CS3.301 Operating Systems and Networks

Networking - Sockets and Networking Layers

Karthik Vaidhyanathan

https://karthikvaidhyanathan.com

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

SERC
Software Engineering Research Centre

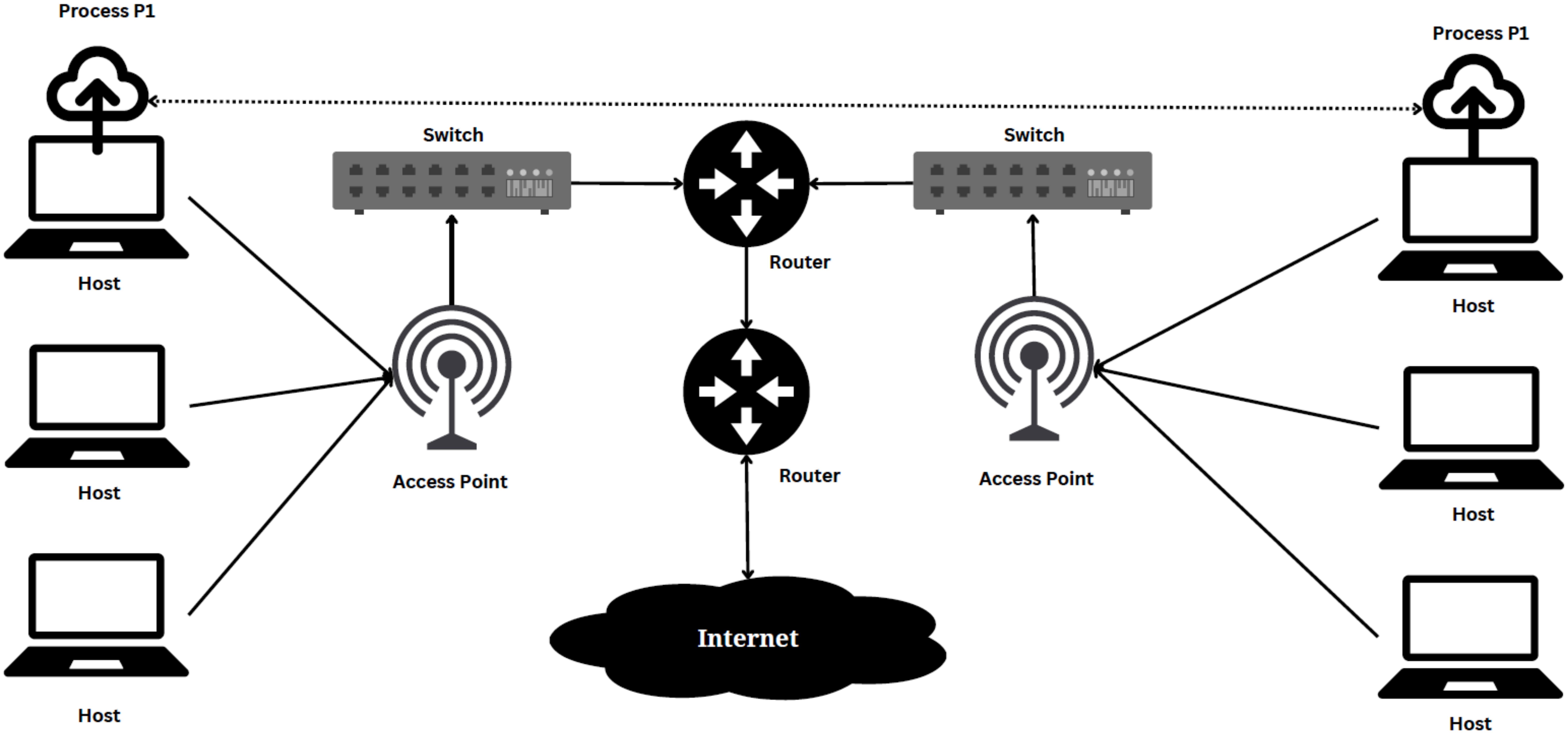
Acknowledgement

The materials used in this presentation have been gathered/adapted/generate from various sources as well as based on my own experiences and knowledge -- Karthik Vaidhyanathan

Sources:

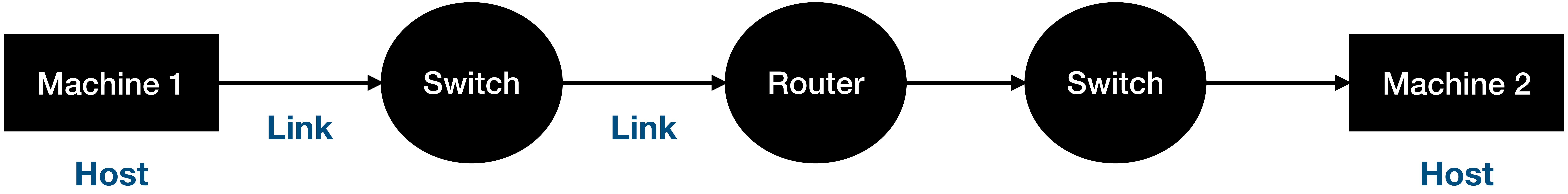
- Computer Networks, 6e by Tanebaum, Teamster and Wetherall
- Computer Networks: A Top Down Approach by Kurose and Ross
- Computer Networking essentials, Youtube Channel
- Other online sources which are duly cited

The Bigger Picture



What happens internally when one process wants to share/ receive data from another host?

Components of a network



Process Communicating over network

- Process A (eg: Whatsapp) is executing in Host 1
- Process B (whatsapp) executing in Host 2
- Process A wants to communicate with Process B
  - Some system call is made to access the network
    - Leads to an interrupt
    - Process A should know where process B is
    - Hardware support is needed (some network device with drivers)

Network Types

Multiple types of networks exist

- **Personal Area Network - PAN**
    - Devices communicate over a very short range
    - Eg: Bluetooth (master slave paradigm)
- **Local Area Network - LAN**
    - Private network operating within and nearby a single building (home, office, factory, etc.)
    - Wireless LANs runs speed from 11 Mbps to 7 Gbps (1 Mbps is 1,000,000 bits per sec) - Power in 10 (not in terms of 2)
    - Wired LANs operate at speed ranging from 100 Mbps to 40 Gbps (low latency)
    - Eg: connecting personal computers and consumer electronics (eg: printer)

Network Types

We have come a long way!

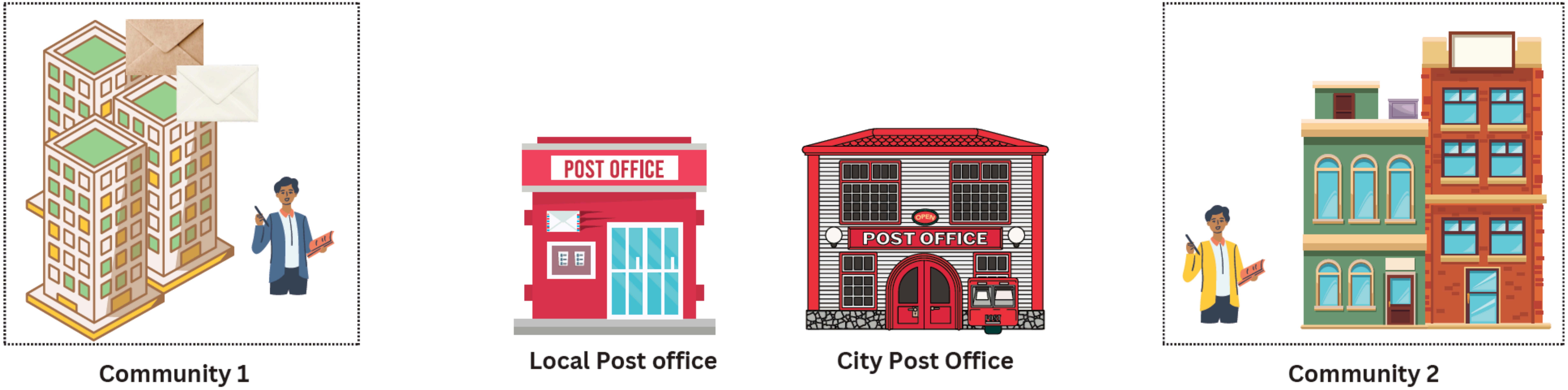
- One large Physical LAN can be divided into smaller logical LANs (Virtual LANs)
- Earlier days it was about broadcasting on a single line!
  - At most one computer could transmit successfully at a time
  - Use static allocation techniques
    - Every machine gets some time to transmit or receive
    - Round robin was used for scheduling
  - Packet collision used to happen (Wait for some random time and try again)

