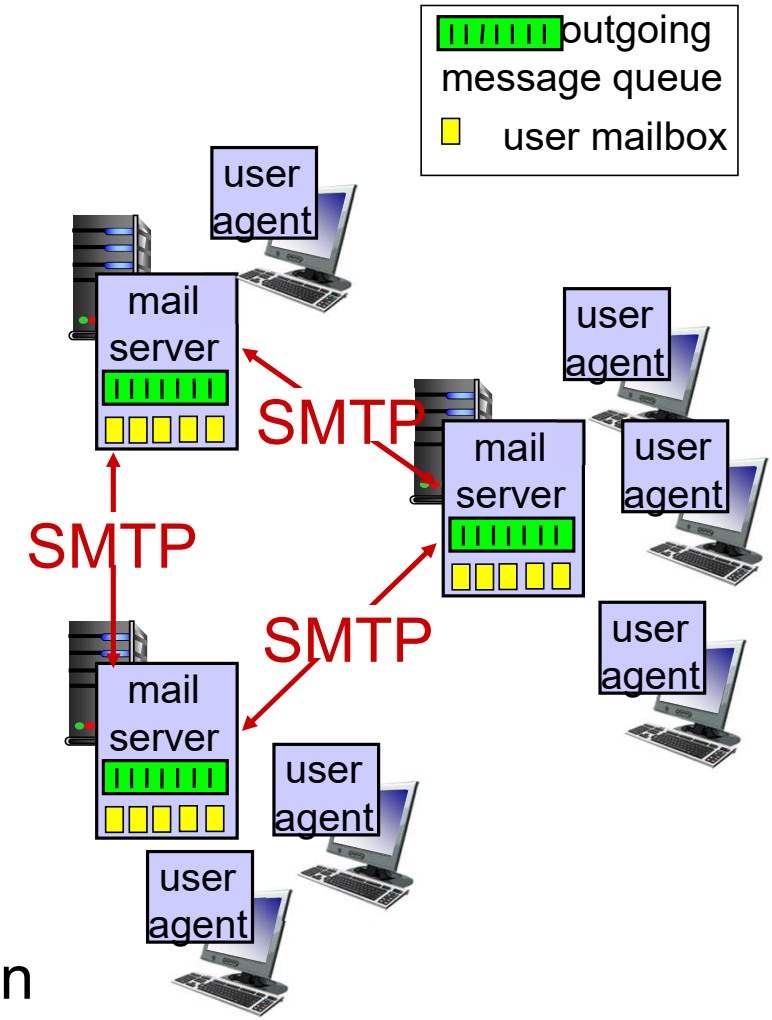
E-mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

User Agent

- a.k.a. “mail reader”
- composing, editing, reading mail messages
- e.g., Outlook, iPhone mail client
- outgoing, incoming messages stored on server



E-mail: the RFC (5321)

- uses TCP to reliably transfer email message from client (mail server initiating connection) to server, port 25
- direct transfer: sending server (acting like client) to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase
- messages must be in 7-bit ASCI

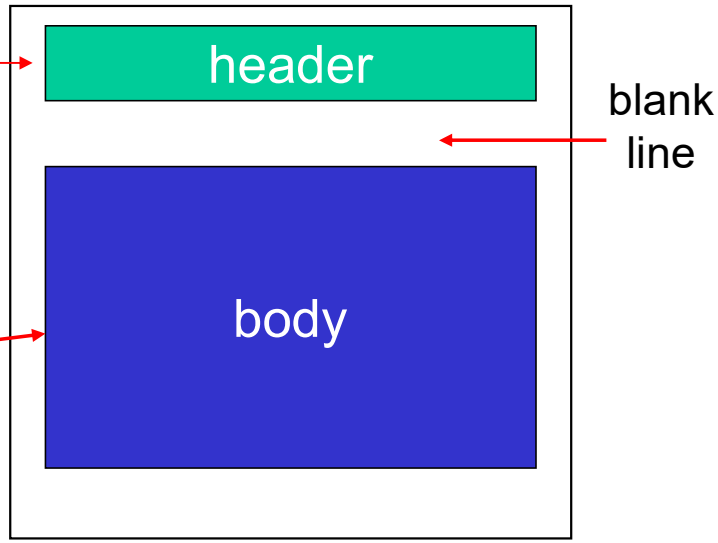
Mail message format

SMTP: protocol for exchanging e-mail messages, defined in RFC 531 (like HTTP)

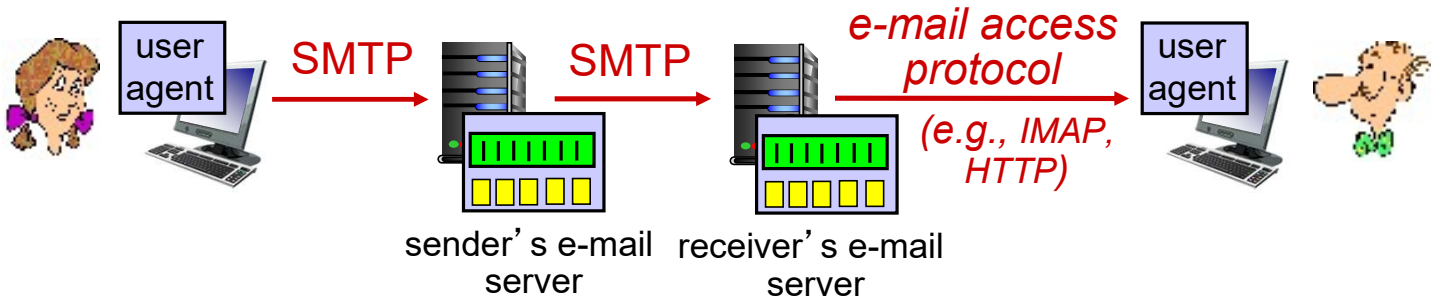
RFC 822 defines *syntax* for e-mail message itself (like HTML), Replaced by RFC 5322

- header lines, e.g.,
  - To:
  - From:
  - Subject:

  these lines, within the body of the email message area different from SMTP MAIL FROM:, RCPT TO: commands!
- Body: the "message", ASCII characters only



Mail access protocols



- SMTP: delivery/storage of e-mail messages to receiver's server
- mail access protocol: retrieval from server
  - IMAP: Internet Mail Access Protocol [RFC 3501]: messages stored on server, IMAP provides retrieval, deletion, folders of stored messages on server
- HTTP: gmail, Hotmail, Yahoo!Mail, etc. provides web-based interface on top of STMP (to send), IMAP (or POP) to retrieve e-mail messages

Domain Name Systems (DNS) Services

DNS: services, structure

DNS services

- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

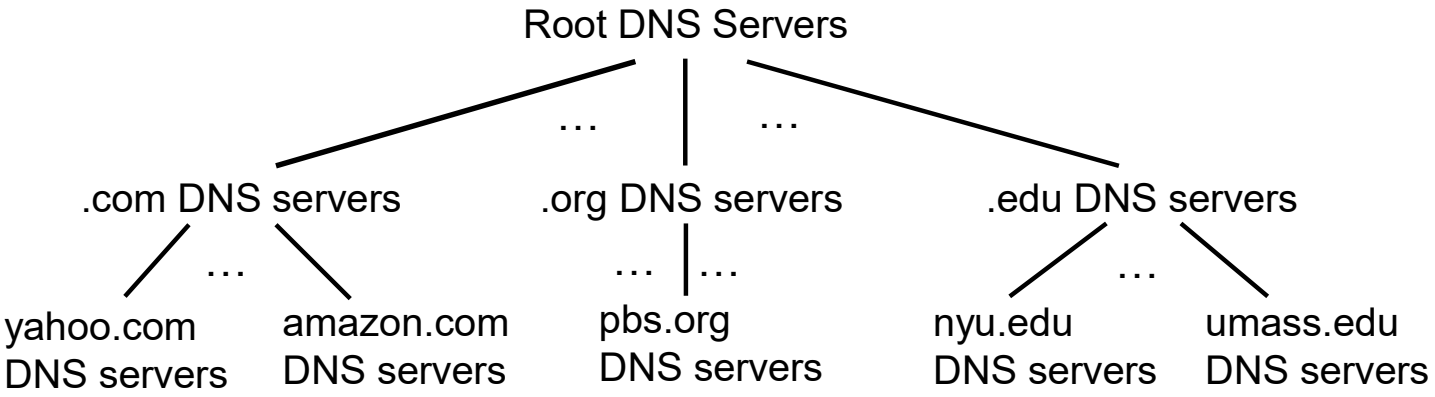
Q: Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance

A: doesn't scale!

- Comcast DNS servers alone: 600B DNS queries per day

DNS: a distributed, hierarchical database



Client wants IP address for www.amazon.com; 1st approximation:

- client queries root server to find .com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

TLD: authoritative servers

Top-Level Domain (TLD) servers:

- responsible for .com, .org, .net, .edu, .aero, .jobs, .museums, and all top-level country domains, e.g.: .cn, .uk, .fr, .ca, .jp
- Network Solutions: authoritative registry for .com, .net TLD
- Educause: .edu TLD

Authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

DNS security

DDoS attacks

- bombard root servers with traffic
  - not successful to date
  - traffic filtering
  - local DNS servers cache IPs of TLD servers, allowing root server bypass
- bombard TLD servers
  - potentially more dangerous

Redirect attacks

- man-in-middle
  - intercept DNS queries
- DNS poisoning
  - send bogus relies to DNS server, which caches

Exploit DNS for DDoS

- send queries with spoofed source address: target IP
- requires amplification

DNSSEC [RFC 4033]

