COMPUTER COMMUNICATION NETWORKS

Prajeesha

Department of Electronics and Communication Engineering

COMPUTER COMMUNICATION NETWORKS

Concept of Layer Based Communication

Prajeesha
Department of Electronics and Communication Engineering

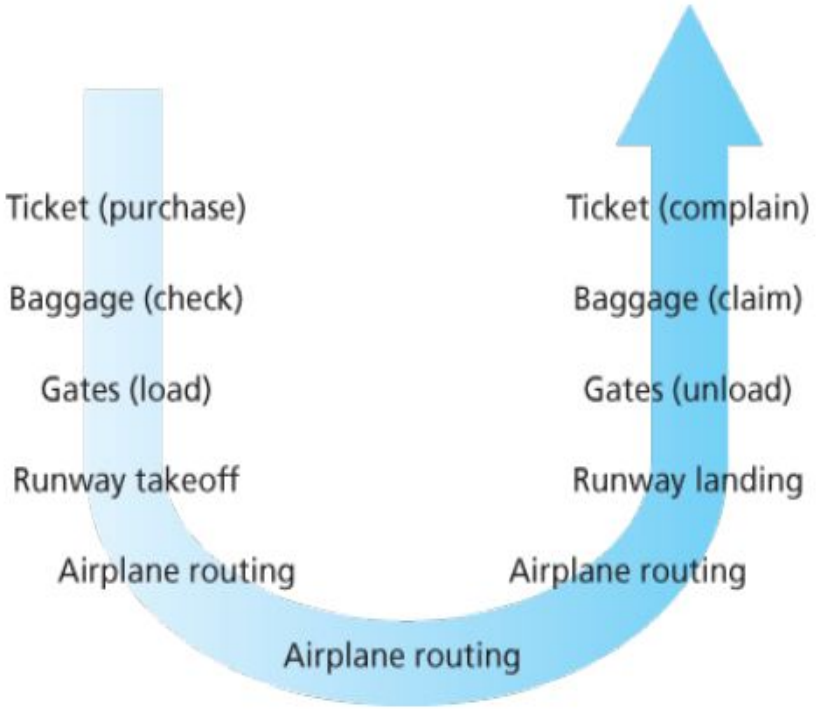
COMPUTER COMMUNICATION NETWORKS

Concept of Protocol Layers

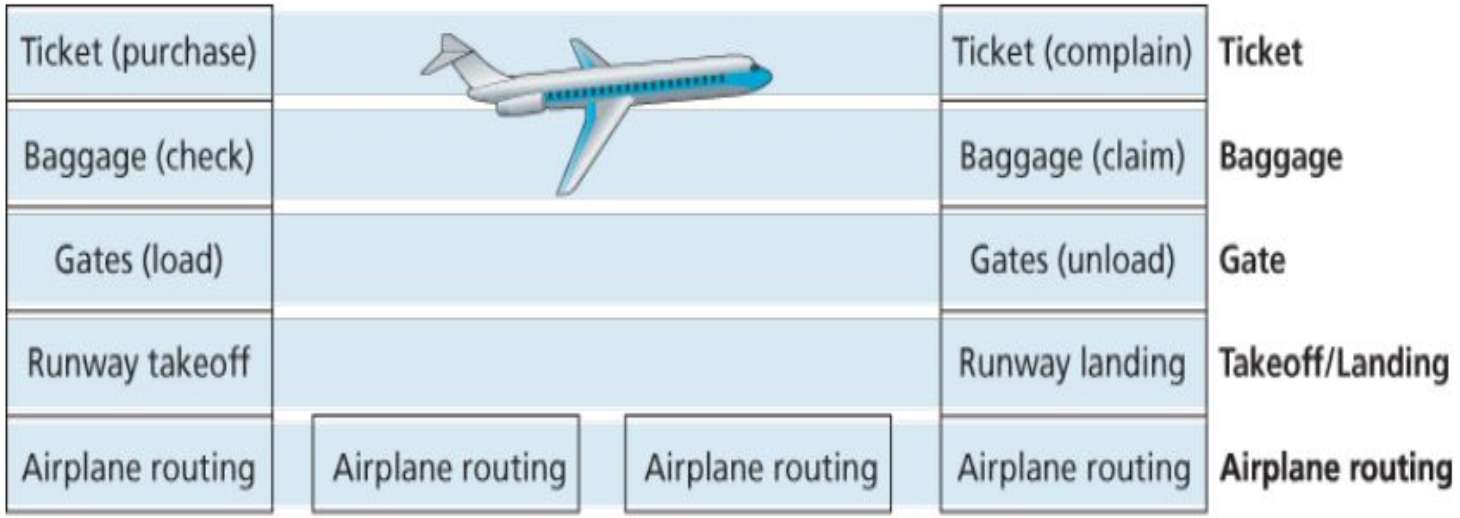
- Data exchange between two hosts over a communication network is a complex task
- The complex task is divided into smaller sub-tasks
  - Maintain simplicity for network devices
  - Put burden on the hosts
- The sub-tasks are completed sequentially
- The entire process can be visualized as layers arranged top to bottom, where
  - Each layer performs its own unique sub-task
  - On the sender side, each layer waits till the above layer finished
    - its sub-task
  - On the receiver side, each layer waits till the below layer finished its sub-task