Network Types

- **Metropolitan Area Network - MAN**
  - Covers a city (City wide networks)
  - Think of cable TV networks (Earlier Antenna on top of the house and from the top of the hill the communication used to happen)
  - Idea of cable TV was used for providing two services (Internet and TV)
- **Wide Area Network - WAN**
  - Spans a large geographical area (often country, continent, etc.)
  - Eg: Internet is a large WAN (Dedicated WANs also exist)
    - There can be WAN that connects offices of organisation in different locations
  - Higher latency and lower transmission speeds. Cost for dedicated ones can be high

An Illustrative Scenario

Sending Letter/Courier



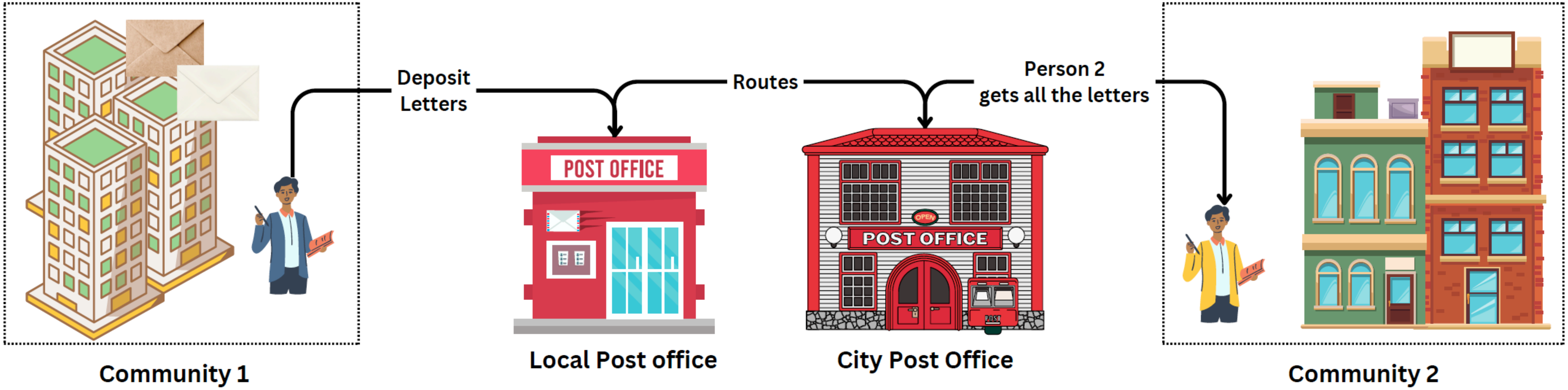
Two large communities in different parts of the city, two people in each community to collect and deliver the letters/couriers

An Illustrative Scenario

People in community send letters/couriers

One person is collecting all the letters/couriers

Person delivers the letter to each person



Community 1

Deposit Letters

Local Post office

Routes

City Post Office

Person 2 gets all the letters

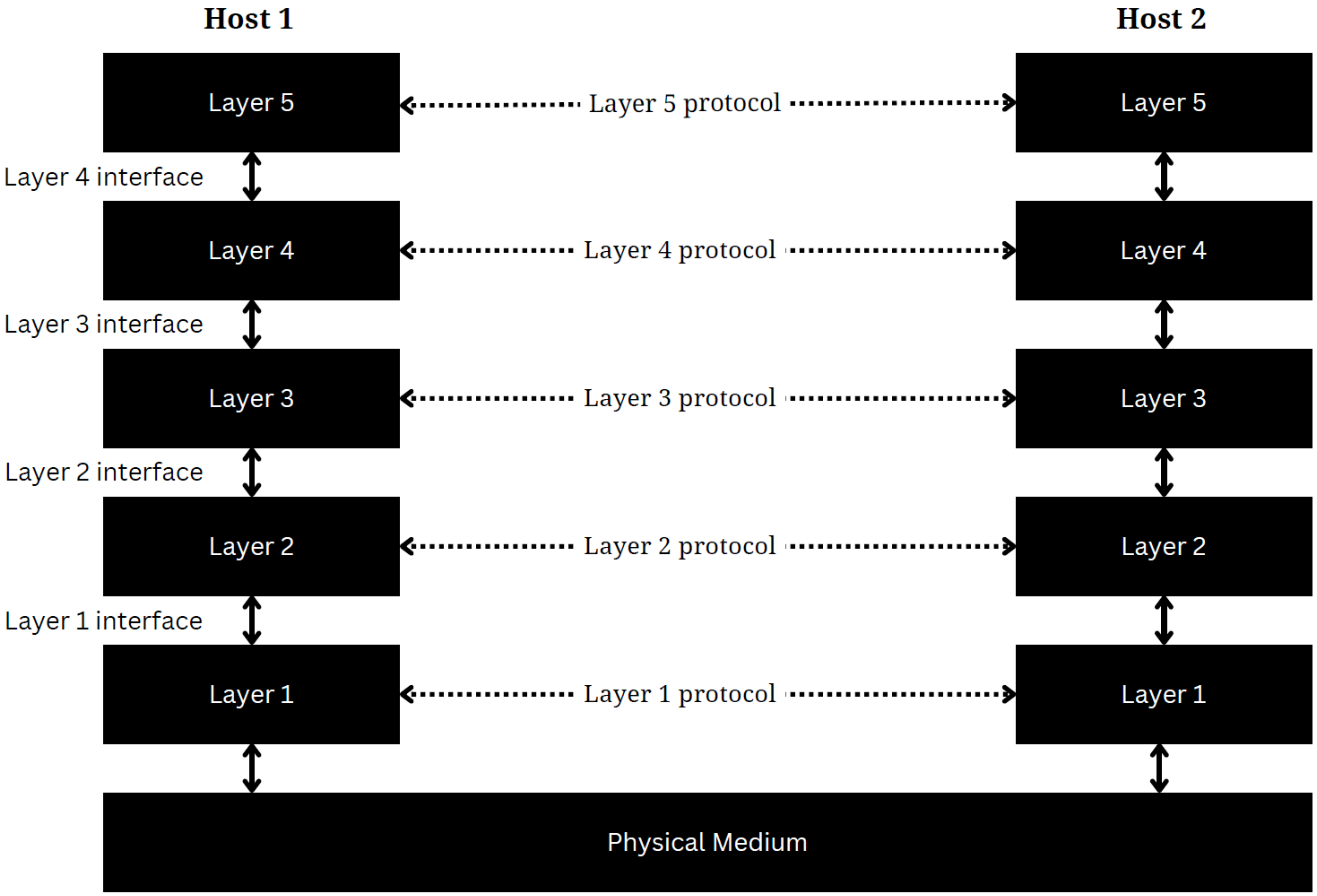
Community 2

Key points: Door number, building number, Post office has an identifier, etc.