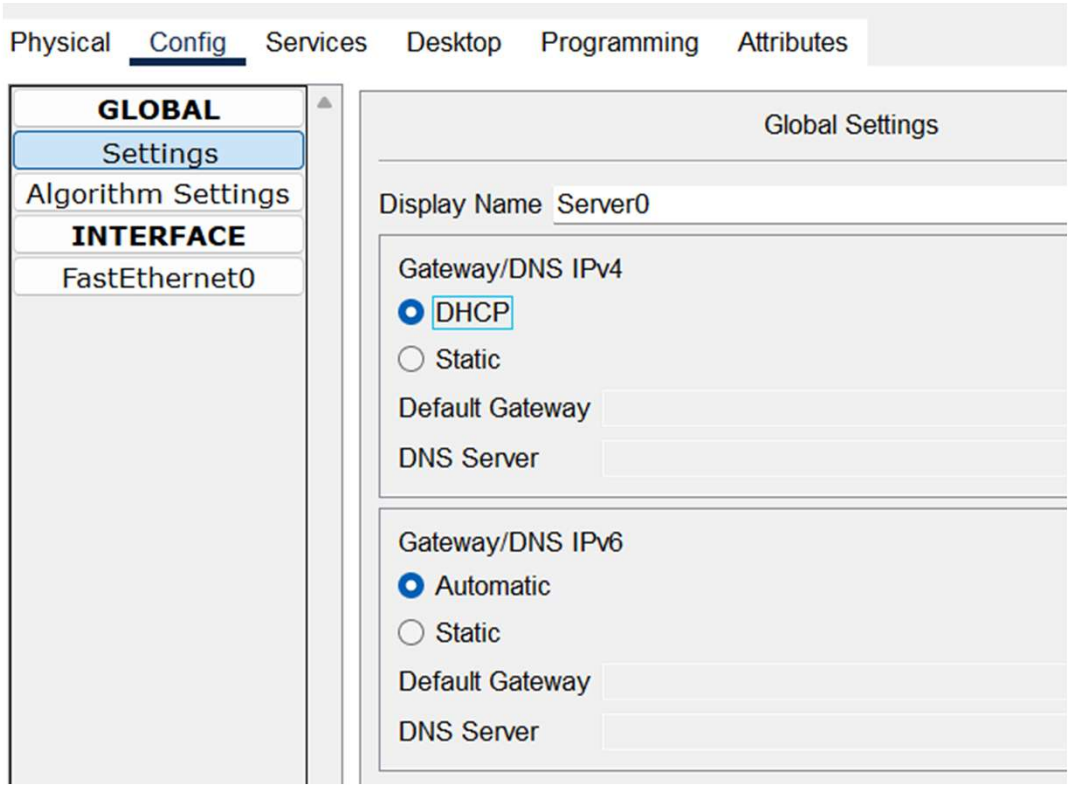
Dynamic Host Configuration Protocol (DHCP)

Dynamic Host Configuration Protocol (DHCP)

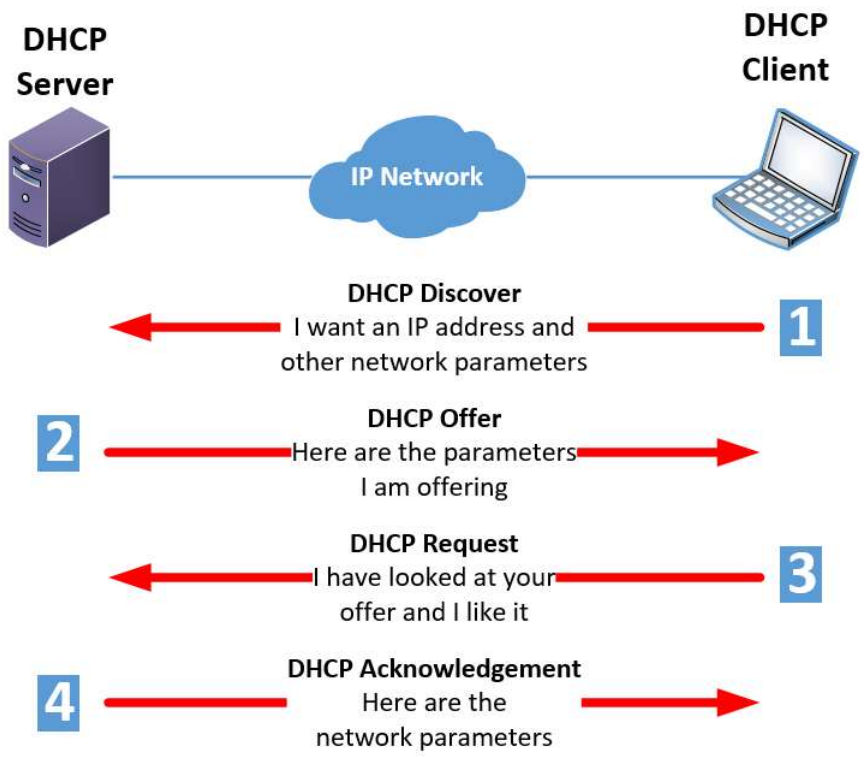
DHCP operates primarily at the Application Layer of the TCP/IP model

DHCP automatically assigns IP addresses and other network configuration parameters

- Subnet mask,
- Default gateway, and
- DNS server
- DHCP uses **UDP** as its transport protocol.
  - Server Port: **UDP 67**
  - Client Port: **UDP 68**



Dynamic Host Configuration Protocol (DHCP)



Key Functions of DHCP:

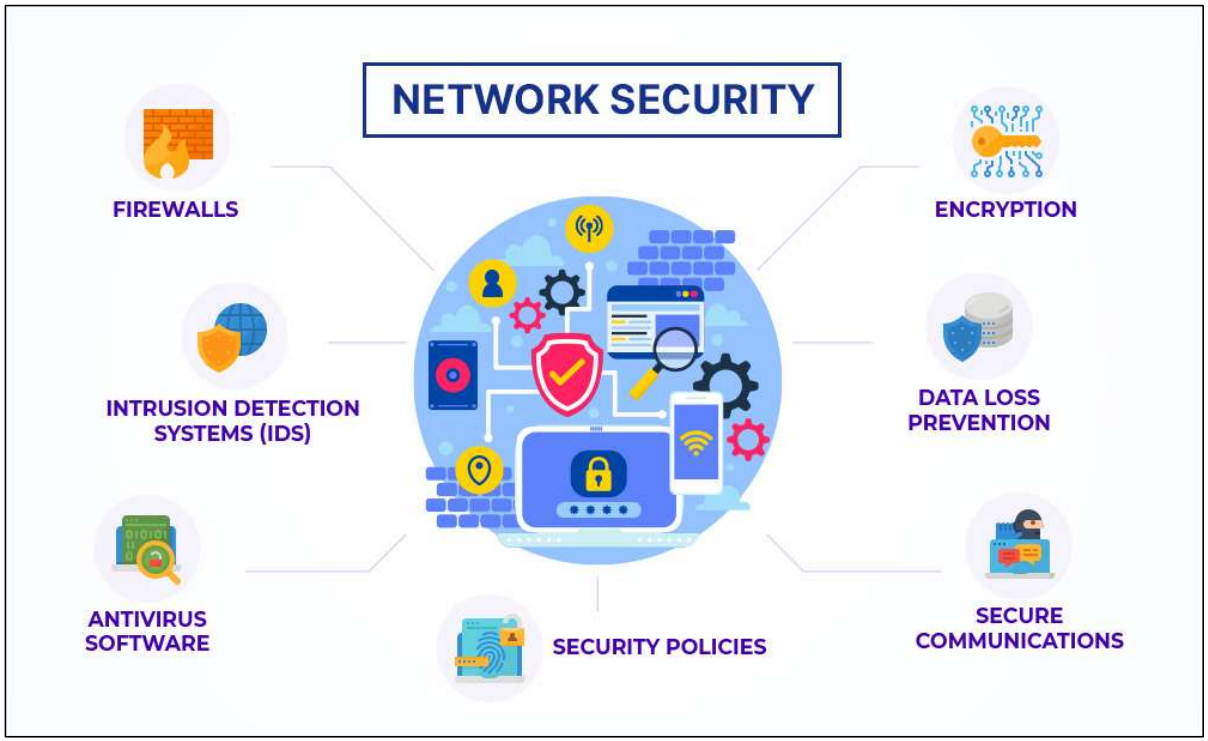
- **IP Address Assignment**: Dynamically assigns IP addresses to clients.
- **Lease Management**: Assigns IP addresses for a specific lease time, after which they may be renewed or reassigned.
- **Configuration Distribution**: Sends other necessary configuration data (gateway, DNS, etc.).
- **Central Management**: Reduces the risk of IP conflicts and simplifies administration.

Encryption, Tunneling, and VPNs

Securing Communication in Modern Networks

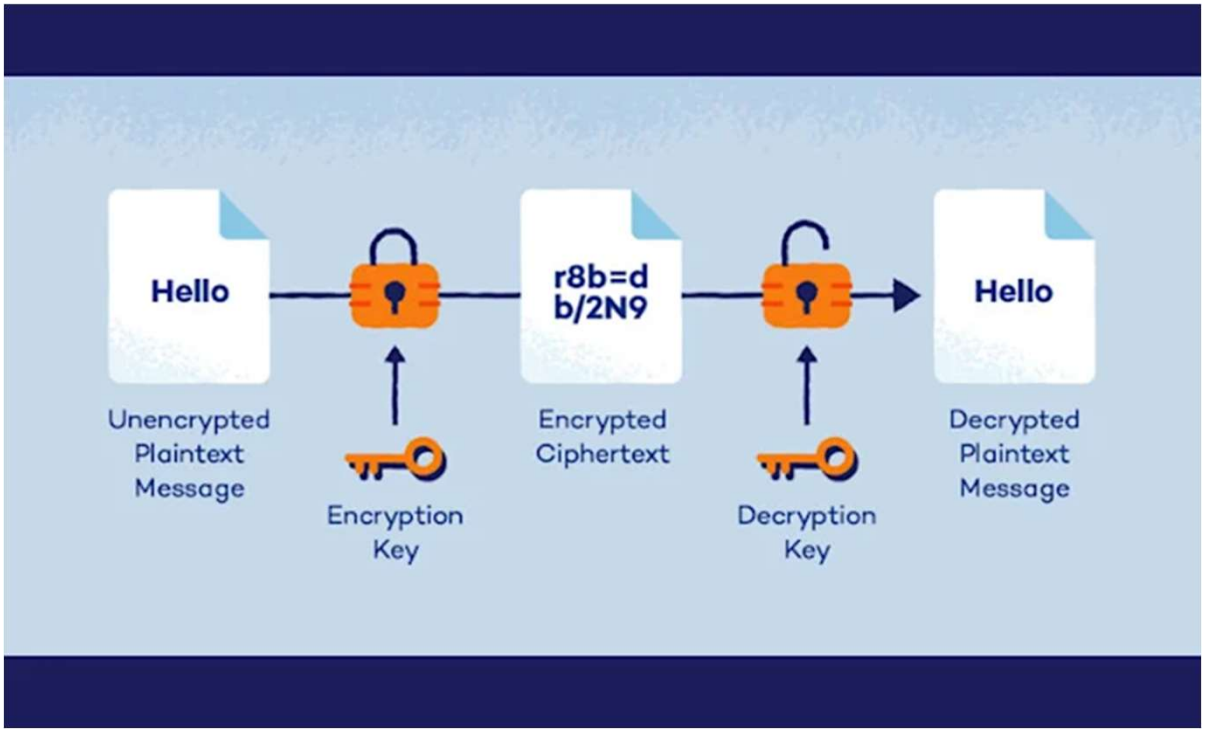
Introduction

- Importance of network security
- Overview of the three core concepts:
  - Encryption
  - Tunneling
  - VPNs