COMPUTER COMMUNICATION NETWORKS

Concept of Protocol Layers



Taking an airplane trip: actions



Horizontal layering of airline functionality

Concept of Protocol Layers

- Communication between two hosts requires the **same layers** to be implemented **in both hosts.**
- Changes to one layer should not result in changes to other layers
- The **peer layers** (i.e., sub-task in sender and its counterpart in the receiver) communicate with one other using formatted blocks of data that obey a set of rules or conventions known as a **protocol.**
  A protocol layer can be implemented in **software**, in **hardware**, or in a **combination of the two**.
  Protocol layering has conceptual and structural **advantages**:
  - **Layering** provides a **structured way** to discuss system components.
  - **Modularity** makes it **easier to update** system components.

Concept of Protocol Layers

Potential Drawbacks of Layering

- One layer may duplicate lower-layer functionality.
- Functionality at one layer may need information (for example, a timestamp value) that is present only in another layer; this violates the goal of separation of layers.

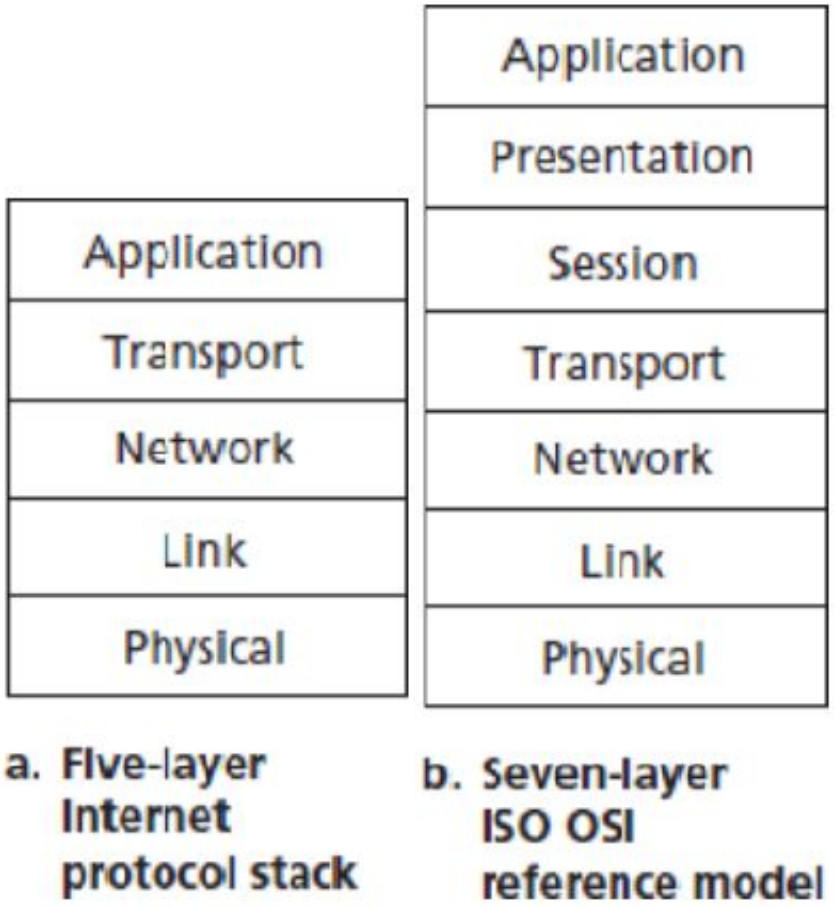
Basics requirements of a Protocol:

- **Syntax:** Concerns the format of the data blocks
- **Semantics:** Includes control information for coordination and error handling
- **Timing:** Includes speed matching and sequencing

COMPUTER COMMUNICATION NETWORKS

Concept of Protocol Layers

- Arranged vertically, the layers on the systems collectively constitute the **protocol architecture**
- Two types of protocol architecture were proposed
  - **TCP/IP**
  - **OSI**
- TCP/IP model or TCP/IP protocol suite
  - Resulted from protocol research under ARPANET
  - Consists of large collection of protocols for various layers called as **protocol stack.**
  - The Internet protocol stack consists of five layers: Application layer, Transport (host-to-host) layer, Network layer (IP layer), Link layer (network access layer), Physical layer



Concept of Protocol Layers

Application layer :

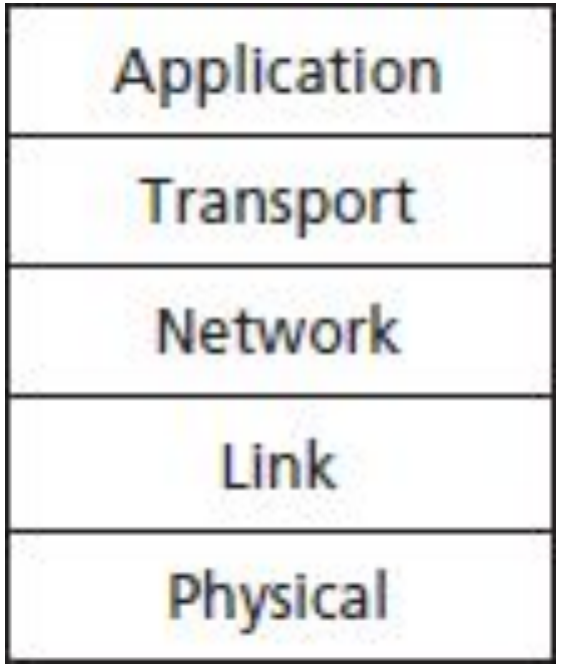
- **Applications (**aka **processes**) running on hosts generate/receive data.
- The data here is referred to as **message.**
- An **active process** on one host initiates communication with an active process on another host.
- Messages are formatted according to the application layer protocol defined by the application developer.
- Messages can be big in size (e.g., audio; video).
- Applications can have QoS requirements.

TCP/IP model

Concept of Protocol Layer

Transport layer:

- Responsible for providing **QoS** for a pair of communicating processes
- Performs **multiplexing at the sender**
- Performs **demultiplexing at the receiver**
- **Maps** each message to a corresponding process inside a host
- This mapping is accomplished by unique identifiers called **ports or sockets**
  - Appends a new header to each message
  - Message plus header is called **segment**

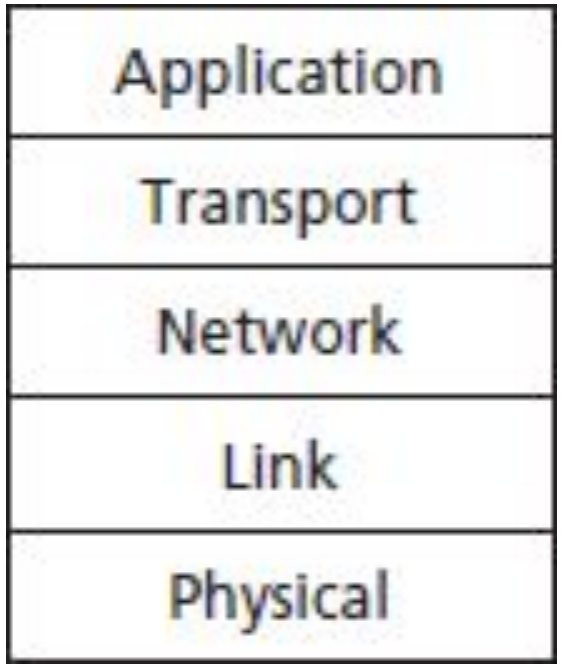


TCP/IP model

Concept of Protocol Layers

Network layer:

- Moves packets hop-by-hop in search of the destination
- E.g., router to router
- Search for the destination is donevia unique identifiers called **IP addresses**
- Network level information is gathered to facilitate discovery of paths to destinations
- Appends a new header to each segment
- Segment plus header is called **datagram**



TCP/IP model

Concept of Protocol Layers

Link layer:

- Ensures **reliable packet flow** across each link on the path between a pair of source and destination independently
- Using respective link layer protocol
- Reliable flow of packets across a link requires identifiers for that link

This is accomplished by **MAC address**

- Provides synchronization between sender and receiver of each link
- Appends a new header to the datagram
- Datagram plus header is called **frame**
- **Checks for errors** in frame

TCP/IP model

Concept of Protocol Layers

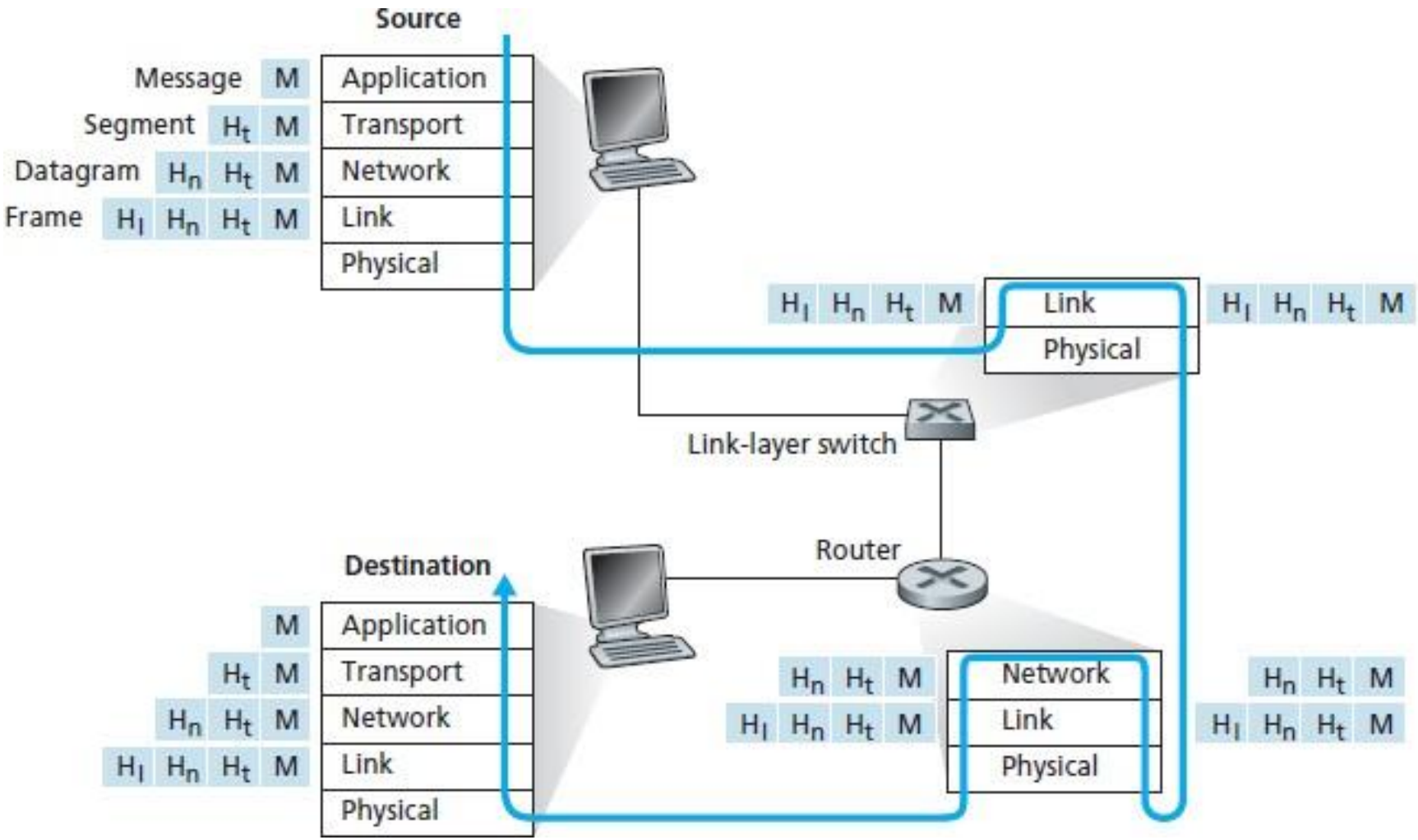
Physical layer:

- Deals with the characteristics of the physical medium of a link
- Converts frames into signals (e.g., radio signal, optical signal)
- Performs signal modulation and demodulation
  - E.g.: Modem and Ethernet card, wireless adapter
- Manages transmit power

TCP/IP model

COMPUTER COMMUNICATION NETWORKS

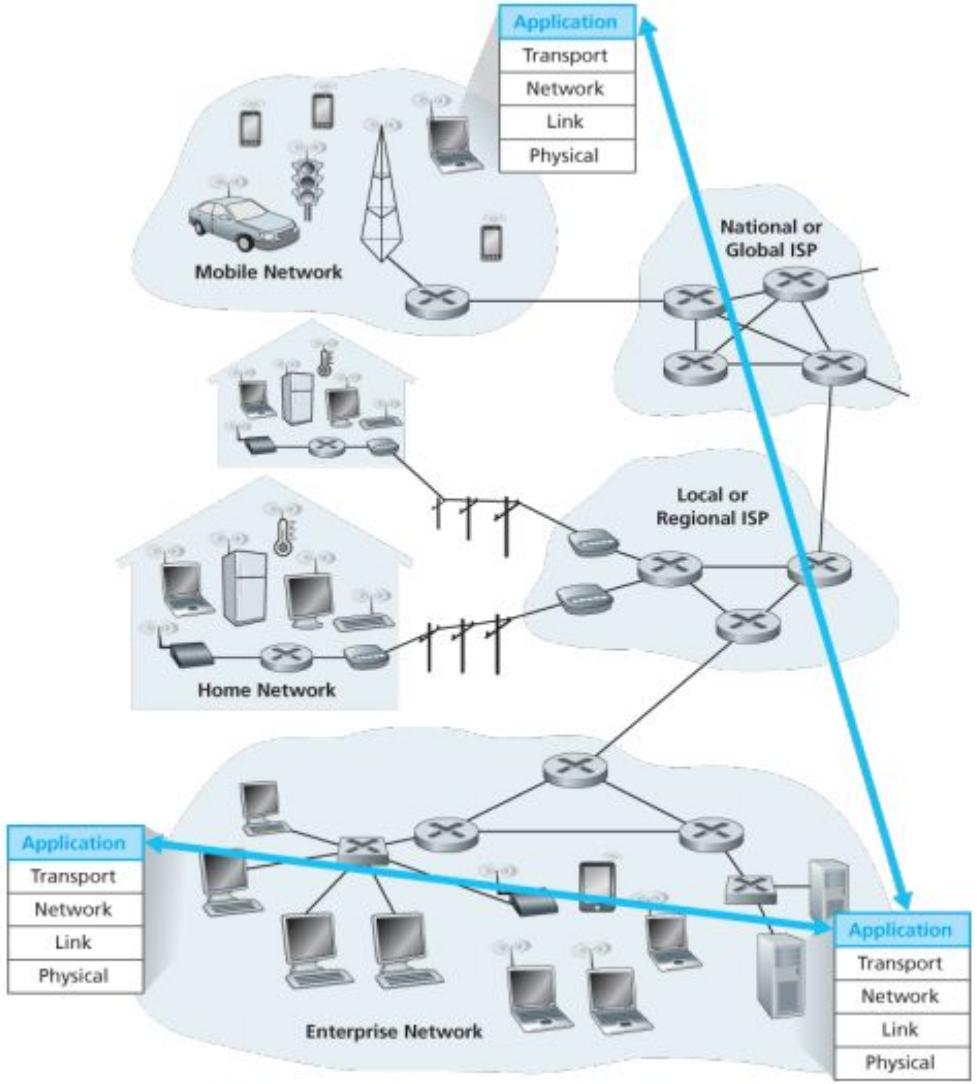
Encapsulation and Decapsulation



COMPUTER COMMUNICATION NETWORKS

Process Communication

- A **process** is a program that is running within an end system.
- When processes are running on the same end system, they can communicate with each other with **inter-process communication**, using rules that are governed by the end system's operating system.
- Processes on two different end systems communicate with each other by **exchanging messages** across the computer network. A **sending process creates and sends** messages into the network; a **receiving process receives** these messages and possibly responds by sending messages back.