Lets take this to Process communication

- Process A (eg: Whatsapp) is executing in Host 1
    - Process B (Whatsapp) is executing in Host 2
- Host 1 will have an address, same is the case with host 2
- How to ensure the data reaches from Host 1 to Host 2?
    - What all needs to be considered?
    - Remember: There will be multiple processes that are executing in a host

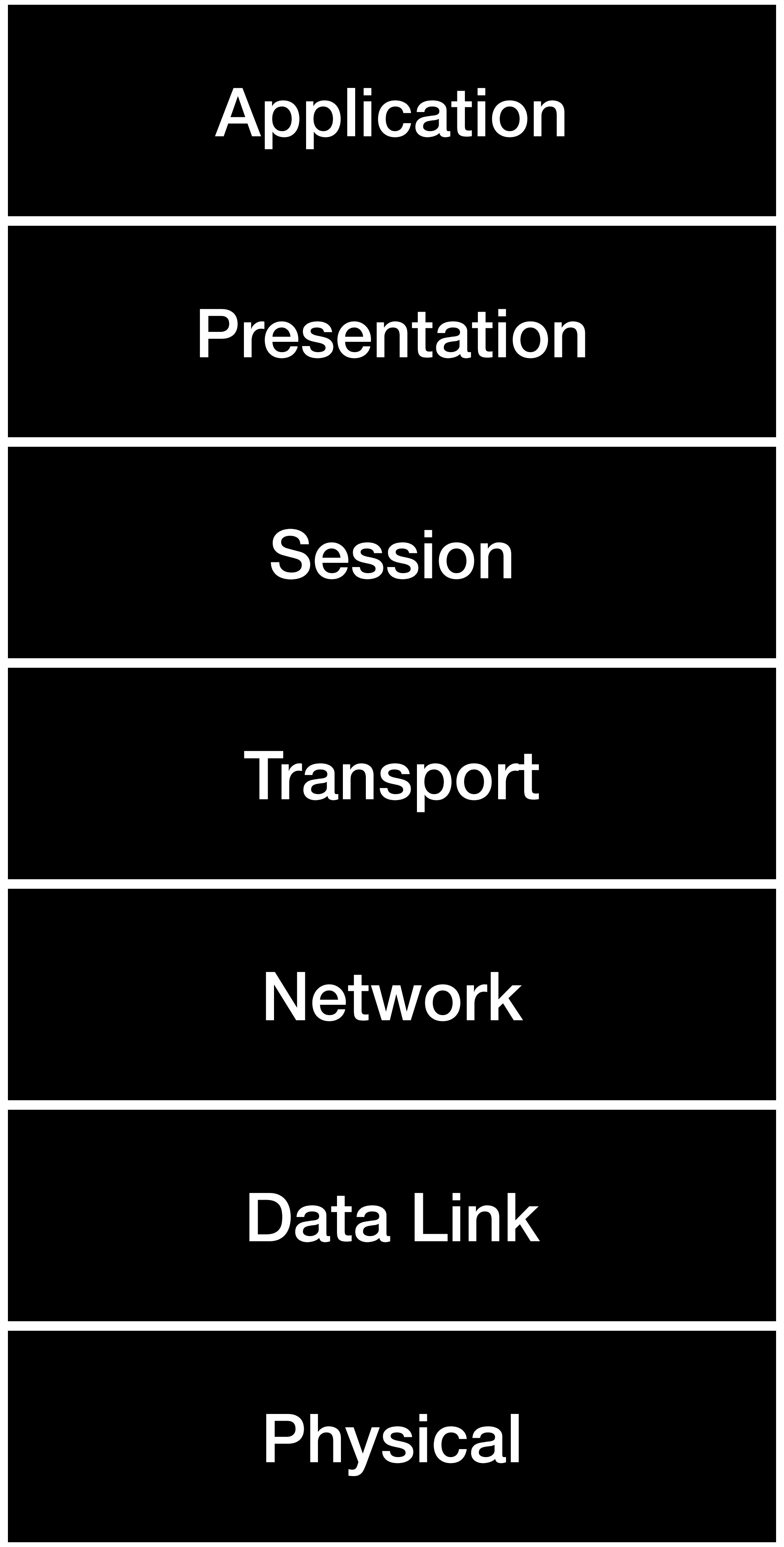
Networking Layers



Networking Layers

- Reduce design complexity, network organised a stack of layers or levels
- The layering provides abstraction in and to other layers
- Communication (in some sense) happens between the corresponding layers
    - Layer n one machine communicates with layer n of another using a **protocol**
    - Protocol is just like agreement between communicating practices
- Set of layers along with protocols => **Network Architecture**
- Between each pair of adjacent layer their is an **interface**
    - Defines the primitive operations and services the lower layer makes available