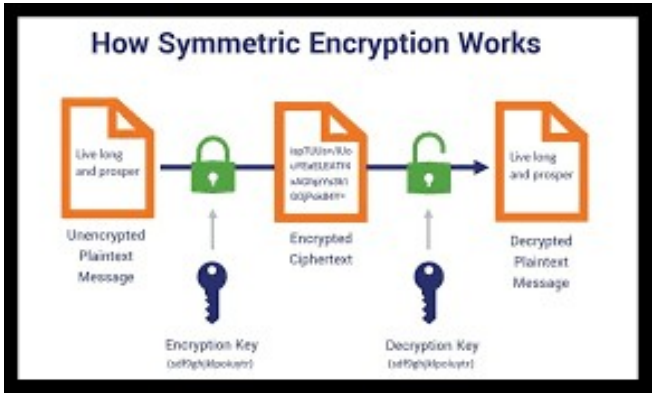
How Encryption Works

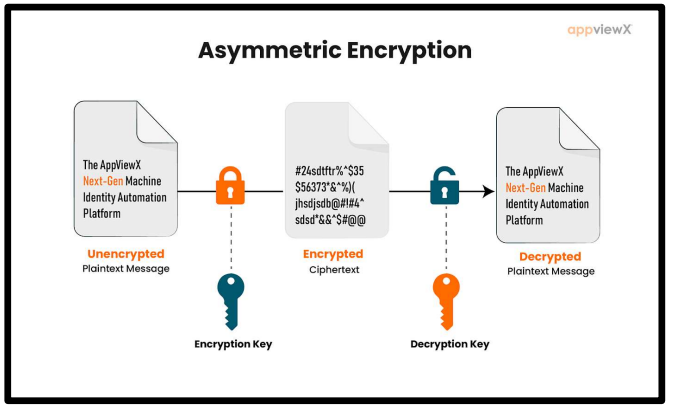
- Plaintext → Ciphertext
- Keys:
- Encryption key vs Decryption key

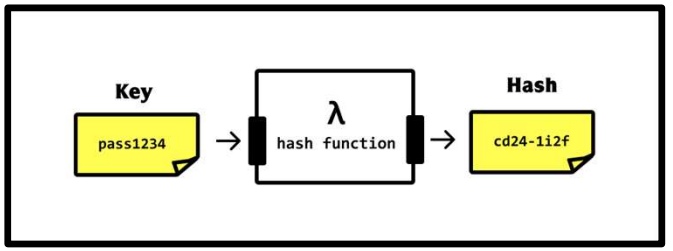


What is Encryption?

- **Definition:** Process of converting data into a coded form
- **Purpose:** Confidentiality, data integrity, authentication
- **Types:**
  - Symmetric (e.g., AES)
  - Asymmetric (e.g., RSA)
  - Hashing
- Used in digital signatures, password protections

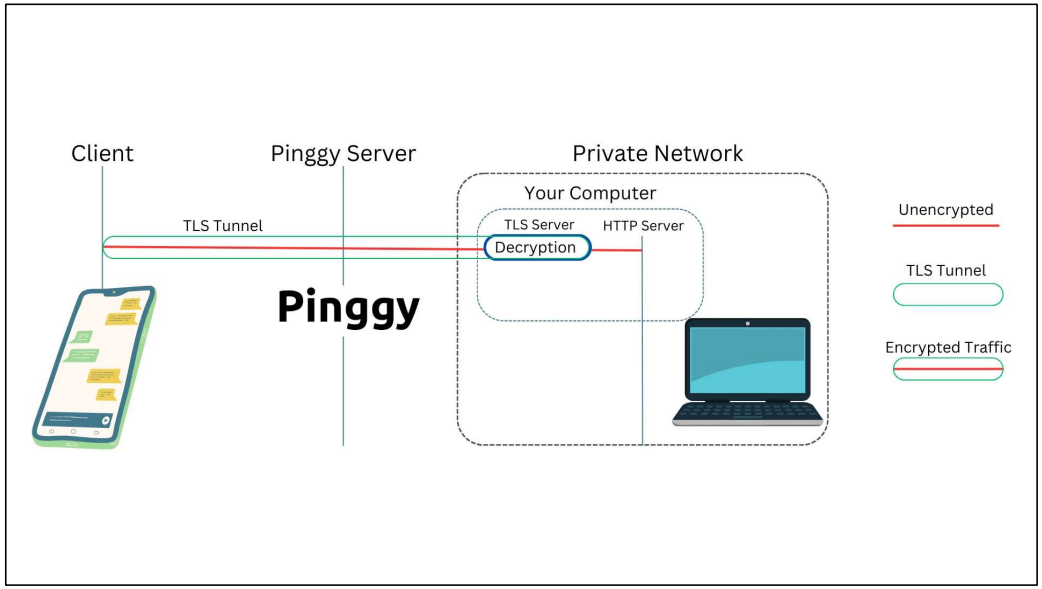






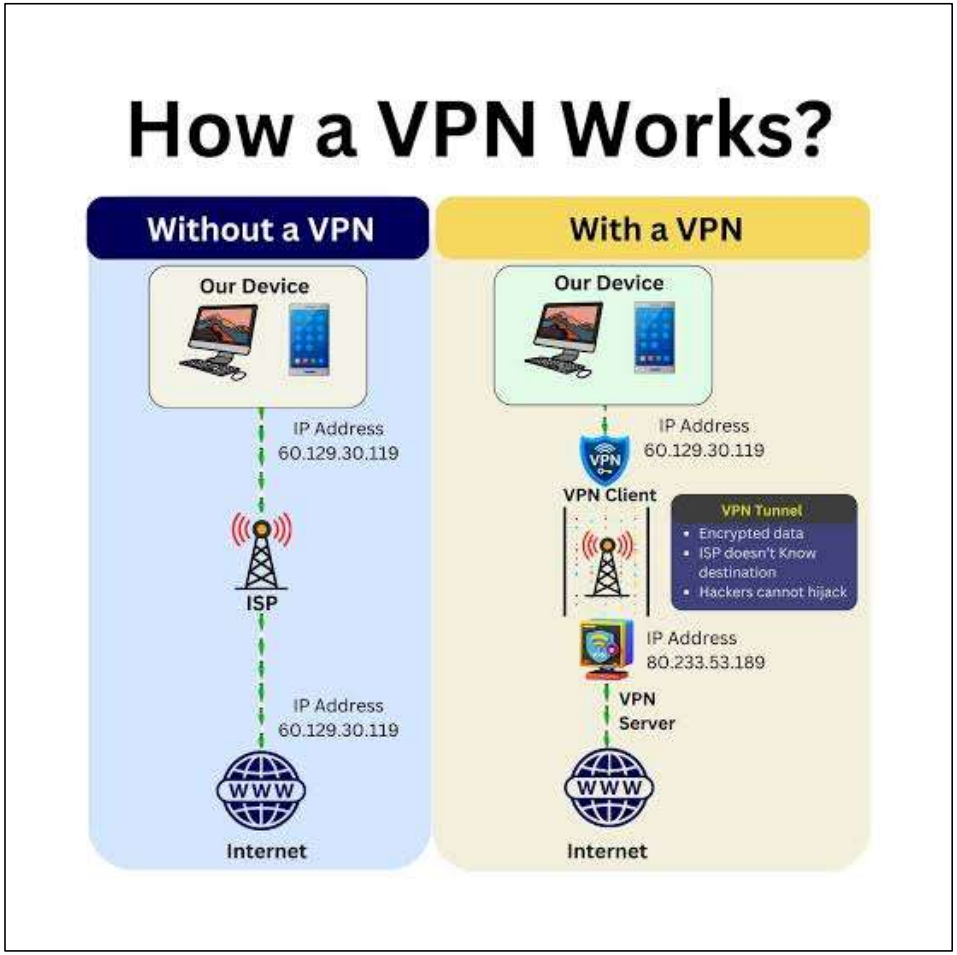
What is a VPN?

- Definition: Virtual Private Network
- Combines encryption and tunneling
- Purpose:
  - Secure remote access
  - Privacy and anonymity
  - Bypassing geo-restrictions



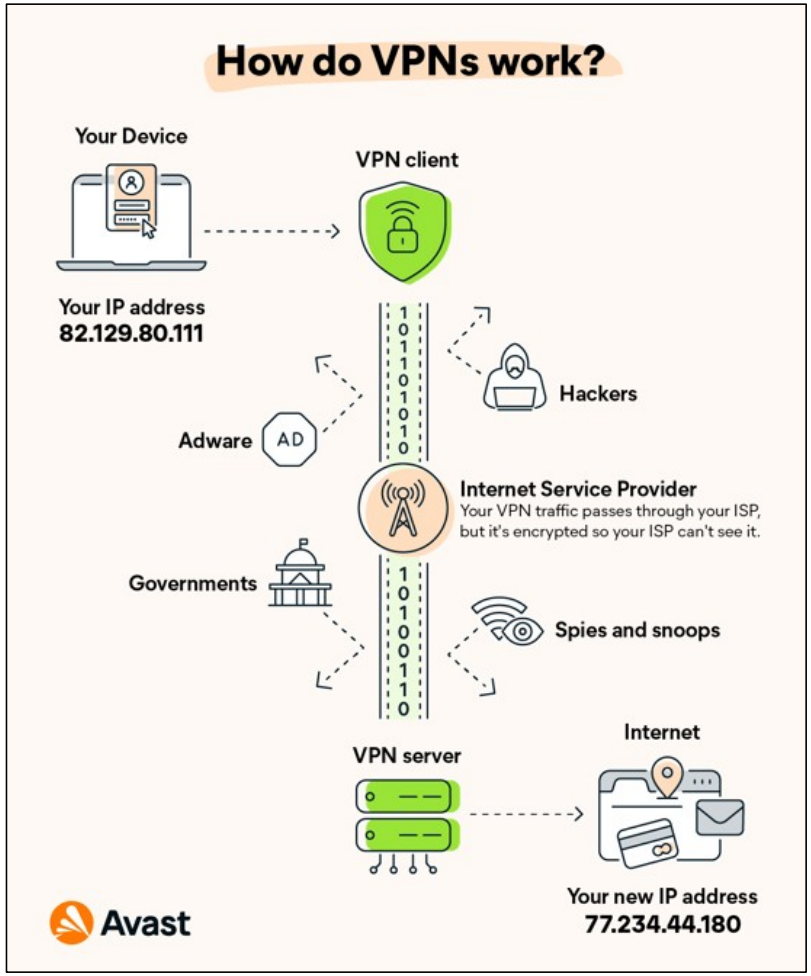
How VPNs Work

- Diagram: Client → Encrypted Tunnel → VPN Server → Internet
- Key elements:
  - VPN client
  - VPN server
  - Tunnel & encryption layer