Communication for a network application takes place between end systems at the application layer

Process communication

- A **process** sends messages into, and receives messages from, the network through a software interface called a **socket.**
- A **socket** is the interface between the application layer and the transport layer within a host. It is also referred to as the **Application Programming Interface (API)** between the application and the network layer.
- The application developer has control of everything on the application-layer side of the socket but has little control of the transport-layer side of the socket.
- The only control that the application developer has on the transport-layer side is

  (1) the choice of transport protocol and

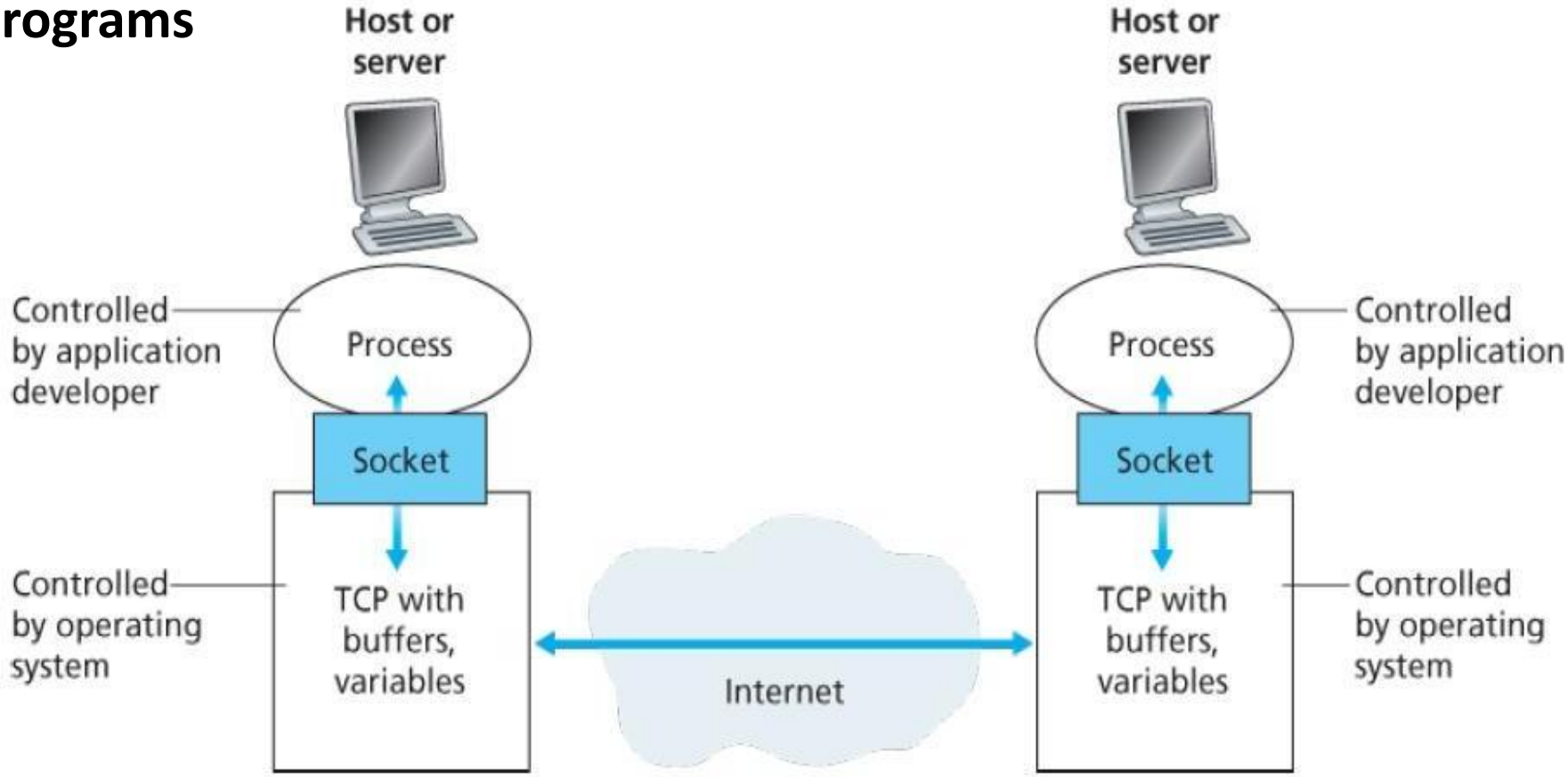
  (2) perhaps the ability to fix a few transport-layer parameters such as maximum buffer and maximum segment sizes