The OSI Model



Application

Presentation

Session

Transport

Network

Data Link

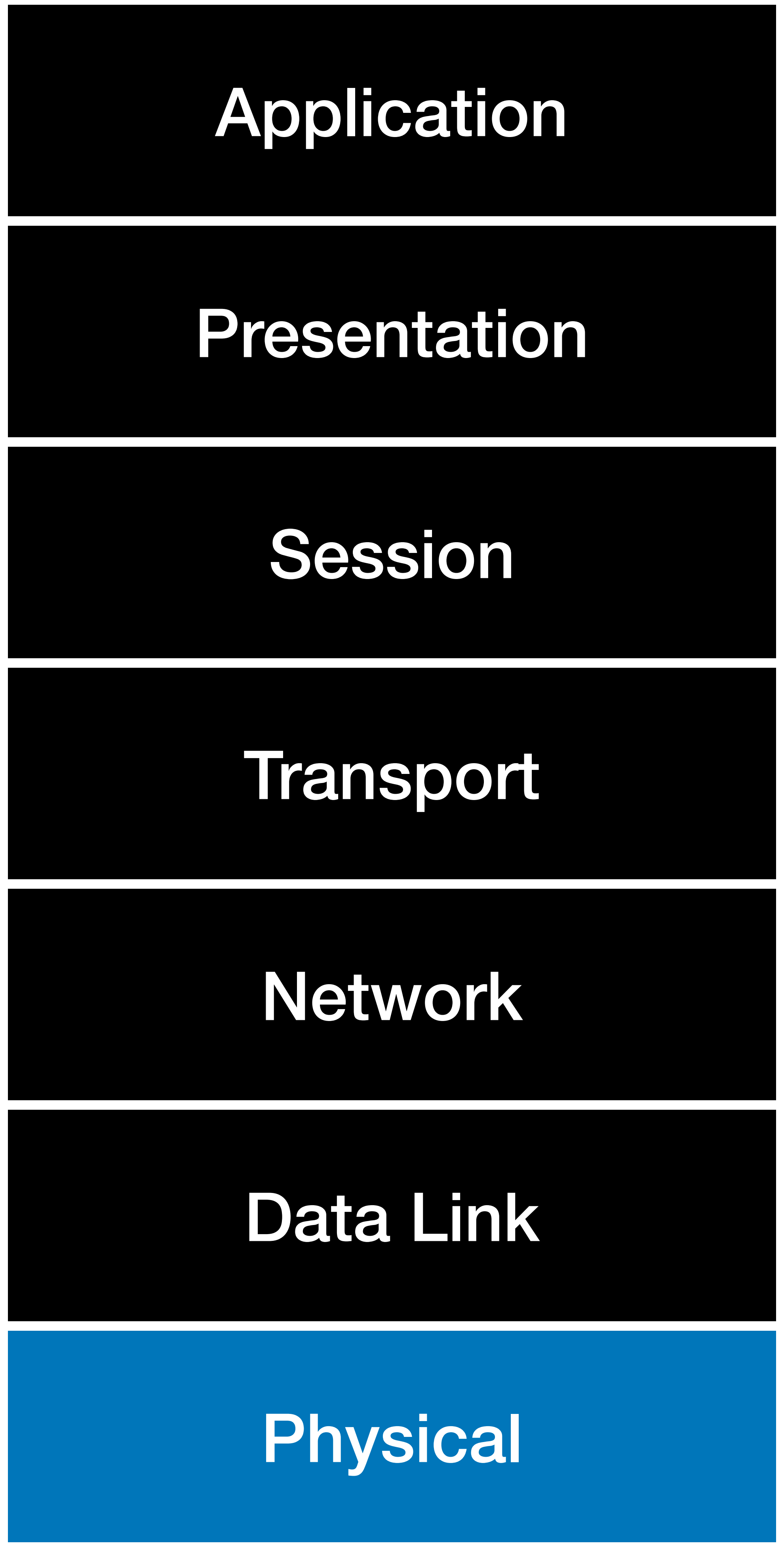
Physical

- Open System Interconnection (OSI)
- A Conceptual framework used to understand how communication works through different layers
- Divides the network communication process into seven layers
- Developed to facilitate interoperability between different technologies
- Each layer has a specific function. If they all do what they are supposed to do => sharing of data

Physical Layer (L1)

Ultimately everything is 0’s and 1’s

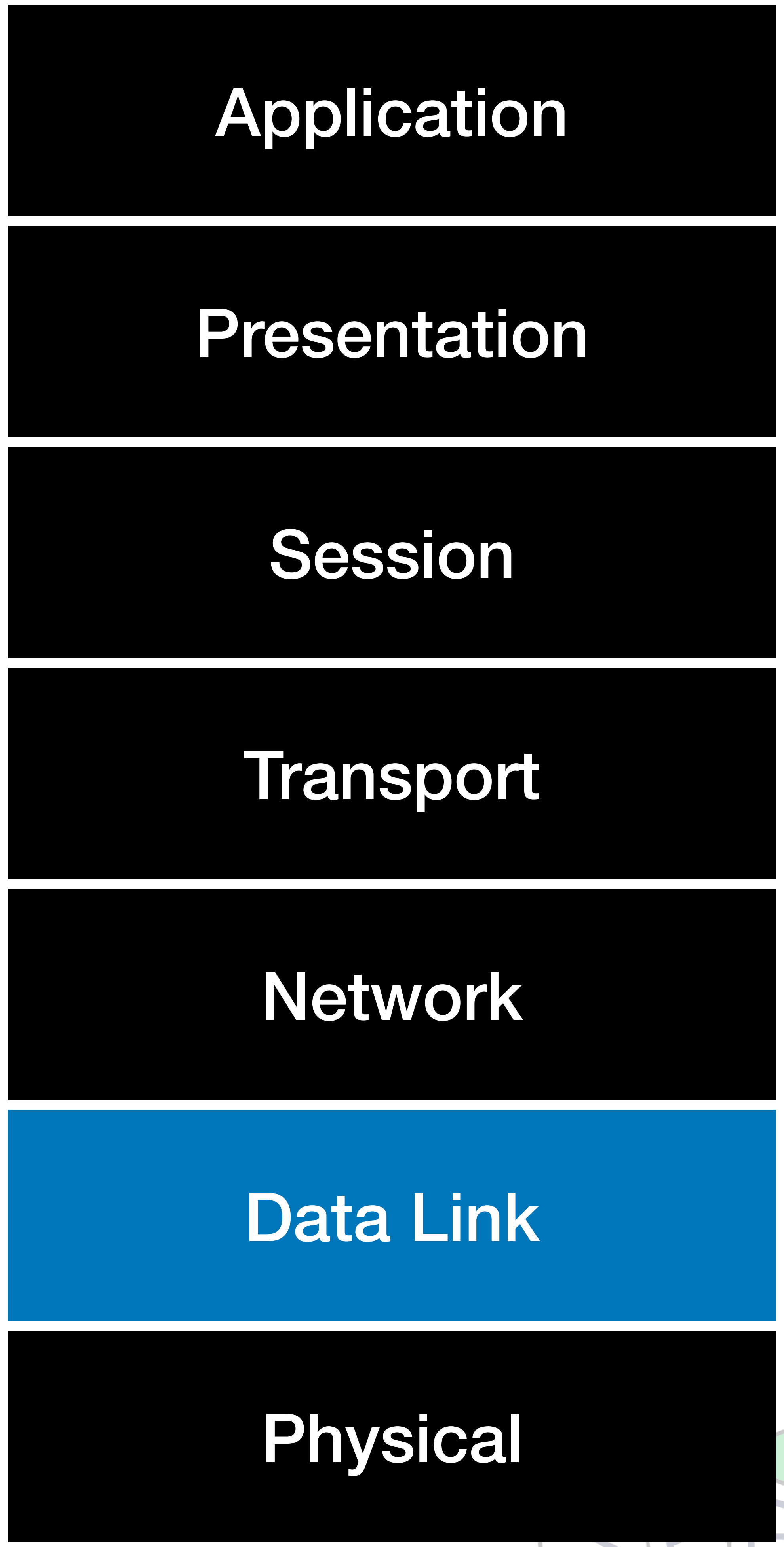
- Data is in the form of bits - 0s and 1s
- Something has to transport the bits from one machine to another - Physical layer
- Concerned with transmission of raw bits over physical medium, like a cable
- L1 technologies: Ethernet cables, Optical fiber, Coaxial cable, etc.
    - Even WiFi is L1 technology, hub, repeater, etc.



Data Link Layer (L2)

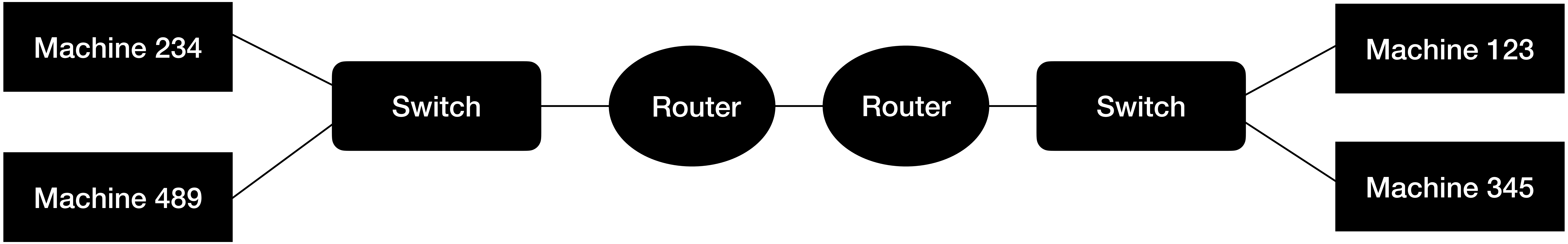
Hop to Hop connectivity

- Interacts with the physical medium
- Adds/takes data to/from L1 technology
- Responsible for creating a reliable link between two directly connected nodes, error correction, etc.
- L2's responsibility is mainly taking data from one hop to another
    - Uses an addressing scheme - MAC addressing
    - 48 bits represented as 12 hex digits
- L2 Technologies : NIC, WiFi access cards
- Switches are also L2 technology (help in moving data)



Communication is just more than Hop to Hop

What about communication from 234 to 345?