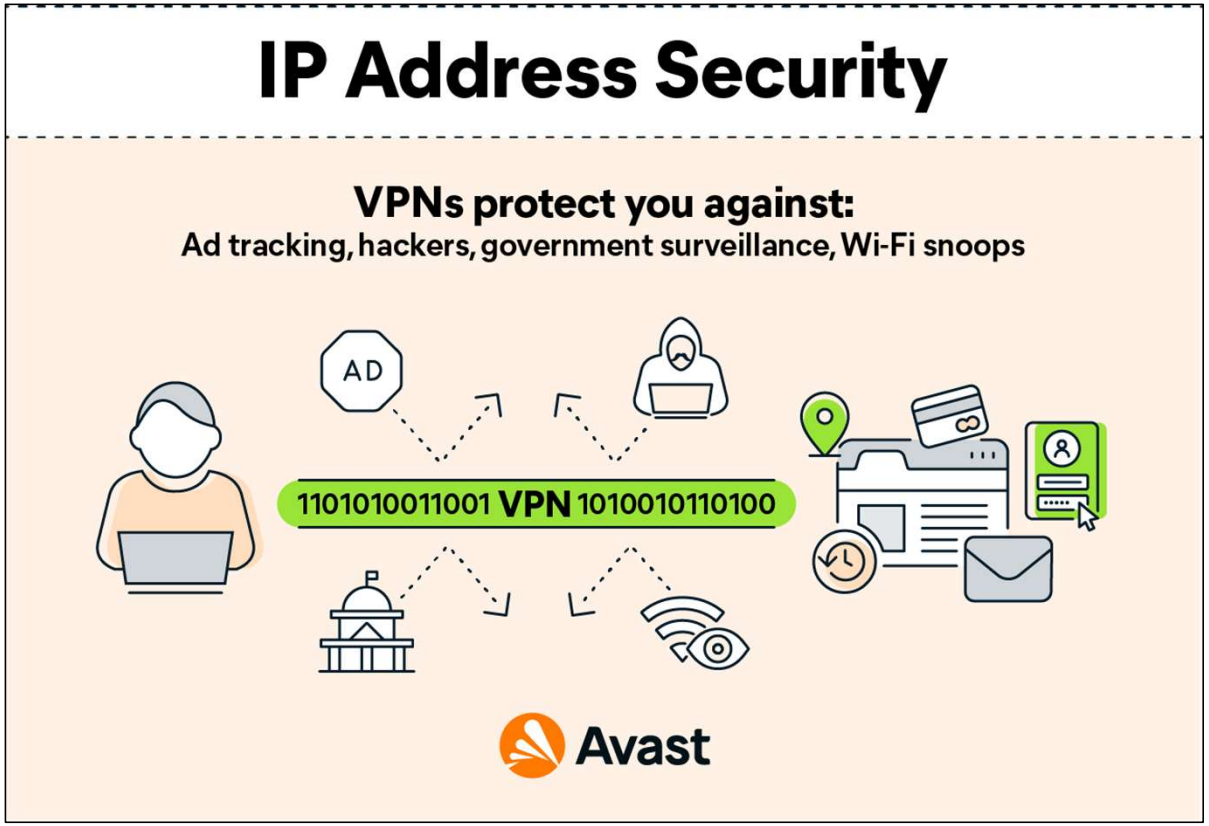
Benefits of Using a VPN

- Data security on public Wi-Fi
- Masking IP address
- Accessing region-restricted content
- Avoiding censorship



Section Recap

- Encryption secures data
- Tunneling hides the path
- VPNs combine both to secure communication



TCP/IP Model
Transport Layer

Transport layer: Roadmap

Transport-layer services

- Connectionless transport: UDP
- Connection-oriented transport: TCP
- Principles of congestion control
- TCP congestion control
- Evolution of transport-layer functionality

Transport Layer Services

- This is the third layer from the bottom as is responsible for the overall transfer of data and helps establish an end-to-end logical connectivity between the source & destination host and the devices in a network.
- Two protocols are used to perform these tasks:
  - First is the Transmission control protocol (TCP), which is a connection-based and reliable protocol.
  - Second is the User datagram protocol (UDP), which is a connection-less protocol.
- Before exploring these two protocols in deep, we will discuss the concept of PORT NUMBER which is used by both these protocols.

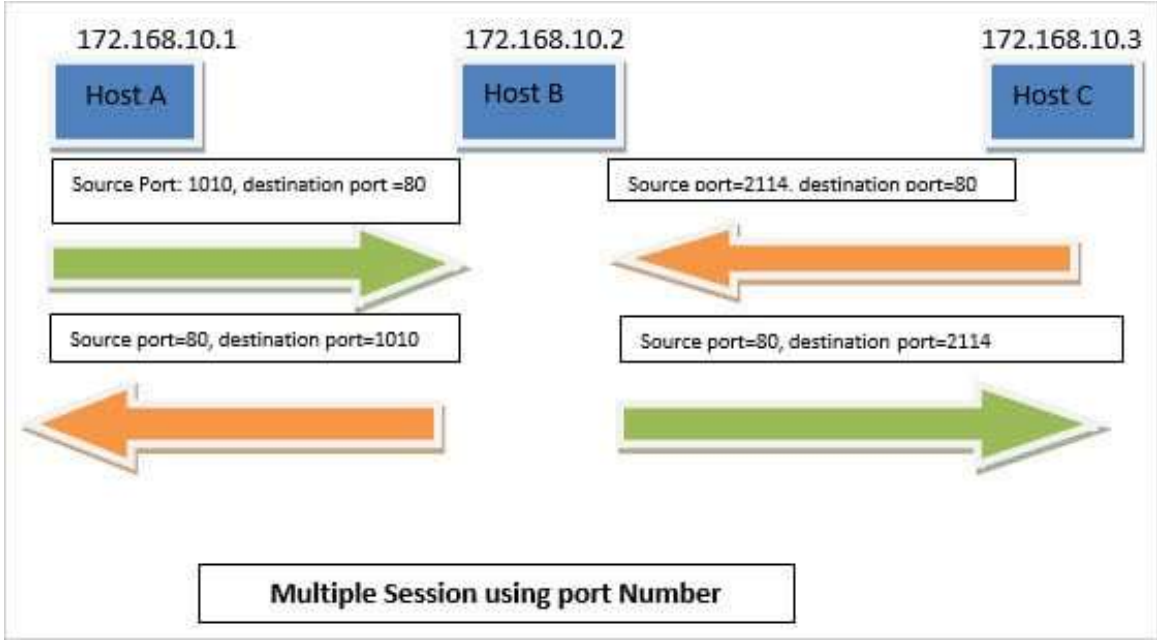
Transport Layer – Port Numbers

- In a network, a host device may send or receive traffic from several sources at the same time.
- In such a situation, the system will not recognize which of the applications the data belongs to.
- TCP and UDP protocols resolve these issues by putting a port number in their headers. The well-known application layer protocols are allocated with the port number in the range 1 to 1024.
- At the source end, every TCP or UDP session is allocated with a random port number.
- The IP address, port number, and type of protocol used in combination reforms a socket at both the source and destination end.
- As every socket is exclusive, several hosts can send or receive traffic at the same interval of time.
- The next slide table shows the port number that is assigned to several application layer protocols corresponding to the transport layer protocol.

Transport Layer – Port Numbers

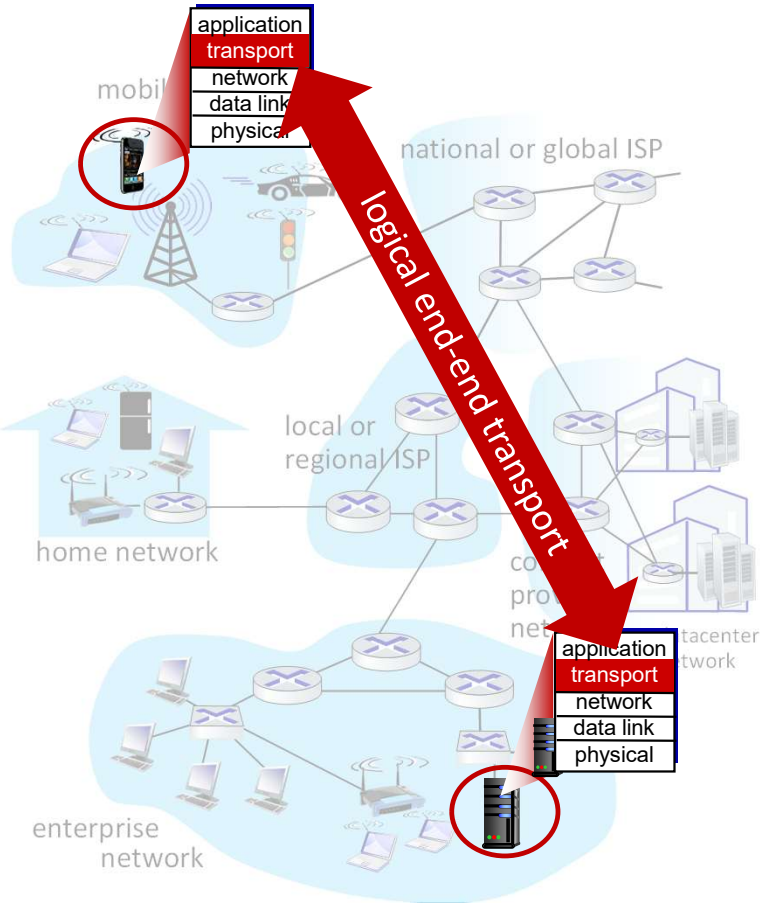
Application Protocol	Transport Protocol	Port Number
HTTP	TCP	80
HTTPS	TCP	443
FTP(control)	TCP	21
FTP(data)	TCP	20
SSH	TCP	22
Telnet	TCP	23
DNS	TCP, UDP	53
SMTP	TCP	25
TFTP	UDP	69