Process communication

- Application developers provide **unique IDs** for **identifying processes** running in a host (e.g., HTTP server process 80; DNS server process 53)
- Application developers build a **pair of processes** which usually run on different hosts.
  - Example: A browser program running on a laptop and a server program running on a web server
- When one process initiates an action it is transmitted in the form of a special message and the corresponding process which receives this message takes suitable action and returns another special message back to the initiator.
  - Example: A browser program sends a request for a webpage when a URL is entered by a user. The web server receiving the request may return the requested object over the internet to be displayed on the browser

Process communication

- These special messages are formatted as defined by the application developer, stored in buffers, transmitted/received using appropriate process identifiers. All these actions can be coded and referred to as **socket programs**



Addressing Processes

- For a process running on one host to send packets to a process running on another host, the receiving process needs to have an address.

- To identify the receiving process, two pieces of information need to be specified:
  - the address of the host and
  - an identifier that specifies the receiving process in the destination host

- In the Internet, the host is identified by its IP address.

- The sending process identifies the receiving process (the receiving socket) running in the host using a destination port number.

COMPUTER COMMUNICATION NETWORKS

Client - Server Model

Server process:

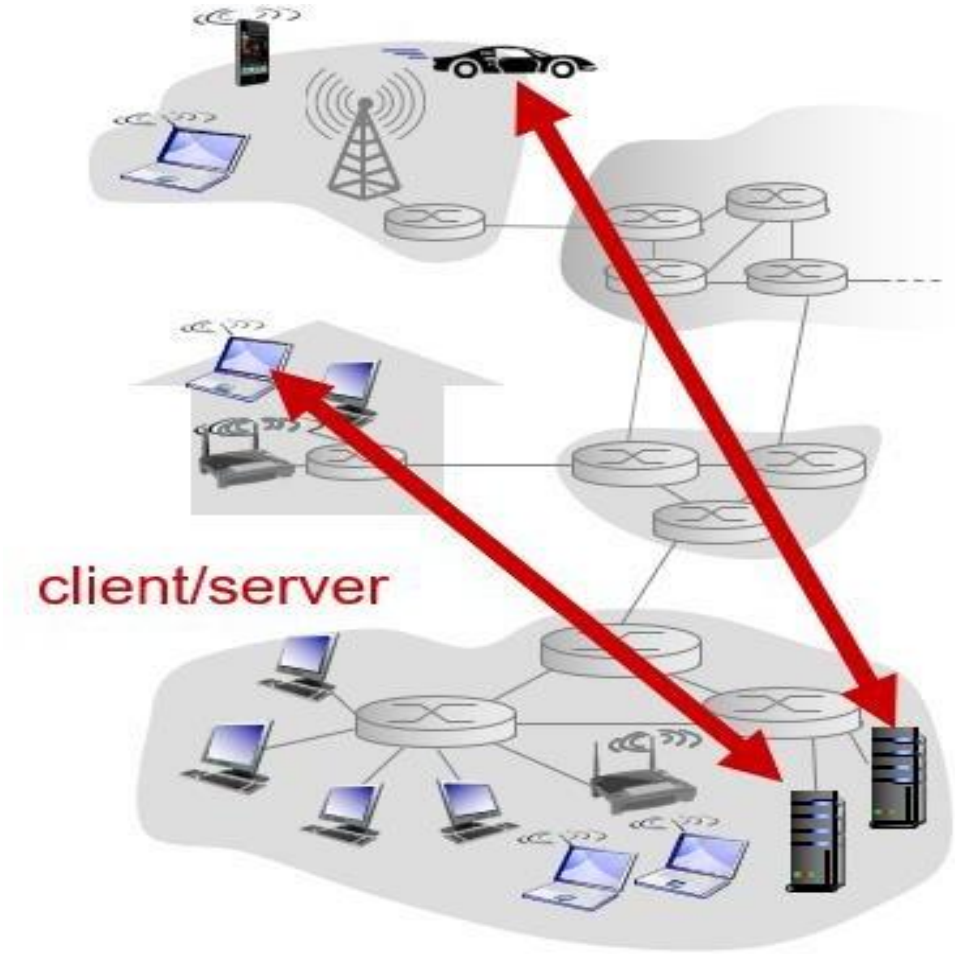
- Always running
- Uniquely identified by socket and IP address