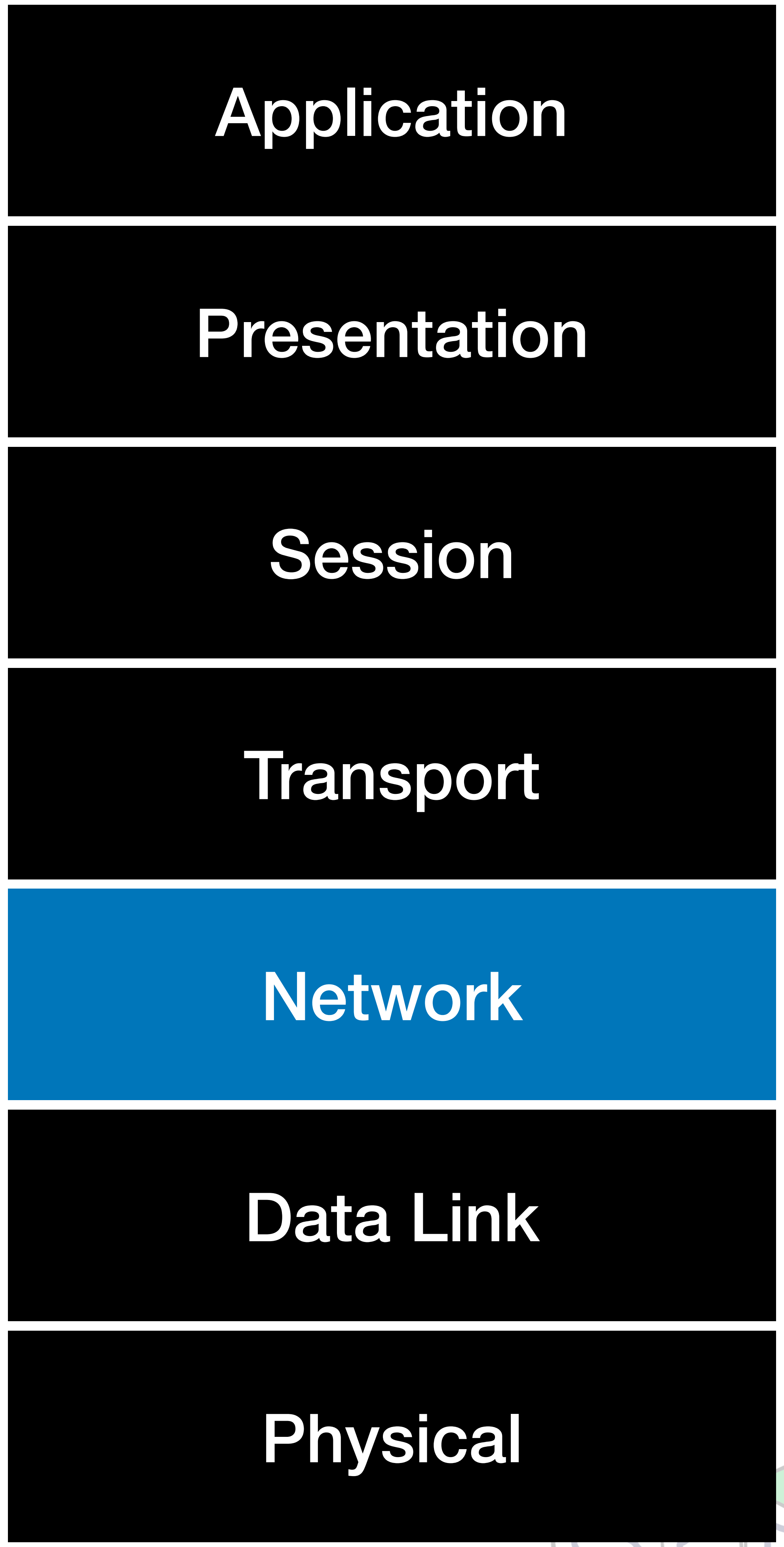


- Hop to hop is guaranteed by L2
- What about end-to-end delivery?

Network Layer (L3)

End-to-end Communication

- Manages routing through different routes in a large network
- Uses an addressing scheme - IP addressing
  - 32 bits represented as 4 octets (IPv4)
- Performs functionalities such as Logical addressing (IP), Path selection and packet forwarding
- L3 technologies: routers, even hosts are L3, L3 switches



Application

Presentation

Session

Transport

Network

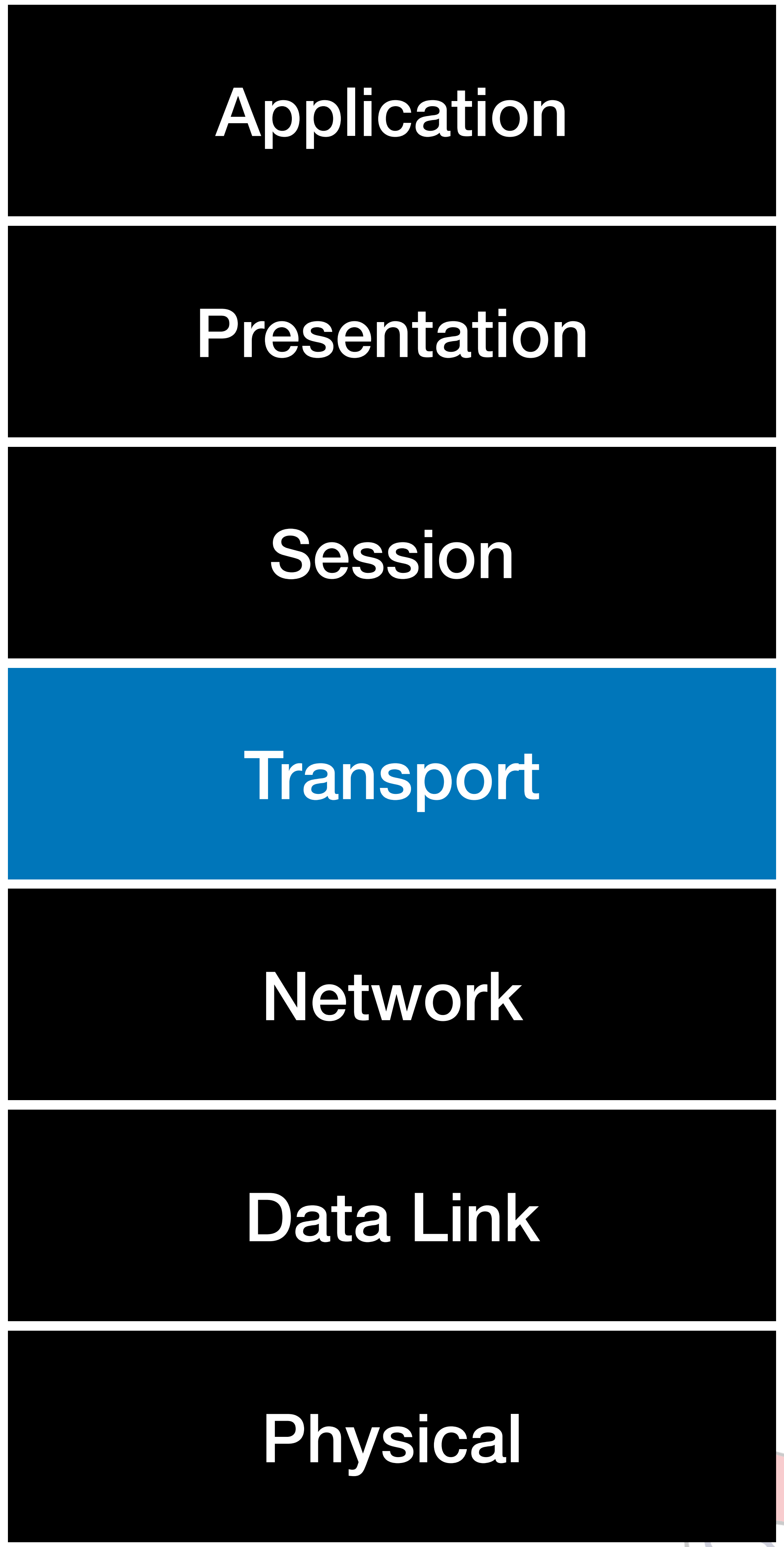
Data Link

Physical

Transport Layer (L4)