Transport Layer -
Multiple Sessions using port numbers



Transport services and protocols

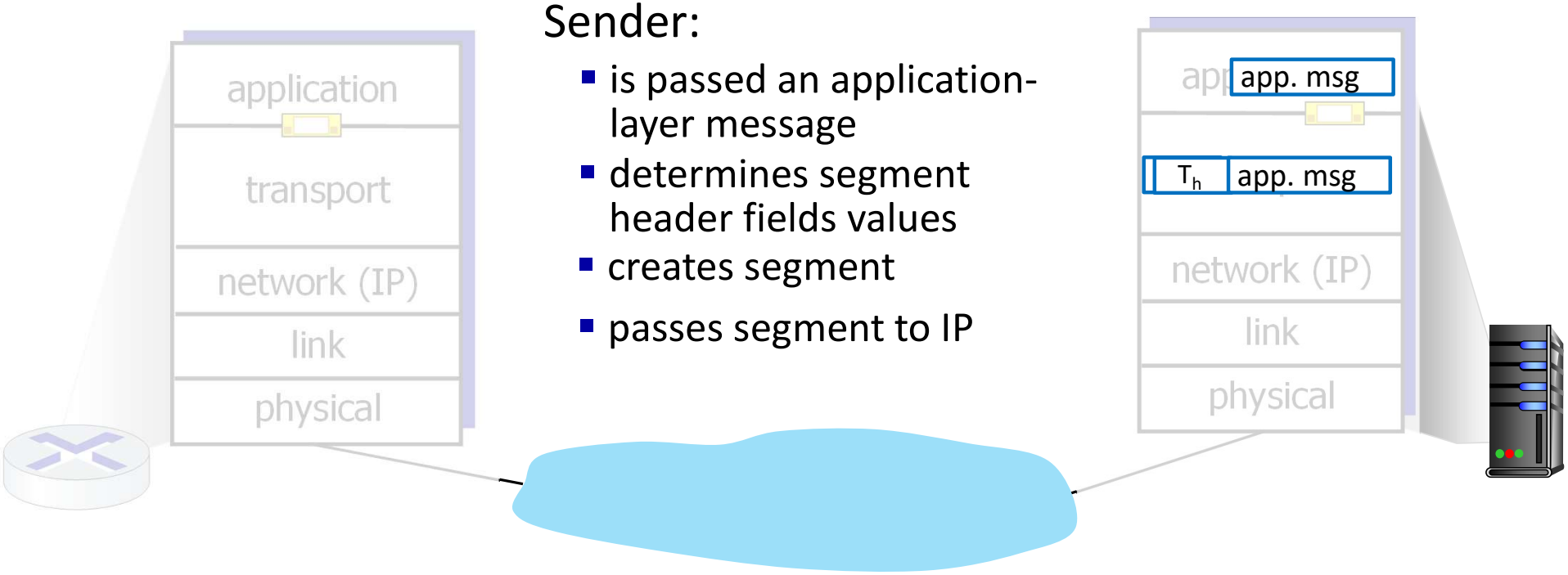
- provide *logical communication* between application processes running on different hosts
- transport protocols actions in end systems:
  - sender: breaks application messages into *segments*, passes to network layer
  - receiver: reassembles segments into messages, passes to application layer
- two transport protocols available to Internet applications
  - TCP, UDP



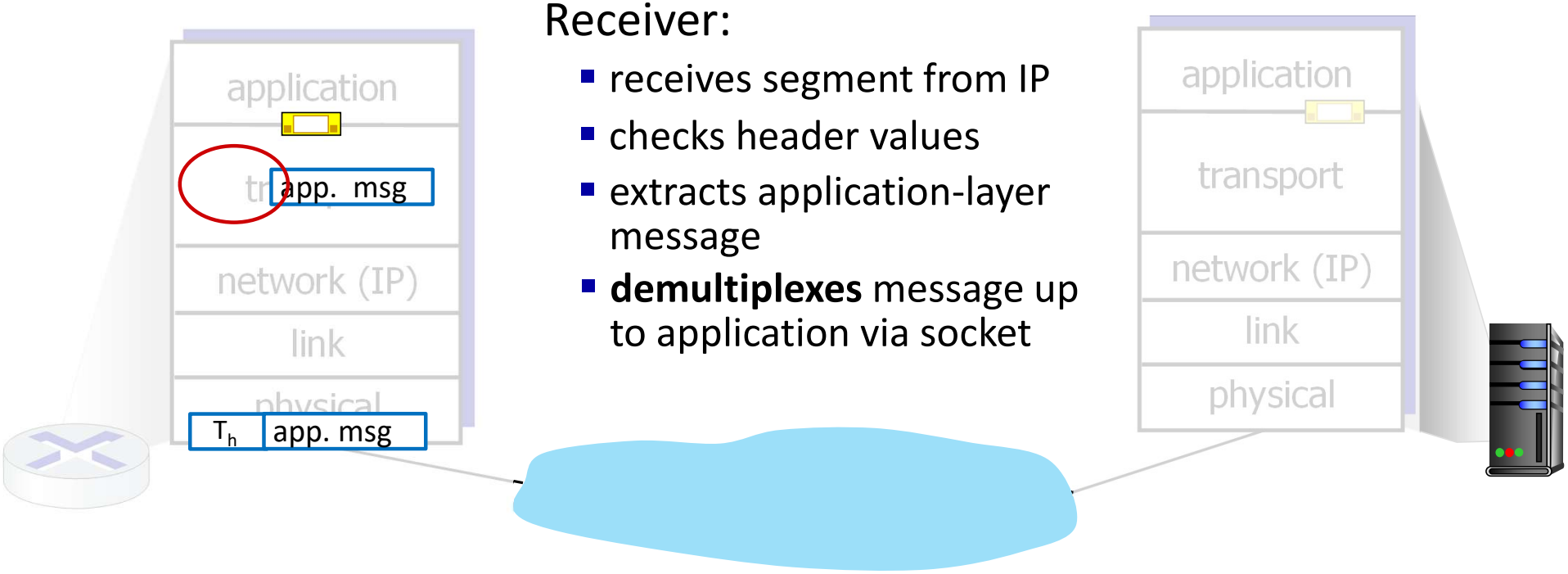
Transport Layer Actions

Sender:

- is passed an application-layer message
- determines segment header fields values
- creates segment
- passes segment to IP



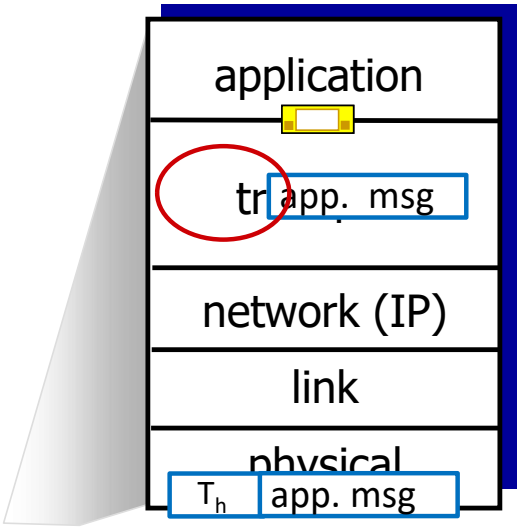
Transport Layer Actions



Receiver:

- receives segment from IP
- checks header values
- extracts application-layer message
- **demultiplexes** message up to application via socket

Transport Layer Multiplexing vs. Demultiplexing



- **Multiplexing** = Combining multiple messages/streams onto the same channel for transmission.

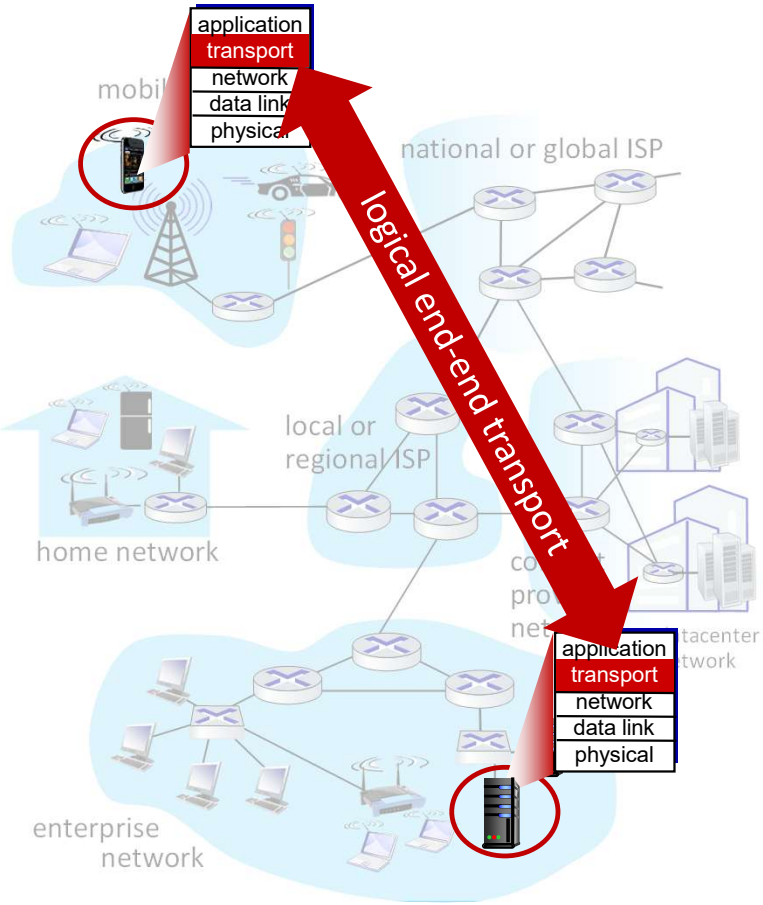
- **Demultiplexing** = The reverse process, where the receiver **separates incoming data** and delivers each part to the correct application or process

- **How Demultiplexing Works**

When a packet arrives at a host: The **IP layer** looks at the **destination IP address** to make sure the packet belongs to this host.
The packet is passed up to the **transport layer** (TCP or UDP).
The **transport layer header** contains a **destination port number**.
The OS uses this port number to **demultiplex** (deliver) the message to the correct application (like a web browser, email client, or video call).