Client process:

- Activated on demand
- Many client process may communicate with one server process
- Each client process randomly chooses a socket for its ID

Client-server model examples

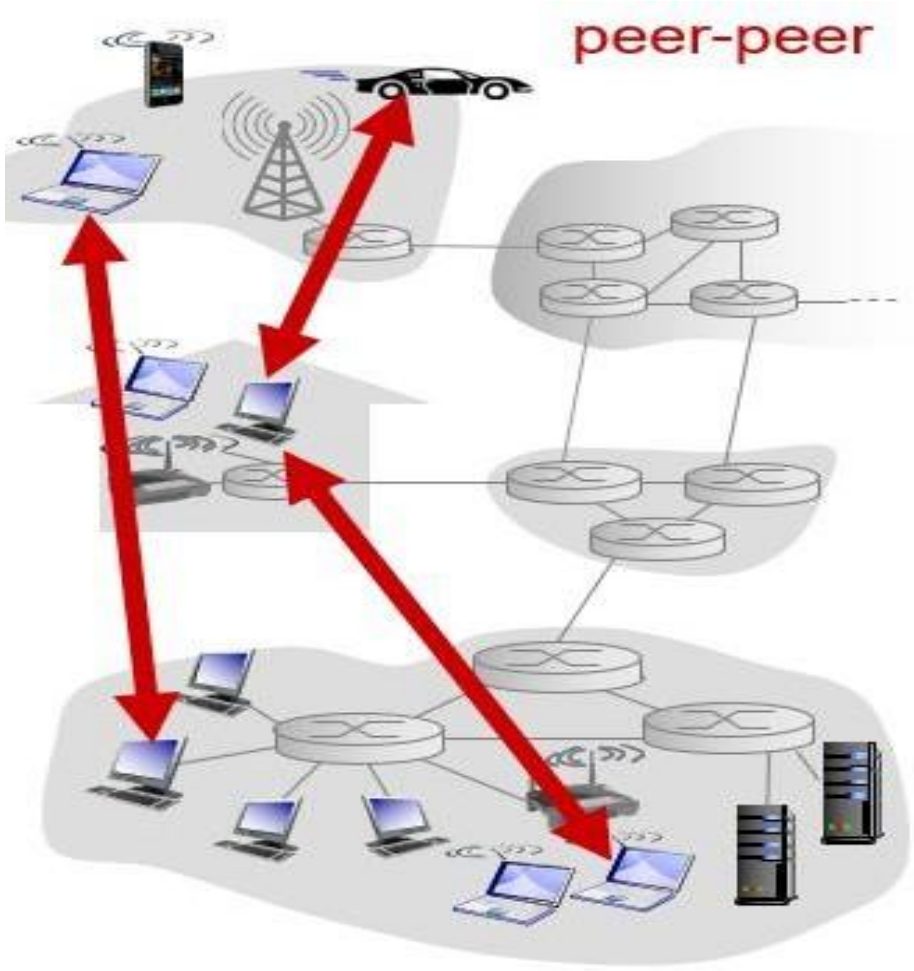
- Web, Email, File Transfer



COMPUTER COMMUNICATION NETWORKS

Peer to Peer Model

- Peer processes are activated on demand
- Peers discover sockets and IP addresses of one another
- Involve complex process management to ensure good performance
- Applications based on peer-to-peer model exhibit self scalability and adaptability
- Best suited for file distribution
  - Examples: BitTorrent, Skype



THANK YOU

Prajeesha

Department of Electronics and Communication Engineering