Service to Service

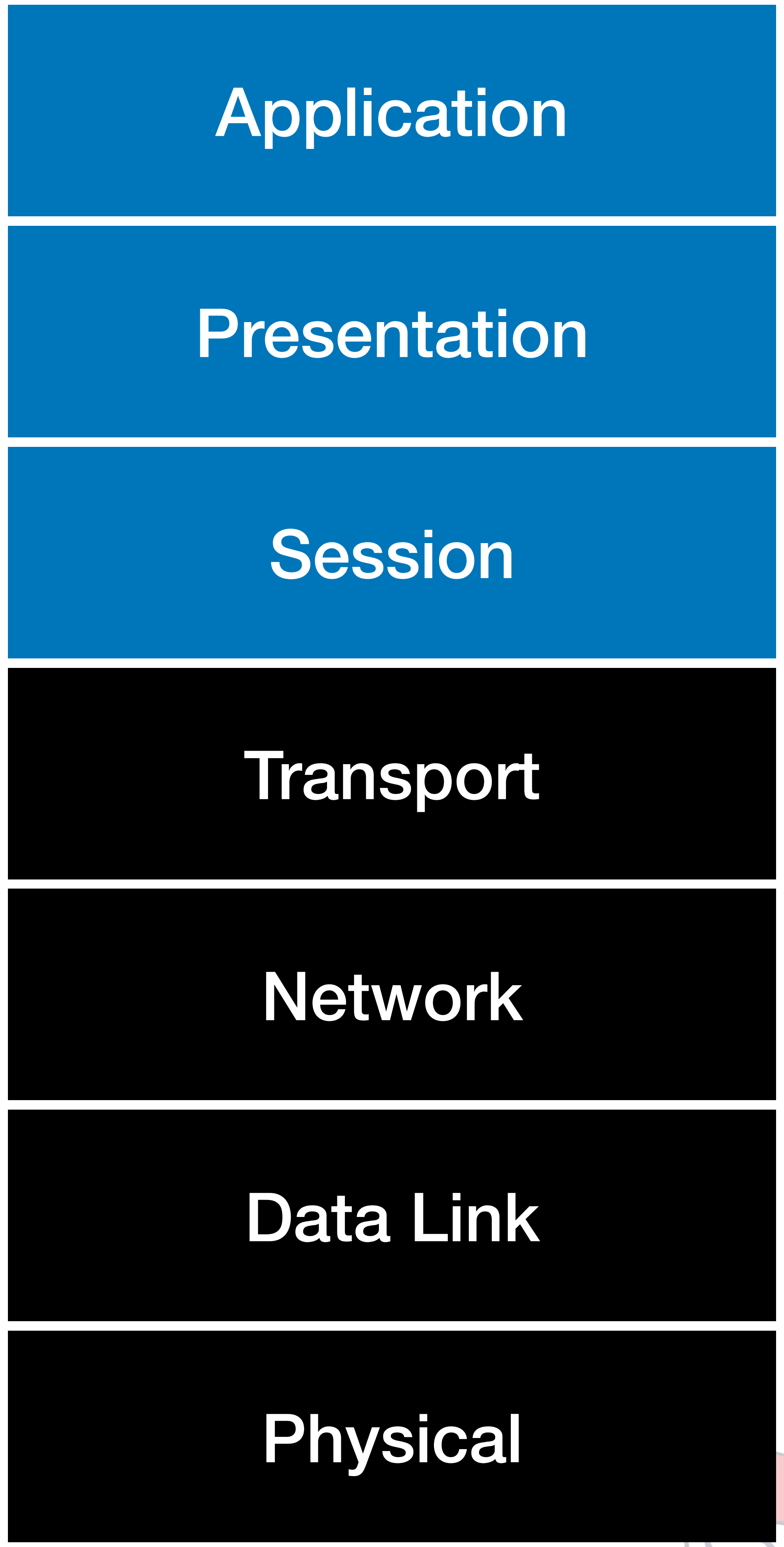
- How to ensure that the right process receives the data?
  - Many process will be executing or are waiting to execute?
- Ensures data transfer is reliable, sequential and free from others
- Manages flow control and error correction
- Layer 4 has an addressing scheme to guarantee message delivery
  - Ports! (0 - 65535), Privileged: 0-1023, Registered: 1024 - 49151
  - Every process will have a port through which sending or receiving data
- L4 Technologies: TCP, UDP