Two principal Internet transport protocols

- **TCP:** Transmission Control Protocol
  - reliable, in-order delivery
  - congestion control
  - flow control
  - connection setup
- **UDP:** User Datagram Protocol
  - unreliable, unordered delivery
  - no-frills extension of “best-effort” IP
- services not available:
  - delay guarantees
  - bandwidth guarantees



Transport Layer – UDP

User Datagram Protocol (UDP):
It is an unreliable and connection-less protocol for data transmission.

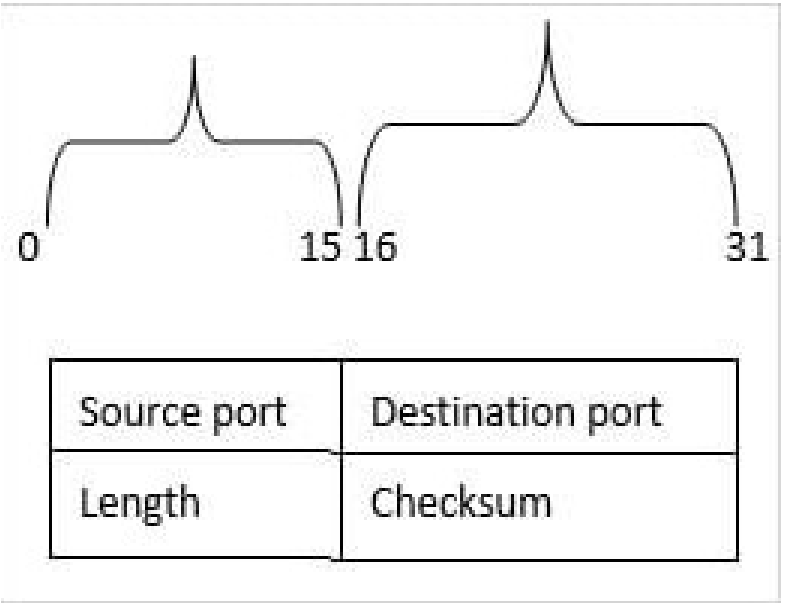
In this protocol, unlike TCP it doesn’t generate any ACK flag, hence the source host will not wait for a response from the destination end, and it will transmit the data without any delay and wait for ACK.

In a real-time scenario, UDP is used as the dropping of the data packets is chosen over waiting for packets for re-transmission.

Thus, it is most widely used in gaming, watching videos online, chatting, etc. where acknowledgment of data is not a concern.

In these scenarios, error checking and correction take place at the application layer.

Transport Layer – UDP Header



Source Port: It classifies the source end packet information which is 16 bits of size.

Destination port: It is also 16 bits in size and is used to classify the type of data service at the destination node.

Length: It indicates the overall size of the UDP datagram. The maximum size of the length field can be the overall size of the UDP header itself.

Checksum: It saves the checksum value evaluated by the source end before transmission. If it doesn't hold any value, then all of its bits are set to zero.

UDP: User Datagram Protocol

- "no frills," "bare bones" Internet transport protocol
- "best effort" service, UDP segments may be:
  - lost
  - delivered out-of-order to app
- *connectionless:*
  - no handshaking between UDP sender, receiver
  - each UDP segment handled independently of others

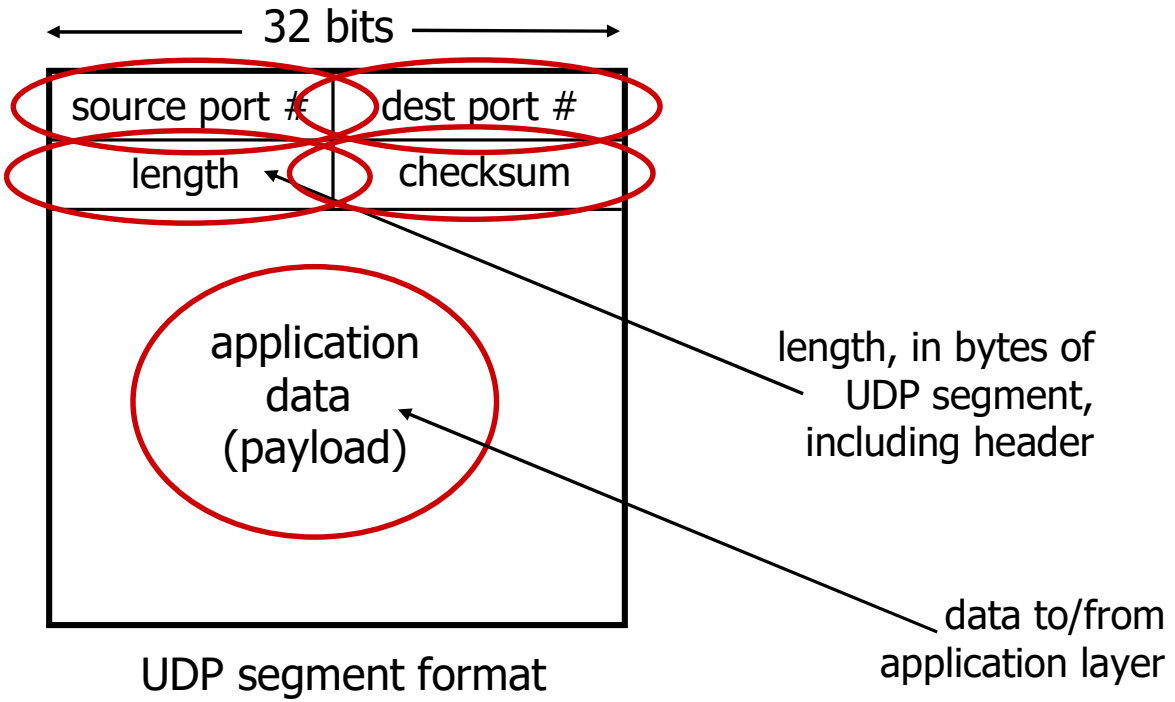
Why is there a UDP?

- no connection establishment (which can add RTT delay)
- simple: no connection state at sender, receiver
- small header size
- no congestion control
  - UDP can blast away as fast as desired!
  - can function in the face of congestion

UDP: User Datagram Protocol

- UDP use:
  - streaming multimedia apps (loss tolerant, rate sensitive)
  - DNS
  - SNMP
  - HTTP/3
- if reliable transfer needed over UDP (e.g., HTTP/3):
  - add needed reliability at application layer
  - add congestion control at application layer

UDP segment Header



UDP segment format

Internet checksum

Goal: detect errors (*i.e.,* flipped bits) in transmitted segment

sender:

- treat contents of UDP segment (including UDP header fields and IP addresses) as sequence of 16-bit integers
- checksum: addition (one's complement sum) of segment content
- checksum value put into UDP checksum field

receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - not equal - error detected
  - equal - no error detected. *But maybe errors nonetheless?* More later ….

UDP Applications

•It provisions datagram; thus, it is appropriate for IP tunneling and network file system.

•Simple in use, hence it is used in DHCP and trivial file transfer protocol.

•Being stateless makes it efficient for streaming media applications like IPTV.

•Also suitable for voice-over IP and real-time streaming programs.

•It backs the multicast; thus, it is appropriate for broadcast services such as Bluetooth and routing information protocol.

Transport Layer – UDP Applications

- Dynamic Host Configuration Protocol (DHCP) and Domain Name Services (DNS) are both essential network services,

- They both commonly use the UDP (User Datagram Protocol) because of:

  - low overhead and
  - fast transmission, which suits their use cases.

Transport Layer – UDP Application DHCP

🔌 **DHCP (Dynamic Host Configuration Protocol) and UDP**

✅ **What DHCP does:**

DHCP automatically assigns **IP addresses** and other network configuration parameters (like gateway, DNS server, subnet mask) to devices on a network.

🛠️ **Why it uses UDP:**

•DHCP communication happens **before the client has an IP address**.

•UDP is **connectionless**, so it works well when the client doesn't yet have a valid IP or full network stack setup.

•UDP is **faster and simpler** than TCP, which is ideal for bootstrapping (initializing) network communication.

🧭 **Ports used by DHCP**

Direction	Port
Client → Server	UDP Port 67
Server → Client	UDP Port 68

Transport Layer – UDP Application DHCP

🔁 **DHCP Message Exchange using UDP:**

DHCPDISCOVER (client → server): Broadcast request to find a DHCP server.

DHCPOFFER (server → client): Server offers an IP address.

DHCPREQUEST (client → server): Client requests the offered IP.

DHCPACK (server → client): Server acknowledges and leases the IP.

🔄 **Broadcasting with UDP:**

Clients **broadcast** messages to 255.255.255.255 (because they don’t know the server's IP yet).

UDP allows broadcasting, whereas TCP does not support it natively.

When does DNS use TCP instead of UDP?

✅ **What DNS does:**
DNS translates **domain names** (like example.com) into **IP addresses** that computers use to identify each other on the network.

🛠️ **Why it uses UDP:**
DNS queries are usually **short and quick**.
Using UDP avoids the overhead of establishing a TCP connection, making DNS lookups **fast and efficient**.

DNS can tolerate occasional loss — clients can simply **retry** if no response is received.

🧭 **Ports used by DNS**

Direction	Port
Client → Server	UDP Port 53
Server → Client	UDP Port 53

Transport Layer – TCP

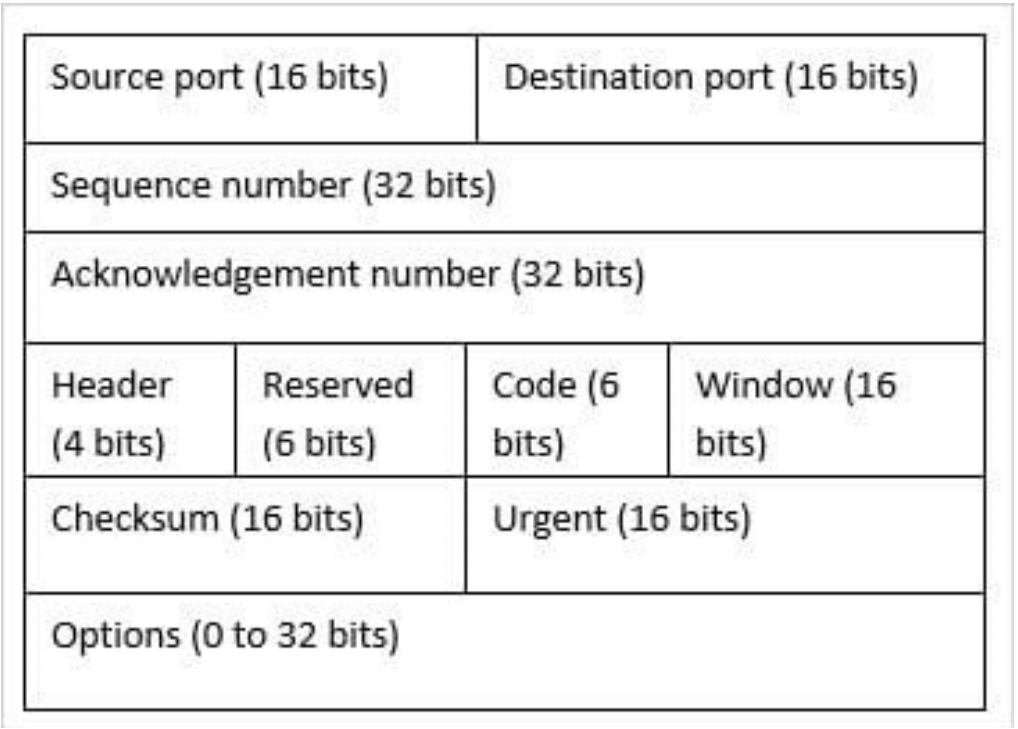
•Whenever the application layer needs to circulate the flow of huge traffic or data, it sends it to the transport layer in which the TCP performs all the end-to-end communication between networks.