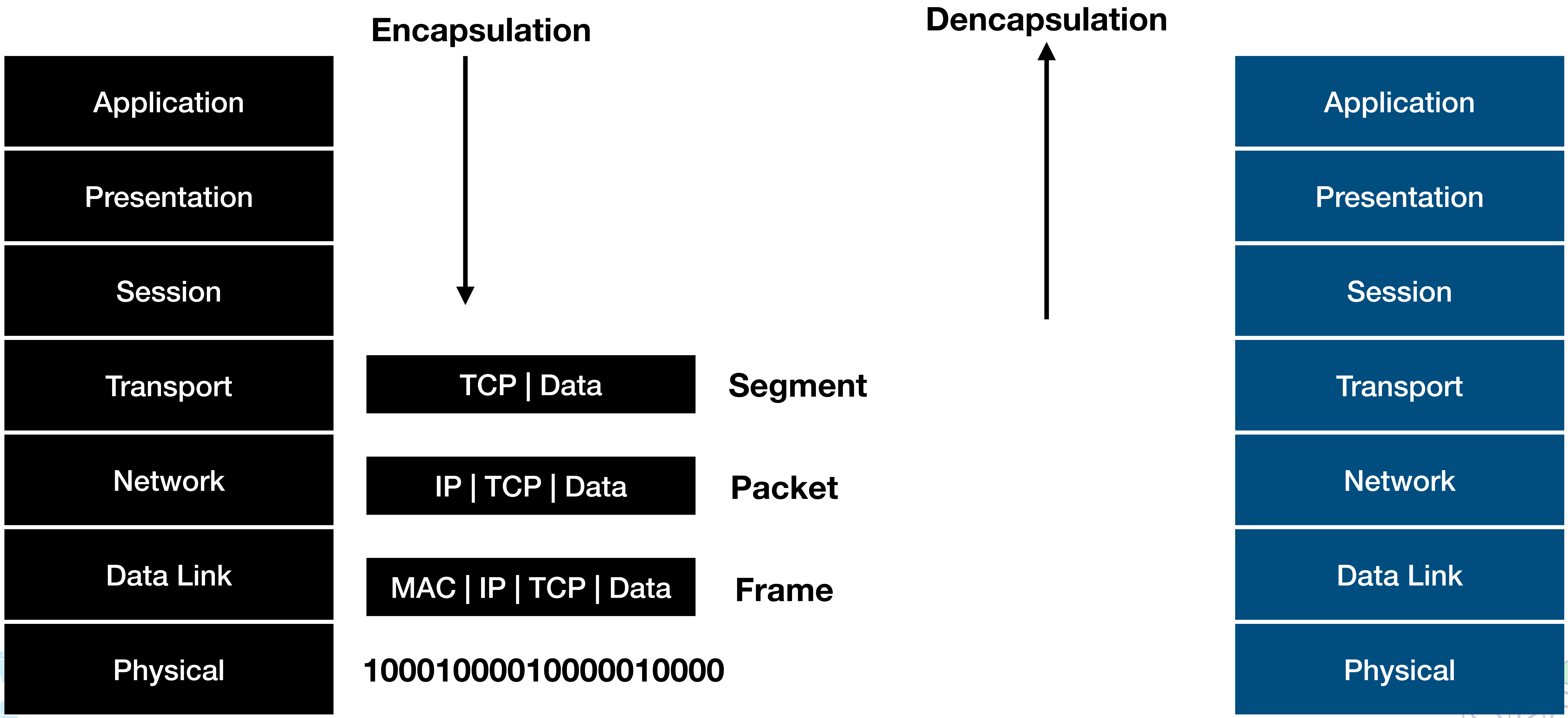
Session, Presentation and Application

Application to Application

- **Session Layer (L5)**
  - Manages connection between different devices
  - Establishing, maintaining and terminating connections
- **Presentation Layer (L6)**
  - Ensures that data is in format that sender and receiver can understand
  - Manages data encryption, compression
- **Application Layer (L7)**
  - Provides network services to the application processes
  - Eg: web browser, email clients, other softwares/apps



Putting It Together



Encapsulation

Dencapsulation

Application

Presentation

Session

Transport

Network

Data Link

Physical

TCP | Data — Segment

IP | TCP | Data — Packet

MAC | IP | TCP | Data — Frame

10001000010000010000

Application

Presentation

Session

Transport

Network

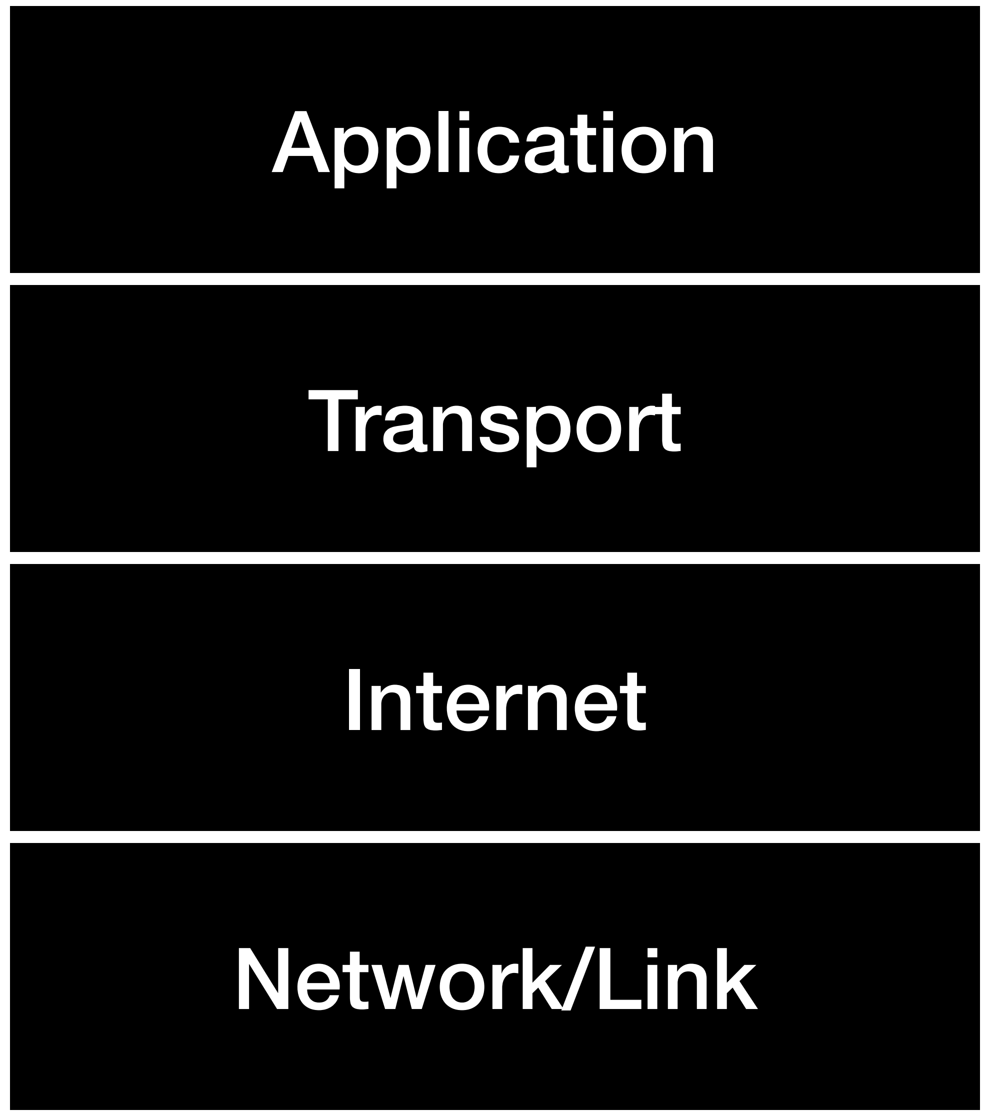
Data Link

Physical

The 4 Layer Model

Internet Model or TCP/IP model,

- OSI model is more educational purpose
- 4 layer model more used in reality
- Application layer - Corresponds to application, presentation and session
- Transport layer - Transport layer of OSI
- Internet layer - Network layer of OSI
- Network - Physical and data link layers of OSI



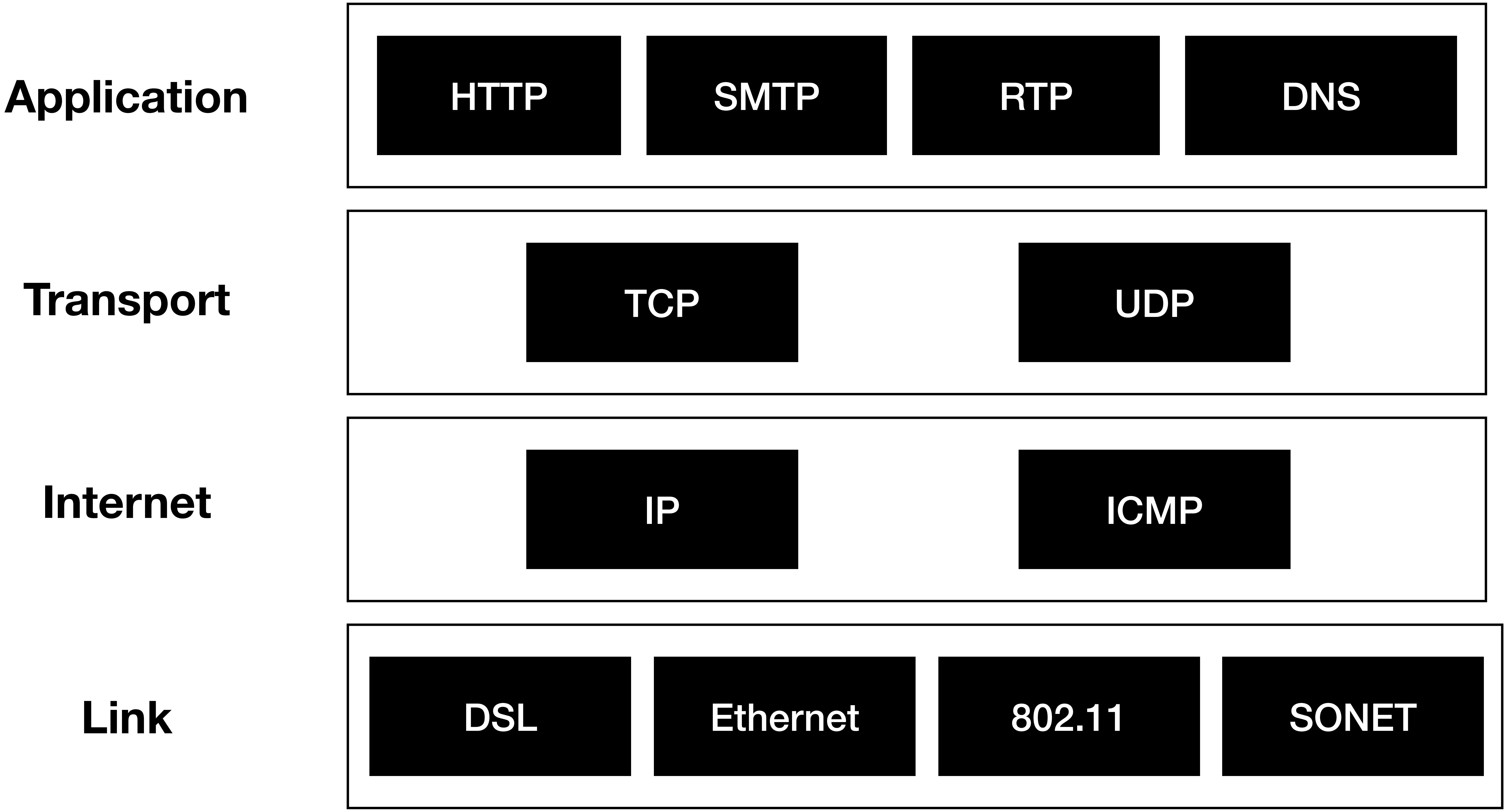
Application

Transport

Internet

Network/Link

Network Protocol Stack



Application: HTTP, SMTP, RTP, DNS

Transport: TCP, UDP

Internet: IP, ICMP

Link: DSL, Ethernet, 802.11, SONET

Onto Transport Layer

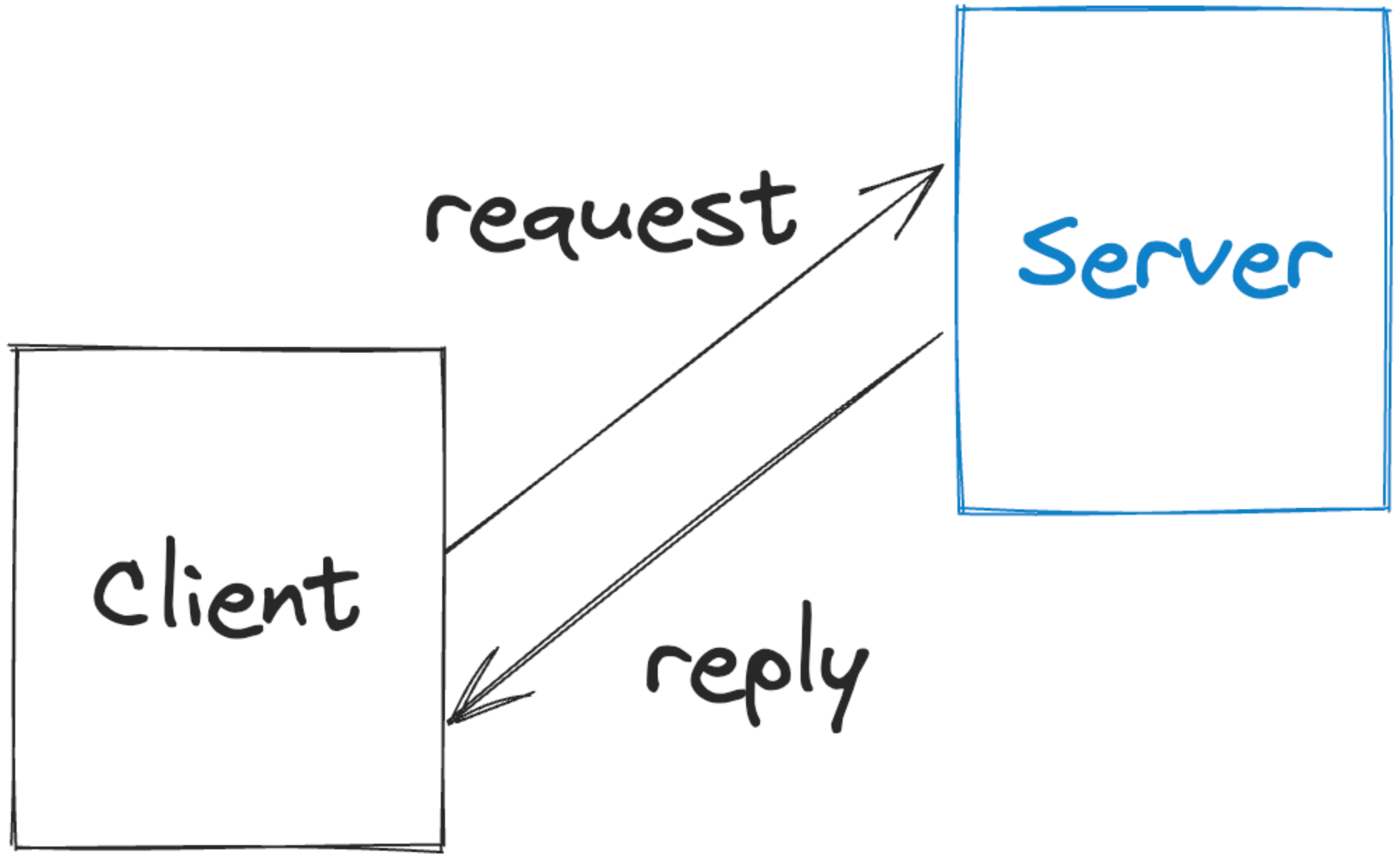
Service to Service Delivery

- **Multiplexing** and **Demultiplexing**
- Addressing scheme: Ports
- Two strategies/protocols that allows this
    - Transmission Control Protocol (TCP) - favours reliability
    - User Datagram Protocol (UDP) - favours efficiency

Lets go back

Two process wants to communicate

- Client sends a request to the server
- The server provides a reply based on the request made by client
- Eg: File transfer to get files
- Eg: web browsing, sent a url and get page
- How to transport data from process to process?



The role of Operating System

User space
kernel space
process
Kernel
protocol stack
drivers
Client
Network
process
Kernel
protocol stack
drivers
Server
User space
kernel space

- Software component in the OS that supports network calls - **Protocol stack**
- Provides Service primitives which are nothing but system calls - **Some API?**

Any idea on what should be some functionalities that should be made available?

Hint: Think of process API

The Socket API

- Simple abstraction to use the network
- The network service API used to write all network applications
- Part of all OS and all language [Berkley Unix, 1983]
  - Allows user space applications to interact with networking subsystem
- Two services:
  - Streams: Reliable (Connection-oriented)
  - Datagram: Unreliable (Connection-less)
- Allows applications to attach to the network at different ports

The Socket API

Function	Description
socket()	Creates a new socket of a certain type (depending on TCP or UDP) and returns file descriptor
bind()	Associates the socket with a specific IP and port
listen()	For server sockets, it allows sockets to listen for incoming connections
accept()	For server sockets, it waits for client to connect and then return a new file descriptor
connect()	For client sockets, it initiates a connection to a server.
send() / receive()	Transmit data or receive data
close()	Terminate the connection

SCAN ME

Thank you

Course site: karthikv1392.github.io/cs3301_osn
Email: karthik.vaidhyanathan@iiit.ac.in
Twitter: @karthi_ishere