•TCP initially set up a three-way handshake process between the source and destination and then it splits the data into small chunks known as segments, includes a header into every segment, and then forwards it to the Internet layer.

TCP: overview RFCs: 793,1122, 2018, 5681, 7323

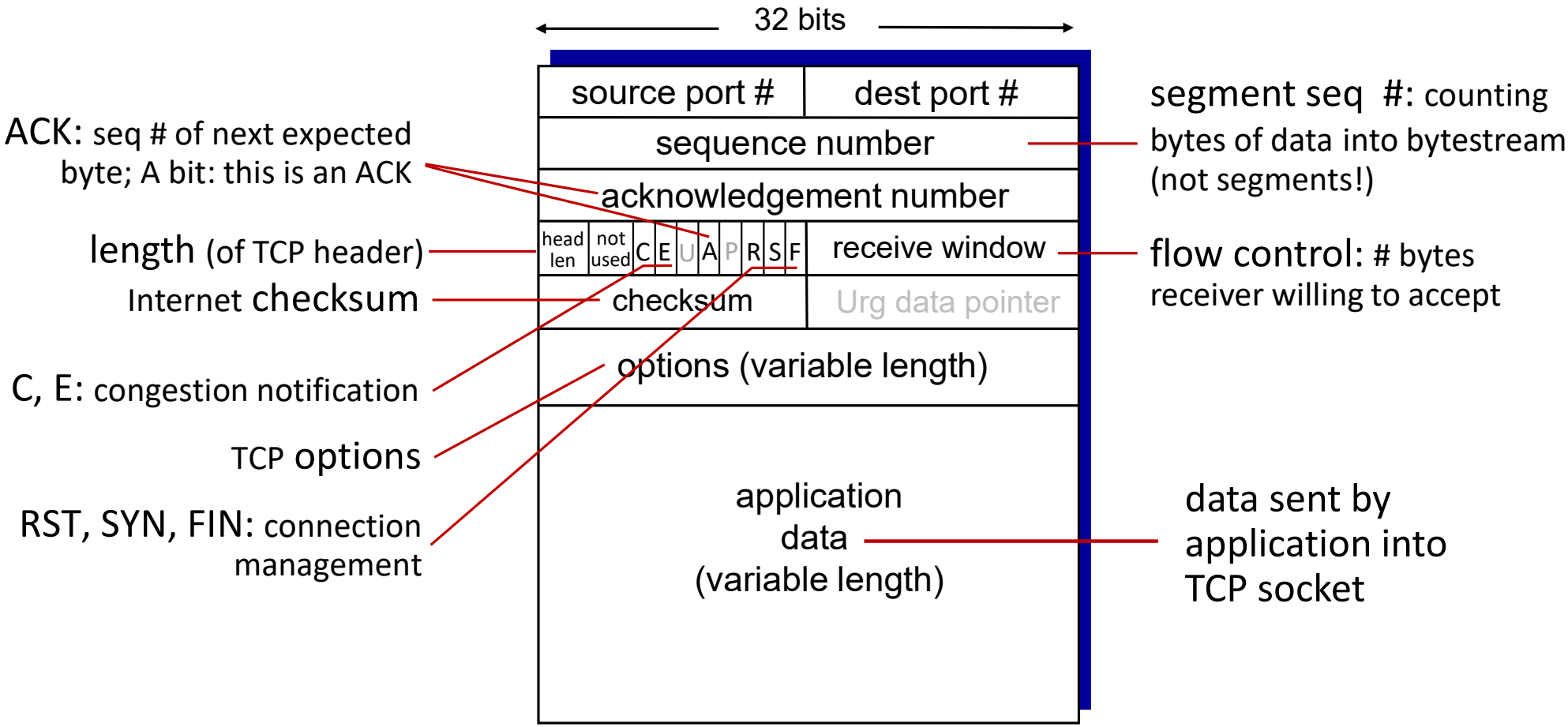
- point-to-point:
  - one sender, one receiver
- reliable, in-order *byte steam:*
  - no “message boundaries"
- full duplex data:
  - bi-directional data flow in same connection
  - MSS: maximum segment size
  - Typical MSS is 1460 bytes
- cumulative ACKs
- pipelining:
  - TCP congestion and flow control set window size
- connection-oriented:
  - handshaking (exchange of control messages) initializes sender, receiver state before data exchange
- flow controlled:
  - sender will not overwhelm receiver

Transport Layer – TCP Header



This figure shows the format of the TCP Header

TCP segment structure



Transport Layer – Data Segmentation

- **Data segmentations**:
  - This is one of the features of the TCP protocol. The application layer sends a large amount of data for transmission to the destination to the transport layer.
  - The transport layer limits the size of data to be sent in one go. This is done by splitting up the data into small segments.
  - To recognize the sequence of data segments, a sequence number is used in the TCP header, and that describes the byte number of the whole data segment.

Transport Layer – Three-Way Handshake

•**Three-Way Handshake:** It is the process deployed by TCP to establish a connection between the source and destination host in the network.
•It is used to perform reliable data transmission. It deploys **SYN** and **ACK** flags of code bits of the TCP header to perform the task.
• It provisions reliable communication by performing **positive acknowledgment** with re-transmission and is also known as **PAR.**
• The system using PAR will re-transmit the data segment until it receives the ACK.
•Whenever the receiver discards the data, the sender must re-transmit the data until it receives the positive ACK from the receiver.

Transport Layer – TCP

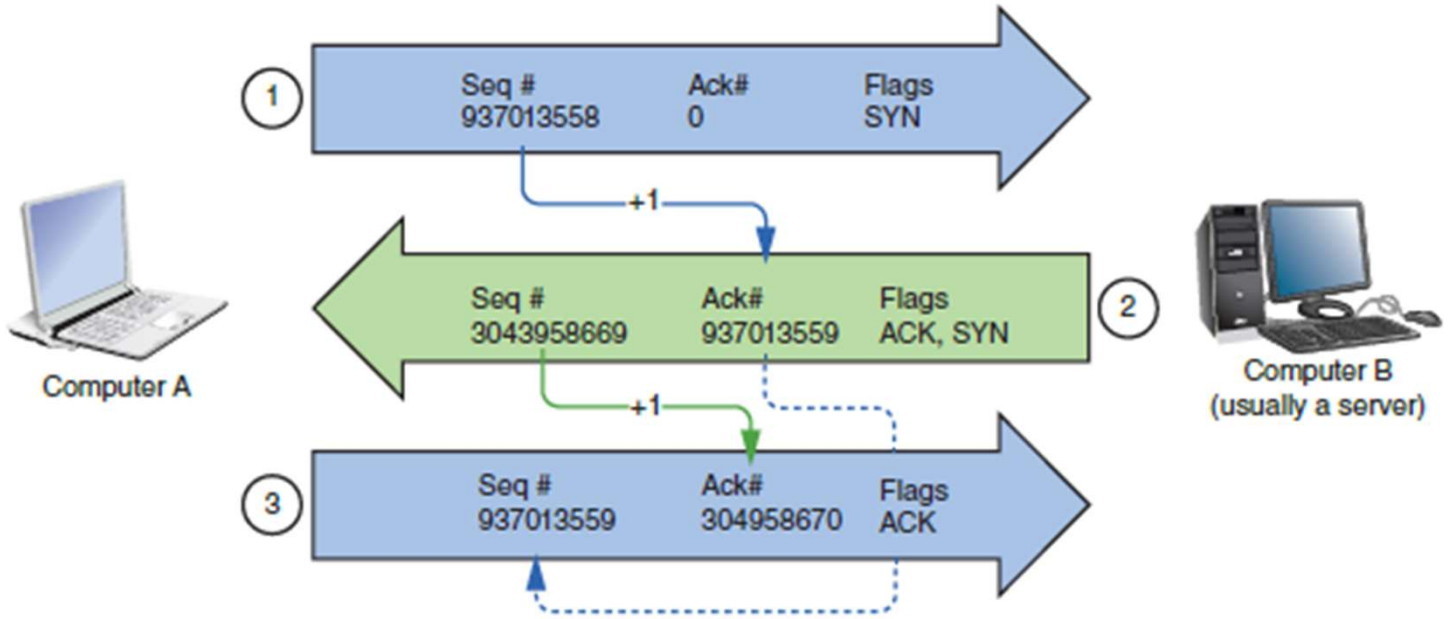


Figure 4-4 The three-way handshake process establishes a TCP session

TCP IP supports the client-server model of the communication system.

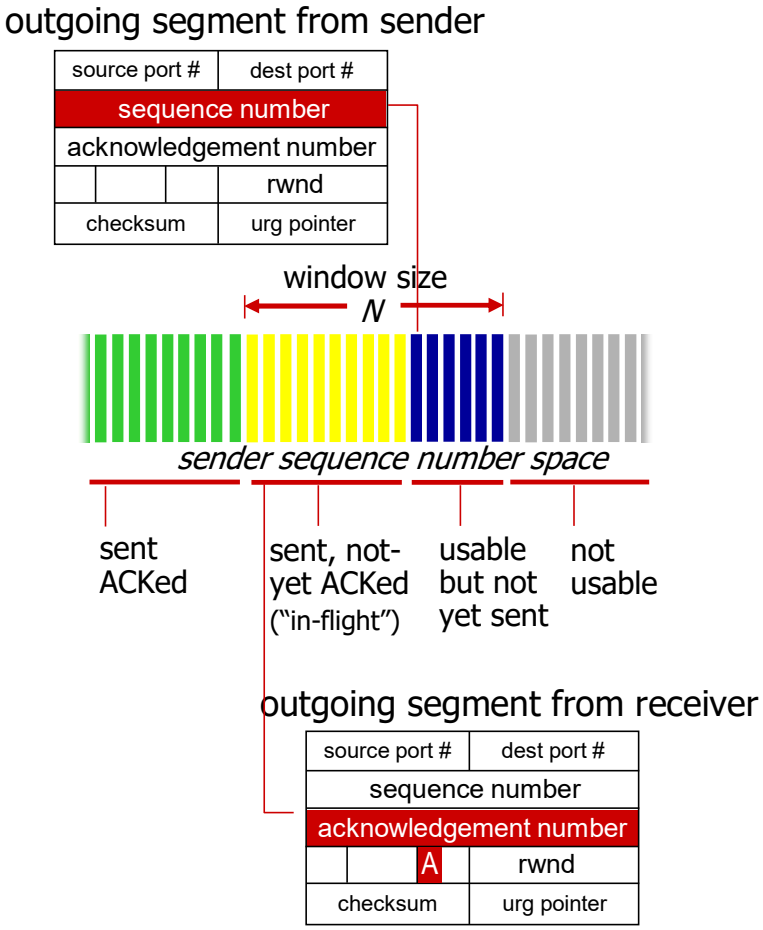
TCP sequence numbers, ACKs

Sequence numbers:

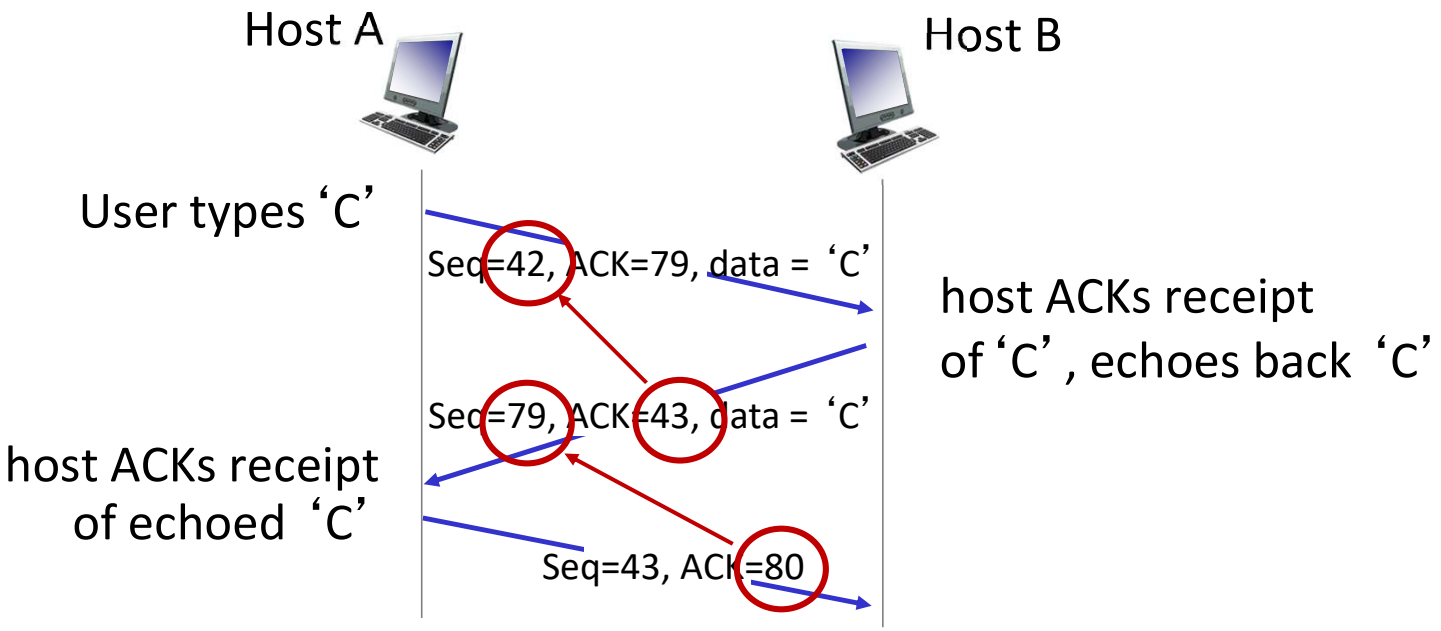
- byte stream "number" of first byte in segment's data

Acknowledgements:

- seq # of next byte expected from other side
- cumulative ACK



TCP sequence numbers, ACKs



simple telnet scenario

Transport Layer – Ordered Delivery

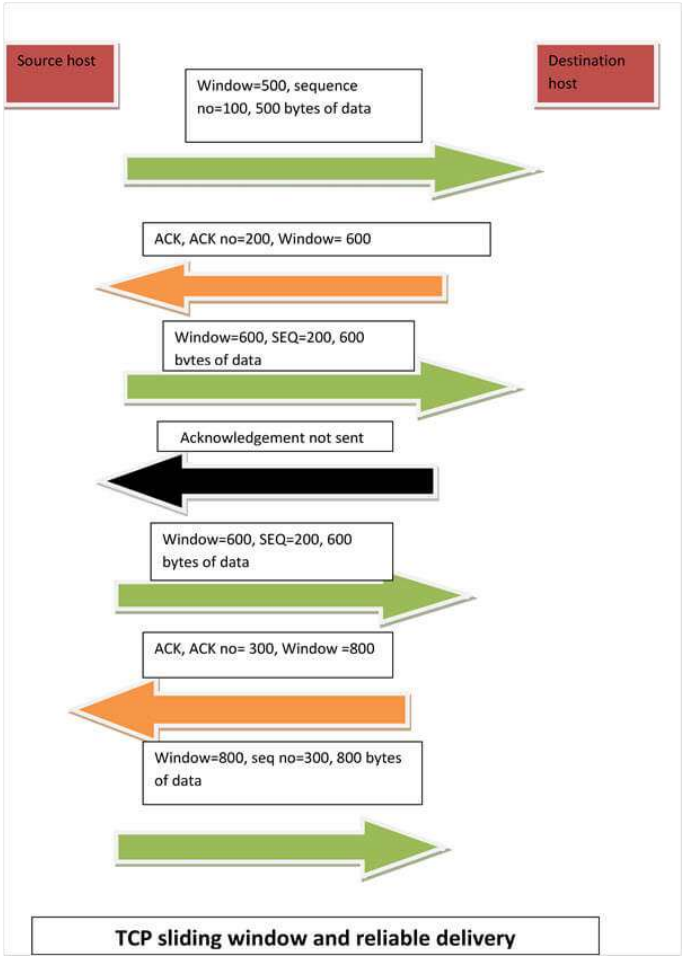
Ordered Delivery and Connection Termination:

•. **Ordered Delivery**:

- The TCP ensures the sequential delivery of data to the destination. It delivers the data in the order in which it receives it from the application layer for delivery to the destination host.
- Thus, for maintaining ordered delivery, it uses sequence numbers during transmission of data segments.

•**Connection Termination**:

- When the data transmission between source and destination is completed, the TCP will conclude the session by sending FIN and ACK flags and use a four-way handshake to close it.



Transport Layer Sliding Window

Error Detection:

Checksums: TCP adds a checksum to each segment to check for errors in the transmission. If the receiver detects any errors (e.g., corrupted data), it will discard the packet and not send an ACK. The sender will then retransmit the packet.

This ensures that data corruption is detected, and incorrect data is not acknowledged or processed

Retransmissions:

If the sender doesn't receive an ACK for a sent packet within a certain time limit (a timeout period), it will retransmit that packet.

Transport Layer – Flow Control

📌 **Flow Control – Receiver Protection**

🔍 **What it is:**

- Flow control prevents the **sender** from overwhelming the **receiver** with more data than it can process or store in its buffer.

🧠 **How it works in TCP:**

- **The receiver advertises** a **window size** (called the **receive window** or **rwnd**). This tells the sender how much data it can send before waiting for an acknowledgment.
- **The sender** must respect this limit and **pause or slow down** if the receiver can't keep up.

🛠 **Example:**

If the receiver's buffer is 16 KB and 10 KB is currently used, it will advertise a window of 6 KB. The sender can only send up to 6 KB more until space frees

Principles of congestion control

Congestion:

- informally: "too many sources sending too much data too fast for *network* to handle"
- manifestations:
  - long delays (queueing in router buffers)
  - packet loss (buffer overflow at routers)
- different from flow control!
- a top-10 problem!

congestion control:
too many senders,
sending too fast

flow control: one sender
too fast for one receiver

Congestion Control-Network Protection

🚦 **Congestion Control – Network Protection**

🔍 **What it is:**

Congestion control prevents **too much data** from being injected into the **network**, which can lead to packet loss, delays, and retransmissions due to **network congestion**.

🧠 **How it works in TCP:**

•TCP dynamically adjusts the **congestion window (cwnd)** based on perceived network conditions.

•It uses algorithms like:

- **Slow Start**: Start sending slowly, then ramp up.
- **Congestion Avoidance**: Increase cwnd cautiously to avoid congestion.
- **Fast Retransmit / Recovery**: Detect and recover from packet loss quickly.

🛠 **Example:**

If routers in the path get overloaded, packets get dropped. The sender interprets the lack of ACKs or duplicate ACKs as a sign of congestion and **reduces its sending rate**.

Congestion control vs. Flow Control

✅ **Key Differences at a Glance**

Feature	Flow Control	Congestion Control
Purpose	Prevent overwhelming the **receiver**	Prevent overwhelming the **network**
Concerned with	Receiver's capacity (buffer size, processing speed)	Network's capacity (routers, links)
Location of problem	End-to-end (sender ↔ receiver)	Network-wide (sender ↔ routers ↔ receiver)
Implemented at	Transport layer (mostly TCP)	Transport layer (TCP) but depends on network feedback
Example Mechanism	TCP Sliding Window	TCP Slow Start, Congestion Avoidance, Fast Retransmit

Networking

End of Review Session

420-421-VA Networking
Test 1
15% of Final Mark

Sept 30, 16:00-18:00

- **40 x Multiple Choice (1 point each)**
- **20 x Matching (1 point each)**
- **4 x Short Answer (5 points each)**
- **4 x Describe the commands (5 points each)**